INSTITUTO SUPERIOR DE TECNOLOGIAS AVANÇADAS

Projeto Global

Redes Informáticas Estruturadas

Virtualização

Aluno: Bruno Alexandre Duarte Marçalo | 2036

Orientador: Profº. Pedro Brandão

21/06/2017

--- Lisboa ---

Índice

Índice de Imagens

Resumo

Objetivo do Trabalho

O Objetivo deste projeto jaz na criação de uma rede empresarial, com base em tecnologias de Virtualização, esta devendo incluir as capacidades de albergar um sistema capaz de providenciar uma aplicação ADO.NET aos utilizadores tanto dentro da rede como exteriores à rede via VPN, estes que deverão ter acesso a dados armazenados de forma persistente numa base de dados relacionada à aplicação num servidor SQL.

Resultados Alcançados

Com a ajuda da Tecnologia de Virtualização da *Microsoft – Hyper-V*, foi construído um laboratório consistindo em:

- 2 Controladores de Domínio - para criação de um Domínio, ou seja, por poucas palavras, uma infraestrutura empresarial para gestão de recursos e utilizadores, sendo que o segundo funciona em modo de replicação;
- Servidor de *Routing* - para acesso\ligação ao exterior providenciando *Internet*;
- Servidor SQL - para armazenamento de informação em Base de dados relacionada com a aplicação inclusive a plataforma para a criação da mesma (*Microsoft Visual Studio*);
- Servidor NPS - que irá implementar um método de autenticação via VPN a clientes vindos do exterior da rede empresarial;
- Máquina cliente - que fará uso da aplicação, que à mesma terá acesso e permissão via ligação VPN.

Com este laboratório em prática e devidamente configurado, possibilita à nossa máquina cliente que é exterior à rede de domínio, ligar-se através de uma conexão VPN ao nosso Servidor NPS, que lhe irá conceder acesso ao mesmo. Estando esta ligada e pertencendo ao domínio, poderá executar a aplicação da empresa e interagir com a sua informação.

ABSTRACT

Project's Objective

The Objective of this project lies in the creation of an enterprise network, based in Virtualization technologies, whilst supporting a system capable of providing an ADO.NET Application to its network and outside-the-network users, this one's that should be privileged with the capacity of access to stored data within the database related to the application provided embedded in the SQL Server.

Achieved Results

With the help from Microsoft's Virtualization Technology – Hyper-V, a laboratory was built with the following:

- 2 Domains Controllers – to create a Domain, or so to say, an enterprise infrastructure to manage resources and users, noting that the second DC functions on replication;
- Routing Server – to access/connect to the outside providing Internet;
- SQL Server – to store information in the database related to the application including the platform of software development (Microsoft Visual Studio);
- NPS Server – will implement an authentication method by VPN that will connect users to the enterprise network;
- Client Machine – will make use of the application, that from which will have access and permissions by VPN.

With the lab implemented and properly configured, it enables the client machine which is on the outside of the domain network, to connect by VPN to our NPS Server, which will grant access to the before mentioned network. Being connected and belonging in the domain, will privilege the user to execute the enterprise application e interact with its data.

Introdução

Acredita-se que o conceito de virtualização teve as suas origens nos dias da *Mainframe* no fim dos anos 60, quando a IBM investiu uma grande quantidade de tempo e esforço em desenvolvimento de soluções *time-sharing* robustas. *Time-sharing* refere o uso partilhado de recursos de computação por entre um grupo de utilizadores, tendo como objetivo o aumento da eficiência de ambos os utilizadores e os recursos partilhados dos computadores.

A importância deste projeto é mostrar o quanto, desde o aparecimento da sua ideia, já esta tecnologia progrediu após vários anos, como é tão versátil e económico para as empresas de hoje em dia, visto que em conjunção com o conceito de *Cloud Computing*, esta apresenta uma facilidade enorme em aprovisionamento de recursos em meros minutos de demanda, sendo que em anos passados, quando ainda a virtualização estava nos seus dias de infância, uma empresa para obter mais recursos de computação, ou seja por exemplo, na aquisição de um novo servidor, este processo demoraria certamente uns meses. A virtualização vem também mostrar que, desde que o *Hypervisor* seja *high-spec*, uma máquina física apenas, possa albergar inúmeras máquinas virtuais (ex. Rede de uma empresa) poupando economicamente a longo prazo e ser uma grande alternativa "Green" que diminui a pegada de carbono da empresa.

Tendo mencionado o *Cloud Computing*, deve ser explícito que certamente este será o próximo passo no que toca a tudo o que é redes e empresas, com o seu grande pilar bem assente na virtualização, este conceito complementa em tudo o que a virtualização pressupõe e muito mais.

ESTADO DA ARTE

Abstract

This state of the art addresses the technologies that support the concept of cloud computing, which has in the past few years been emerging as a new way to host and deliver services over the internet. This new kind of technology is very efficient in eliminating the requirements needed for provisioning, and its characteristics allows enterprises to start using it slowly and expand its resources in accordance to service demand. However, even though this technology offers such huge benefits to the IT industry, its development is at its infancy, with 'who knows' how much issues to be addressed. Behind the cloud, magnificent technologies and tools are used to make it happen, such as the technology of virtualization (Hyper-V), databases (SQL and ADO.NET) and the operating system that will work on (Windows Server 2012-R2). In this document, I present the current state of research with the objective of providing a better understanding of cloud computing key concepts and what's behind it.

Keywords: Cloud Computing – Virtualization – Windows Server 2012 – ADO.NET – SQL

Introdução

O *Cloud Computing* representa um culminar de evolução de modelos TIC prévios, visto poder ter surgido devido a condições tecnológicas únicas, como a Internet de banda larga e a elevada capacidade de computação a custo reduzido. Com a *Cloud*, as TIC ganham outra importância dentro das empresas, sendo que estimulam o desenvolvimento, deixando assim as tecnologias de serem uma preocupação para antes serem instrumentos ágeis, fiáveis e de elevado impacto. O conceito de nuvem, mais teoricamente advém da eliminação da complexidade tecnológica e o aumento do impacto das TIC nas empresas, que de forma abstrata suporta vários cenários de utilização, sendo que dá resposta às necessidades das pequenas, médias e grandes empresas. Analisado em pormenor, o *Cloud Computing* é um conceito e não uma tecnologia visando a disponibilidade de recursos de computação a pedido com capacidade escalável, tudo desde aplicações a *datacenters*, pela internet à base de pagamento por uso.

Diante deste tópico, não podemos deixar de falar da virtualização, que é *software* que separa as infraestruturas físicas para criar vários recursos dedicados, e atua como impulsionador à computação em nuvem, não esquecendo que reduz custos às empresas enquanto aumenta a eficiência, utilização e flexibilidade do *hardware* existente. Temos também as bases de dados que providenciam capacidade de armazenar de forma organizada a informação da *Cloud*, denotando que uma das maiores preocupações neste campo é que as bases de dados incorporadas num sistema *Cloud*, tal como ela própria devem de ter a capacidade escalável dinâmica ou a habilidade de aprovisionar e descomissionar servidores a pedido.

O *Windows Server 2012 R2* faz um papel importante neste estudo porque é a base e uma das opções onde este software todo mencionado pode ser executado, com as ferramentas de virtualização (*Hyper-V* e *System Center Virtual Machine Manager*) incluídas ao sistema operativo, este é capaz de criar um ambiente de virtualização e geri-lo conforme a situação.

Windows Server 2012 R2

Quando o *Windows Server 2012* foi lançado, veio com a escolha possível entre duas edições (*Standard* ou *Datacenter*) e ambas permitiam ainda a preferência entre as duas versões (*Server Core* ou *GUI*). Com o lançamento da versão mais atual deste S.O, *Windows Server 2012 R2*, o leque de escolha expandiu, tendo sido disponibilizado mais duas edições (*Foundation* e *Essentials*).

- **Edição *Standard*** – É a edição principal deste S.O que representa um servidor de *Cloud* de nível empresarial, é rico em funcionalidades e permite satisfazer quase todas as necessidades de rede. Este servidor poderá ser utilizado para múltiplos propósitos ou funções individuais e ainda ser desmantelado até ao *core* para uma melhor segurança e desempenho;
- **Edição *Datacenter*** – Esta edição é virada para servidores de virtualização garantindo direito a instâncias virtuais ilimitadas, mostrando-se apenas pela única diferença em relação à *standard* apesar do custo ser quatro vezes maior;
- **Edição *Foundation*** – O *Foundation* pressupõe maior parte das funcionalidades, mas detém algumas limitações a ter em conta antes da sua implementação, tais como a função de serviços de certificados do *Active Directory* está limitado apenas a entidades certificadoras, existir um máximo de 15 utilizadores, número limitado de ligações a certos serviços, permissão limitada ao uso de apenas um *CPU Socket* e a impossibilidade de alojar máquinas virtuais;
- **Edição *Essentials*** – O objetivo deste servidor é acomodar organizações pequenas que detêm menos de 25 utilizadores e 50 computadores. Uma das características do mesmo é que se demonstra uma forma pouco dispendiosa de interligar uma empresa com a rede. Algumas funcionalidades novas para esta versão resumem-se na melhoria de implementação cliente, gestão de grupos de utilizadores, restauro do sistema e melhorias na implementação cliente e histórico de ficheiros.

(Minasi et al., 2014)

Com o aparecimento da *Cloud*, a *Microsoft* começou-se a dedicar a tal conceito e decidiu investir de forma mais avançada com o seu S.O mais recente (*Windows Server 2012 R2*).

Segundo (Microsoft Press, 2014), o *Windows Server 2012 R2* está no epicentro do sistema operativo de *Cloud* da *Microsoft* e oferece um servidor único e uma plataforma de *datacenter* que permite facilmente e de forma não dispendiosa, otimizar os negócios via *Cloud*. A *Microsoft* acredita que o S.O detêm um conjunto de qualidades que fazem jus às promessas de um *datacenter* e aplicações modernas, às quais são:

- **Classe empresarial** - O *Windows Server 2012 R2* oferece um *datacenter* de nível empresarial e uma plataforma *Cloud* que consegue aguentar enormes cargas de trabalho enquanto que possibilita, opções robustas de recuperação para proteção de informação contra interrupções de serviço;
- **Simples e não dispendioso** - O *Windows Server 2012 R2* disponibiliza capacidades de armazenamento resilientes e de rede para uma variedade alargada de cargas de trabalho. Consegue satisfazer estas capacidades por uma fração do custo de outras soluções, por métodos de aquisição de hardware industrialmente estandardizados. Adicionalmente o S.O simplifica a implementação de cargas de trabalho maiores e aumenta a eficiência operacional;
- **Focada em aplicações** - O *Windows Server 2012 R2* permite executar, implementar e escalar aplicações e *websites* rapidamente. Em sincronia com o *Windows Azure* e o *System Center 2012 R2*, desbloqueia a melhoria da portabilidade de aplicações entre infraestruturas *on-premises* para provedores de *Cloud,* aumentando a flexibilidade e elasticidade dos serviços de TI;
- **Centrado nos utilizadores** - O *Windows Server 2012 R2* permite-lhe dar poder aos seus utilizadores finais, dando-lhes acesso a recursos corporativos em dispositivos escolhidos pelos mesmos, enquanto que a sua informação estará sempre segura. Pode ainda gerir as identidades dos utilizadores na *Cloud*, fornecer acesso remoto, definir os recursos e nível de acesso do utilizador à informação com base em que são, ao que é que estão a aceder e vindo de que dispositivo. Pode ainda também gerir ambos dispositivos de empresa e pessoal via infraestrutura unificada, tornando mais fácil para os administradores identificarem e ajudarem a atingir conformidade.

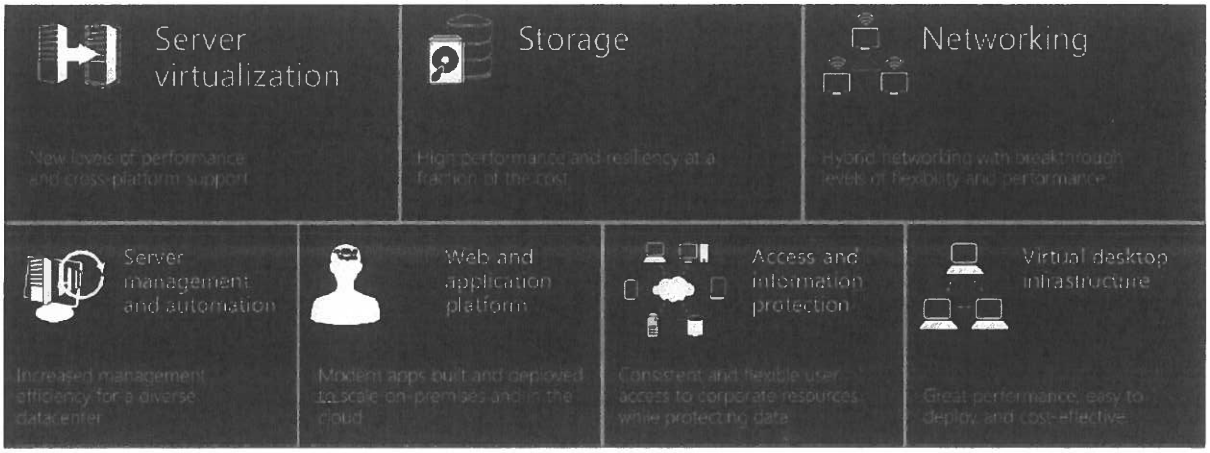

Figura 1 – Capacidades do *Windows Server 2012 R2*. (Microsoft Press, 2014)

Active Directory

Um S.O servidor necessita de um sistema de gestão de rede central automatizado para os dados dos clientes, segurança e recursos partilhados, tal esse já implementado desde a arquitetura do *Windows 2000*, conhecido por *Active Directory*.

(Tulloch & Team, 2013) refere que o *Active Directory* é a identidade central do servidor e destaca-se como a tecnologia de controlo de acesso mais usado por infraestruturas de TI empresariais. Na plataforma do *Windows Server*, esta consiste em serviços que desempenham 5 importantes funções:

- ***Active Directory Domain Services (AD DS)*** que oferece às organizações uma infraestrutura manejável, segura e escalável para a gestão de utilizadores e recursos;
- ***Active Directory Federation Services (AD FS)*** que oferece às organizações as capacidades *single sign-on* e uma simples e segura *identity federation*;
- ***Active Directory Lightweight Directory Services (AD LDS)*** que oferece uma implementação do *LDAP* para que as organizações consigam implementar o serviço diretório que não inclua dependências e restrições relacionadas com o domínio do *AD DS*;
- ***Active Directory Rights Management Services (AD RMS)*** que oferece às organizações, ferramentas de implementação e de gestão à encriptação, certificados e autenticação para a finalidade de deter soluções de proteção de informação viáveis;
- ***Active Directory Certificate Services (AD CS)*** que permite às organizações implementar infraestruturas de chave pública para que consigam pôr em ação e gerir

capacidades de criptografia, certificados digitais e assinatura digital aos utilizadores e máquinas.

Com a vinda do *Windows Server 2012 R2*, a Microsoft introduziu algumas novidades perante a versão anterior, isto para permitir a gestão de risco das TI enquanto a própria dá poder aos seus utilizadores para serem produtivos por diversos dispositivos:

- Administradores de TI podem permitir dispositivos se associarem com o *Active Directory* da organização e usar essa associação como um fator secundário de autenticação;
- Permitir que os utilizadores usem o método de autenticação *single sign-on* nos dispositivos que estejam associados ao *Active Directory* da organização;
- Permitir que os utilizadores acedam a aplicações e serviços a partir de qualquer lugar via *Web Application Proxy*;

Com o uso de Autenticação multi-fator e controlo de acesso multi-fator, é possível gerir o risco de os utilizadores trabalharem de qualquer lugar e aceder a informação protegida pelos seus dispositivos.(Microsoft Technet, 2014)

Hyper-V

Claro que sem a virtualização, a *Cloud* não era o que é hoje, até bem de longe. Assim a Microsoft em 2008, com essas prospetivas para o futuro decidiu não ignorar a tendência da virtualização lançando a tecnologia Hyper-V no Windows Server 2008, com o tempo foi melhorando a mesma até à última versão do S.O servidor (*Windows Server 20120 R2*).

O *Windows Server Hyper-V* é uma plataforma de virtualização sofisticada e rica em funcionalidades que assiste as organizações de qualquer grandeza a poupar custos consideráveis e a serem mais eficientes operacionalmente. Com o *Windows Server 2012 R2*, a virtualização com *Hyper-V* lidera a competição em termos de tamanho e escala industrial, tornando-se na plataforma de escolha para executar cargas de trabalho de valor crítico. Portanto se as organizações procurarem mobilidade de máquinas virtuais, aumento de máquinas virtuais disponíveis, suporte a ambientes "múltiplos inquilinos", aumento da sua escala ou até maior flexibilidade, o *Hyper-V* fornece-lhe a plataforma que necessita para aumento de agilidade empresarial. Escala a nível empresarial é uma característica da plataforma que providencia a capacidade de transformar o *datacenter on-premises* da organização numa Cloud sempre disponível e escalável. Outro ponto importante será a mobilidade de máquinas virtuais, o S.O

permite a gestão das máquinas virtuais independentes da infraestrutura física em que se assentam, o que permite manusear a alocação de recursos conforme a procura e a capacidade de configurar máquinas virtuais em execução.(Microsoft Press, 2014)

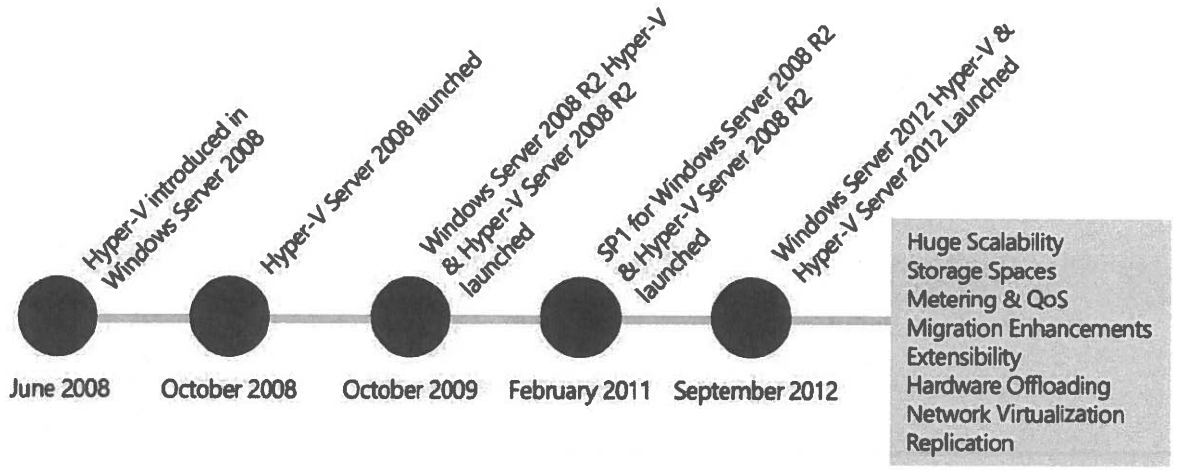

Figura 2 – Linha cronológica sobre o *Hyper-V* e capacidades aglomeradas. (Microsoft, 2016)

Como todas as funcionalidades do *Windows Server* são melhoradas a cada versão do mesmo, o *Hyper-V* não foge à exceção.

A última atualização para a família de servidores Windows Server 2012, trouxe uma data de novas e melhoradas capacidades do *Hyper-V*, às quais englobam escalabilidade, novas funcionalidades de rede e armazenamento, melhoras na migração ao vivo, *integração com hardware* aprofundado, replicação de máquinas virtuais, entre outras…; estas melhorias e novas funcionalidades podem ser agrupadas em 4 áreas importantes:

- **Escalabilidade, Desempenho e Densidade** – Os clientes estão constantemente à procura de executar máquinas virtuais maiores e mais poderosas, para satisfazer os picos de cargas de trabalho. Adicionalmente, à medida que a necessidade de hardware aumenta, os clientes querem tirar proveito dos maiores sistemas físicos para atingir níveis de densidade mais altos e reduzir custos no geral;
- **Segurança e "Múltiplos Inquilinos"** – Os *data centers* virtualizados estão a tornar-se mais populares e mais práticos nos dias de hoje. Organizações de TI e provedores de alojamento começam a oferecer "Infrastructure as a Service" (IaaS – Modelo de Implementação de Cloud), que providencia infraestruturas virtualizadas mais flexíveis aos clientes – Instâncias de servidor a pedido. Por causa desta tendência as organizações de TI e provedores de alojamento devem oferecer aos clientes segurança melhorada e

isolamento das mesmas, em alguns casos, até mesmo encriptação de informação para cumprir exigências;

- **Infraestrutura Flexível** – Num *data center* moderno, clientes procuram agilidade, de forma a responder rapidamente a mudanças da procura da empresa e de forma eficaz. A habilidade de mover cargas de trabalho flexivelmente pela infraestrutura é de alta importância, não esquecendo que os clientes ainda necessitam da possibilidade de escolha, aonde é que as cargas de trabalho devem de ser implementadas com base nas suas necessidades específicas.
- **Alta Disponibilidade e Resiliência** – Conforme a confiança dos clientes na virtualização aumenta e os mesmos virtualizam as mais importantes cargas de trabalho, a importância em manter estas cargas de trabalho continuadamente disponíveis aumenta significativamente. Para além do descrito anteriormente, a plataforma deve, em caso de desastre natural, ser capaz de rapidamente se restaurar noutra localização geográfica, ao qual enfatiza a importância que é a escolha da plataforma para um data center moderno.

(Microsoft, 2016)

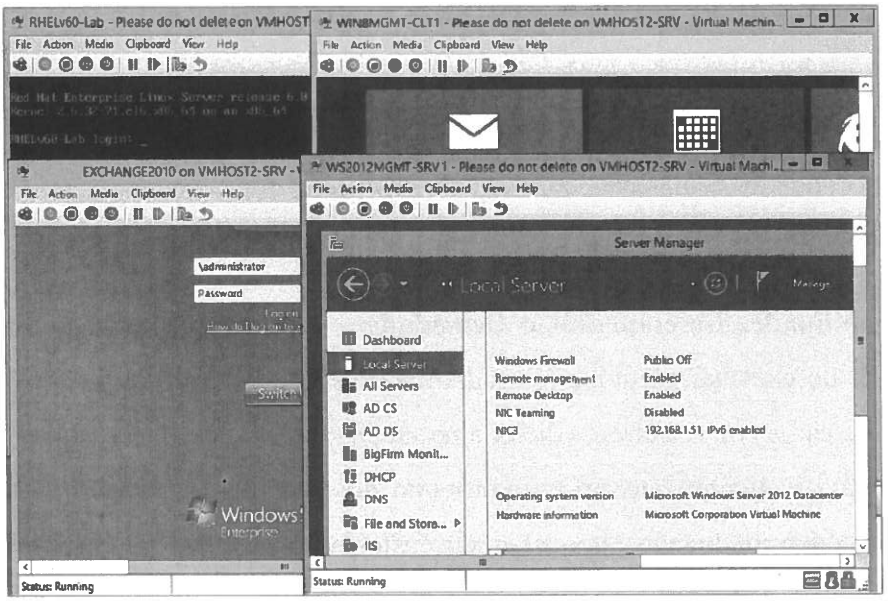

Figura 3 – *Hyper-V* a executar múltiplas máquinas virtuais. (Minasi et al., 2014)

System Center Virtual Machine Manager

Com o S.O como base de execução sobre o hardware e o Hyper-V como plataforma de virtualização de máquinas, só falta uma ferramenta para a gestão das mesmas, essa seria o SCVMM, desenvolvido exclusivamente pela *Microsoft* para os seus produtos.

O Microsoft SCVMM 2012 R2 oferece uma aproximação ao conceito de "múltiplos serviços num só sítio" para administração central da *Cloud* usando apenas uma interface gráfica para o utilizador. Este disponibiliza vastos serviços de gestão da *Cloud* e enorme paridade de funcionalidades em comparação às ofertas da concorrência, nomeadamente *VMware*. Apesar de ter inúmeras capacidades de monitorização e gestão de máquinas virtuais e físicas, este é maioritariamente utilizado para implementação em ambientes de virtualização com software da *Microsoft*. Esta função principal permite aos administradores rapidamente e de forma inteligente implementarem máquinas virtuais para variados usos, incluindo *Infrastructure as a Service* (IaaS) nas localizações de alojamento de preferência. A maioria das implementações é feita em anfitriões com *Hyper-V*, ainda assim o SCVMM suporta por completo a interoperabilidade entre VMware ESX e outros anfitriões. Este ainda detêm a automatização de armazenamento que é uma das funcionalidades do SCVMM que visa providenciar aos utilizadores um variado aglomerado de funções fulcrais incluindo a habilidade para descobrir, aprovisionar e comissionar recursos dentro de um ambiente de virtualização.(Johnson, 2015)

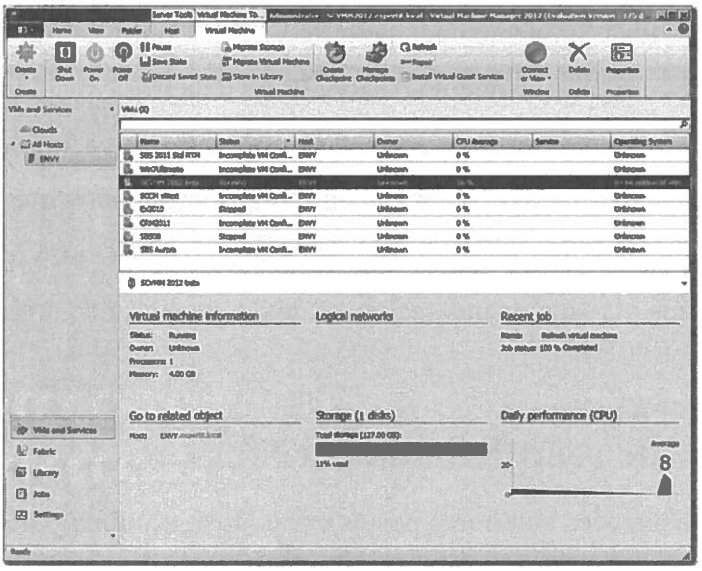

Figura 4 – Ambiente de Virtualização dentro do SCVMM 2012 (Schnackenburg, 2011)

Pela própria visão de (Ziembicki & Tulloch, 2014), o SCVMM pode ser usado para implementar e gerir *data centers* definidos por *software* dos seus componentes essenciais até às máquinas virtuais e a *Cloud*, implementação e gestão de aplicações e serviços a serem executados pelas máquinas virtualizadas.

Implementação de armazenamento definido por Software

O SCVMM poderá ser utilizado para implementar servidores *bare-metal* físicos, ou seja, sem S.O, incluindo configurações *pre-boot*, implementação de S.O e configuração pós-implementação. Isto inclui a configuração da funcionalidade de servidor de ficheiros que pioneira no estabelecimento de infraestruturas de armazenamento definidas por *software*. Assim que os servidores de ficheiros forem aprovisionados, o SCVMM pode criar um aglomerado de servidores de ficheiros escaláveis a partir dos mesmos e iniciar o processo de configuração de *pools* de armazenamento originários de vários espaços de armazenamento ao sistema associados para uma implementação completa de uma infraestrutura de armazenamento definida por software.

Implementação de computação definido por Software

Com a infraestrutura de armazenamento implementada, o SCVMM poderá ser utilizado para implementar servidores sem S.O e configurá-los como anfitriões *Hyper-V*, assim formando um aglomerado de anfitriões *Hyper-V*, para se ser configurado à utilização de uma infraestrutura de armazenamento gerida e implementada pelo SCVMM. Com esta implementação o tempo requerido para colocar em funcionamento o servidor de ficheiros e o aglomerado de anfitriões é reduzido significativamente ao qual aumenta a consistência da implementação aquando comparado com a longa lista de passos para a configuração necessária que teria de ser seguida, se fosse feita de modo igual, mas manualmente em todos os nós. O SCVMM permite também uma elasticidade rápida via adicionando aglomerados ou nós usando a mesma automatização quando necessária.

Implementação de rede definida por Software

Com as duas implementações anteriores postas em prática, a última parte de implementação poderá ser executada o que irá estabelecer uma infraestrutura de rede definida por *software*. Isto implica criar componentes de rede tais como *switches* lógicos, redes virtuais, entre outros...; Este passo poderá também incluir o adicionar de extensões de entidades terceiras ao *switches* virtuais do *Hyper-V* na infraestrutura anfitriã ou configurar certas capacidades como

NIC Teaming, QoS, port ACLs, entre outras...; outro passo crítico a esta implementação é a instalação de *gateways* de virtualização de rede, também totalmente automatizados pelo SCVMM, que permite a conectividade bidirecional a redes virtuais isoladas na infraestrutura. O SCVMM detém modelos de serviço que ajudam na implementação automática de máquinas virtuais para endereçar a funcionalidade dos *gateways* de virtualização de rede. Não esquecendo que este tem a capacidade de gerir endereçamento de IPs, ambos estáticos e dinâmicos, ou até mesmo integrar-se com o a funcionalidade IPAM do Windows Server 2012 R2.

Gestão definida por Software

Com as implementações todas instaladas, o SCVMM tem agora a capacidade de providenciar um número de funcionalidade adicionais. Visto de uma perspetiva técnica, o SCVMM suporta rastreamento e atualização em conformidade a pedido da infraestrutura, para além de poder monitorizar o estado de atualização dos servidores da infraestrutura, rastrear por conformidade e aplicar atualizações a servidores específicos. Este suporta atualizações automatizadas a aglomerados de anfitriões *Hyper-V*, que funciona por modo a meter os nós em modo de manutenção um a um e instalando o novo software. Um dos benefícios principais de um *data center* definido por *Software* é a capacidade de otimizar o uso da infraestrutura de um ponto de vista dinâmico sobre capacidade e poder. Funcionalidades do SCVMM como *Dynamic Optimization* permitem a migração ao vivo de máquinas virtuais entre aglomerados de anfitriões para melhoria do balanceamento de carga por entre anfitriões e de forma a corrigir alguma restrição colocada nas mesmas para otimização dos aglomerados baseados em políticas configuradas pelos administradores. Existe também o *Power Optimization*, que ajuda a gerir a eficiência energética ao desligar anfitriões num aglomerado ao qual não são precisos para atingir os requisitos de recursos para a demanda atual e volta a ligá-los quando necessário. Para além da infraestrutura, o SCVMM é também uma fundação à implementação de aplicações e serviços, incluindo serviços multicamadas complexos que consistem em múltiplas máquinas virtuais. Ainda assim o SCVMM consegue implementar máquinas virtuais individuais, ao qual as funcionalidades atuam apenas numa camada singular.

(Ziembicki & Tulloch, 2014)

SQL Server 2014

Com a vinda da *Cloud*, múltiplas oportunidades se elevam na área de bases de dados, nesse campo a *Microsoft* já trabalha desde 1988, na altura em parceria com a *Sybase* tendo desenvolvido a sua primeira versão de tal software para a plataforma OS/2. Contudo, o *SQL server 2014*, sendo a versão mais atual e estável, servirá como o SGBD que irá gerir as bases de dados relativas em conjunto com a metodologia de *Cloud* e assente no S.O da *Microsoft* anteriormente estudado.

O *SQL Server 2014* é a versão mais atual e estável da plataforma de informação da *Microsoft* que providencia às organizações e clientes uma plataforma consistente para infraestruturas, aplicações e informação que abrange os *data centers* do cliente, alojamento de *data centers* de provedores de serviços, e a *Cloud* pública da *Microsoft*. Os benefícios desta plataforma para os clientes incluem o desenvolvimento trivial, gestão, informação, identidade e virtualização independentemente de aonde a aplicação esteja a ser executada. O *SQL Server 2014* também oferece às organizações a oportunidade, de forma eficiente, proteger, desbloquear e escalar os próprios dados pelos diversos dispositivos e plataformas. Com o massivo sucesso do seu antecessor, este marca um grande impacto, devido às novas capacidades centradas na transformação da indústria das TI com a *Cloud*. Este consegue entregar o melhore dos desempenhos em missões críticas para as aplicações de base de dados mais exigentes enquanto que dispõe de mecanismos de segurança, escalabilidade, alta disponibilidade e de suporte um tanto complexos. Sendo que a sua missão é de forma rápida a entrega introspetiva de toda a informação, mas mais importante, a entrega de inteligência de negócios de modo consumível via ferramentas familiares como o *Excel*. Podemos também dizer que este servidor disponibiliza soluções únicas de *Cloud* Híbridas, sendo que estas soluções poderão afetar positivamente os standards mínimos de uma organização e permitir que a mesma crie soluções inovadoras para as suas aplicações de base de dados.(Mistry & Misner, 2014)

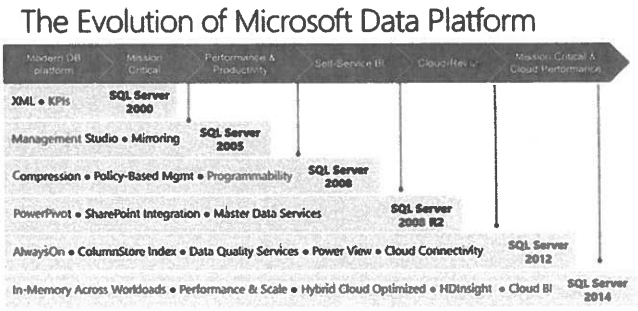

Figura 5 – Evolução do *Microsoft SQL Server* (Spilker, 2014)

Referindo de novo o enorme sucesso do seu antecessor, o *SQL Server 2012*, a *Microsoft* tendo recebido o *feedback* das organizações que ganharam confiança nas suas bases de dados e tempo disponível das mesmas, decidiu melhorar a sua obra-prima nas áreas de Disponibilidade, *Backup* e Restauro, Escalabilidade e Desempenho, e por fim Segurança.

Na área da **Disponibilidade**, os *AlwaysOn Availability Groups*, que é a capacidade de proteção de informação com recuperação a desastres e de alta disponibilidade que sucede ao permitir que uma ou mais bases de dados procedam ao *fail over* como uma unidade singular. No entanto esta funcionalidade foi melhorada na versão de 2014, tais melhorias são a redundância de dados aumentada, proteção e disponibilidade. Em primeiro, o número máximo de réplicas secundárias foi aumentado de quatro para oito, e em último este introduziu cooperação com o *Azure* em que permite que essas mesmas réplicas sejam armazenadas na *Cloud* pública do *Azure*. Outra funcionalidade melhorada foi os AlwaysOn FCI, que providencia proteção a nível de instâncias superior por uso de armazenamentos partilhados e *Windows Server FailOver Clustering*. O problema antecessor é que o requisito de um número unitário lógico impunha uma limitação, porque aquando um administrador de base de dados lhe se esgotava letras de disco ou *mount points*, este perdia também a oportunidade de alojar FCI adicionais, o que veio a ser corrigido na versão 2014, em que foi implementado o uso de CSVs (*Cluster Shared Volumes*).

Na área dos ***Backups* e Restauro**, existia a preocupação de que, independentemente do número de réplicas existentes das bases de dados detidas por uma organização, existe a necessidade de proteger essa informação com *backups*. Consequentemente, a *Microsoft* continuou a investir nesta área e desenvolveu melhorias com a versão de 2014 do *SQL Server*:

- **Gestão de *Backups* de *SQL Server* para o *Windows Azure*** – Os *backups* agora suportam nativamente o serviço de armazenamento em bolha do *Azure* para simplificar os *backups* para a *Cloud*. Os *backups* de *Clouds* híbridas reduzem as despesas capitais e operacionais e melhoram a recuperação devida a desastres, isto tudo porque as cópias de segurança são automaticamente replicadas para o armazenamento do *Azure* distribuídas por vários *data centers* à volta do mundo.
- ***SQL Server Backups* para *URLs*** - Os *backups* foram atualizados para poderem usar *URLs* como opção de destino aquando estes são executados com o estúdio de gestão. Não esquecendo que os *backups* guardados por este método são também armazenados na *Cloud* do *Azure*, visto que utiliza o serviço de armazenamento em bolha do *Windows Azure*.

- **Encriptação dos *Backups*** – Durante anos, os administradores vêm a pedir que o *SQL Server* tenha a capacidade de suportar nativamente a encriptação de informação enquanto cria as devidas cópias de segurança. Nesta última versão isso já é possível, podendo ser escolhido o algoritmo de encriptação e um certificado ou chave assimétrica para assegurar a chave de encriptação. Os algoritmos de encriptação que esta versão vem a suportar incluem AES 128, AES 192, AES 256 e 3DES, inclusive completo suporte à funcionalidade por um armazenamento *Azure* ou local.

Na área da **Escalabilidade e Desempenho**, a *Microsoft* investiu consideravelmente no motor de base de dados. Algumas das melhorias permitem às organizações reforçarem as suas cargas de trabalho do *SQL Server*, especialmente usando o *Windows Server 2012*:

- ***In-Memory LTP*** – Considerado uma das melhoras funcionalidades lançadas, esta encontrasse totalmente integrada na componente do motor do *SQL Server*. Esta funcionalidade vem permitir que as bases de dados OLTP se adaptem à memória, ao qual reduz a despesa de entrada e saída e aumenta a velocidade de desempenho de transações.
- **Recursos de Computação** – O *Windows Server 2012 R2* oferece uma quantidade enorme de recursos que providenciam escalabilidade a bases de dados, não só em ambientes físicos como virtuais também. Este vem suportar até 2048 processadores lógicos de um anfitrião *Hyper-V*, que consequentemente pode lidar com as maiores das aplicações de base de dados.
- **Rede Escalável** – O *Windows Server 2012 R2*, vem aprofundar a virtualização em rede providenciando um nível de abstração, que permite às cargas de trabalho do servidor sejam movidas entre *data centers*. Adicionalmente, com o uso do *NIC Teaming*, este poderá tolerar falhas ao ativar múltiplas interfaces de rede para trabalho cooperativo. *SMB Channel* e QoS poderão ser utilizados em conjunto para melhorar a disponibilidade das aplicações de bases de dados através de rede físicas ou virtuais.
- **Armazenamento Escalável** – O *Windows Server 2012* introduziu os espaços de armazenamento, posteriormente melhoradas no *R2*, em que estes permitem aos administradores das bases de dados tirarem proveito das melhorias de virtualização sofisticadas que conseguem distribuir cargas de trabalho pelos variados aglomerados de armazenamento. O *Windows Server 2012 R2* reconhece e otimiza o armazenamento ao colocar informação de alta consulta na camada mais rápida, e informação menos

utilizada em camadas inferiores, aumentado o desempenho de forma geral e sem aumentar custos.

- **Melhorias do Governador de Recursos** – Nesta última versão, as entradas e saídas foram adicionadas ao Governador de Recursos, que concede que as entradas e saídas sejam aglomeradas e hierarquizadas em conformidade com o critério das organizações.
- ***Buffer Pool Extension*** – Esta funcionalidade permite integração de RAM não-volátil para significativamente melhorar o desempenho das entradas e saídas, visando que usando SSDs (*Solid State Disk*) irá melhorar imenso o desempenho das consultas ao sistema. Alguns benefícios seriam a latência reduzida, taxa de transferência de transações aumentada, leituras mais rápidas e arquitetura de cache que se avantaja de discos de memória de pouco custo.
- **Melhorias do *Sysprep*** – Agora é totalmente suportado as implementações de instâncias de servidores aglomerados. Esta capacidade reduz o tempo de implementação e é importante na construção de *Clouds* públicas e privadas.
- **Melhorias do *Columnstore*** – Estes são usados para acelerar o desempenho das consultas. Com o *SQL Server 2014*, o processo foi melhorado em que a memória interna do *Columnstore* foi modificado para suportar operações de atualização, tais como, *inserts*, *udpates* e *deletes*.

Na área da **Segurança**, a *Microsoft* tem sindo uma das melhores plataformas seguras da indústria, confirmado pelo NIST (*National Institute of Standars and Technology*). Com a última versão, a demanda para a melhoria da segurança e conformidade baseado na sua sólida fundação continua. A *Microsoft* decidiu retirar a funcionalidade de um administrador de base de dados da funcionalidade de um administrador de sistema, visando melhor conformidade e segurança. Com isto, foi introduzido novas capacidades e controlo sobre segurança de nível empresarial melhoradas para permitir às organizações adequar políticas e regulações estritas de conformidade:

- Processo de segurança de engenharia redefinido;
- Certificação em CC de alto nível de segurança;
- Separação de deveres melhorada;
- Encriptação de informação transparente (TDE);
- Gestão de chaves de encriptação;
- Suporte ao *Core* do *Windows Server*.

Foi também introduzido permissões a nível de base de dados e servidor para segurança:

- CONNECT ANY DATABASE;
- IMPERSONATE ANY LOGIN;
- SELECT ALL USER SECURABLES;
- ALTER ANY DATABASES EVENT SESSION.

(Mistry & Misner, 2014)

ADO.NET

Nesta era da *Cloud*, a própria veio colocar as pessoas conscientes sobre a situação de exposição de informação em bruto em serviços e aplicações que queiram utilizá-la. E o ADO.NET vem providenciar isso mesmo, vem introduzir bibliotecas-cliente desenvolvidas para as plataformas da *Microsoft* que permitem o consumo fácil de serviços de informação nas suas aplicações.

ADO.NET é uma biblioteca de acesso a informação para programadores de .NET, sendo que é o coração de muitas tecnologias orientadas a informação da plataforma de desenvolvimento da *Microsoft*. Esta trabalha com C#, *Visual Basic* e outras linguagens com suporte a .NET. No entanto o ADO.NET é conhecido por ser uma família de tecnologias que permitem aos programadores de .NET interagirem com dados de formas standard, estruturadas e desconectadas. As aplicações escritas com o uso do .NET *Framework* dependem das suas próprias bibliotecas .NET, que existem em ficheiros DLL especiais que encapsulam o funcionamento de programação comum num formato fácil de aceder. Maior parte das bibliotecas fornecidas com a *Framework* .NET aparecem no *namespace* do sistema, por exemplo o *System.IO*, inclui classes que deixam interagir ficheiros do disco standard e fluxos de informação relacionados, *System.Security* que permite aceder a funcionalidades de encriptação de informação e ainda *System.Data* que implementa um pequeno conjunto de bibliotecas que tornam o consumo e manipulação de grandes quantidades de informação, simples e direto. O ADO.NET também gere dados internos e externos. Independentemente da fonte, este vulgariza informação relevante e apresenta-a ao seu código numa planilha – género de documento com organização de linhas e colunas.(Patrick, 2010)

Entity Framework

A *Entity Framework* providencia ajuda a um desenvolvedor em forma de uma metodologia de modelos que facilita a quantidade de trabalho que o próprio deve executar para criar uma apresentação realista dos dados. Para tornar as coisas ainda mais fáceis, a *Entity Framework* depende de uma apresentação gráfica dos dados para que o desenvolvedor possa literalmente ver a relação entre várias tabelas e outros itens da base de dados. Esta metodologia, parecendo simples, necessita de conhecimentos, mas trabalhar com estes modelos é definitivamente mais fácil do que trabalhar com ligações feitas à mão.

Quando um arquiteto quer desenhar um edifício por criação de plantas, uma das ferramentas que ele utiliza para se assegurar que a planta é precisa é a maquete/modelo. Estas maquetes/modelos ajudam a terceiros a perceber o que vai na cabeça do arquiteto. Adicionalmente, estes modelos ajudam o arquiteto a decidir se o plano é realístico. Da mesma forma, os desenvolvedores de *software* fiam-se nos modelos como maneira de perceber o *design* do *software*, determinar se o *design* é realístico e se se mostra capaz de fazer os outros perceber o seu *design*. A *Entity Framework* providencia os meios para criar vários tipos de modelos com que um desenvolvedor possa interagir de várias maneiras. Esta é uma extensão da tecnologia ADO.NET, que quando se cria o modelo da base de dados, também se está a fazer possível a criação automática de código requerido para ligação entre aplicações e base de dados. Devido ao modo como o ADO.NET e a *Entity Framework* interagem, torna-se possível criar *designs* extremamente complexos e desseguida usar esses mesmos *designs* diretamente do código de forma a que o programador perceba.(Fallis, 2013)

Definir uma entity

Uma *entity* é a informação associada a um objeto em particular. Por exemplo, um objeto cliente inclui o nome do cliente, endereço, número telefónico, nome da empresa, entre outros...; de uma perspetiva de administrador de base de dados, a *entity* seria uma linha de uma planilha com toda a informação do cliente. Esta inclui tudo o que a base de dados armazena fisicamente em tabelas separadas sobre um cliente em particular. Quando se dá a falar de *entities*, é necessário considerar as 3 visões sobre a informação:

- **Física** – As tabelas, chaves, índices, vistas e outras construções que suportam e descrevem a informação associada a um objeto da vida real. Todos os elementos são otimizados para tornar mais fácil para o DBMS armazenar e manipular a informação

de forma eficaz e confiável, sem erros. Assim sendo, um simples registo de cliente pode aparecer em múltiplas tabelas e requerer o uso de múltiplas chaves para criar uma vista coesiva do cliente. O armazenamento físico da informação é eficiente para o DBMS, mas difícil para o desenvolvedor entender.

- **Lógica** – Os elementos combinados requeridos para definir a informação usada com um objeto, tal como um cliente. De uma perspetiva de base de dados, a vista lógica da informação é esporadicamente encapsulada numa vista. A vista combina informação encontrada em tabelas usando chaves e outros elementos da base de dados para descrever as relações e ordem requerida para recriar o cliente com sucesso. Mesmo assim, a visão lógica de uma base de dados é de alguma forma abstrata e poderá causar problemas ao programador, sem mencionar que requer imenso código para gerir com sucesso. ADO.NET reduz a quantidade de código que o desenvolvedor executa via uso de objetos intrínsecos, mas o desenvolvedor deve ainda perceber a construção física por debaixo da informação.
- **Conceptual** – Visão real sobre a informação que se aplica ao objeto. Quando se vê um cliente, vê-se os atributos que o definem e itens que descrevem o cliente, tal como o seu nome. A visão conceptual da informação é apresentada de uma forma percetível, assim como os objetos se focam na informação e não na estrutura por debaixo da base de dados.

(Fallis, 2013)

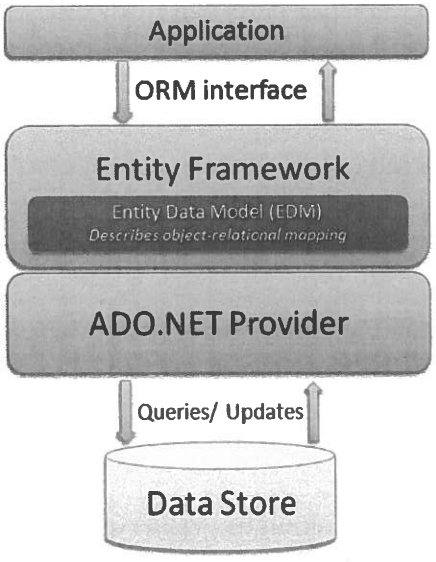

Figura 6 – Arquitetura da *Entity Framework* (DANRESA, 2017)

Virtualização

Na internet, o *Cloud Computing* tem um grande papel e para esta se suceder existe a necessidade de partilha de informação e uma das mais importantes tecnologias relacionadas à *Cloud* que é a virtualização. No entanto, o que é a virtualização?

Segundo (Salem, 2008), a virtualização é o separar, de um modo abstrato, os recursos de computação da implementação dos mesmos.

Porquê a virtualização de servidores?

A virtualização de servidores veio cortar custos ligados à energia e *outsourcing* diante de um cenário de recessão mundial, vindo em boa altura para as Organizações e departamentos de TI. Esta combate o problema da baixa utilização de recursos do servidor focados numa única aplicação que se proliferam por vários *data centers* pertencentes a médias e grandes empresas. Calcula-se que servidores em ambientes não-virtualizados têm uma taxa de utilização aproximada de 20%, o que demonstra o alto desperdício de energia que acaba por ser duplicado, via necessidades de refrigeração para mantimento de temperatura ideal ao seu funcionamento. Com o uso de virtualização, as taxas observadas aumentam até aos 80% visando não prejudicar tanto a vida útil do equipamento dos *data centers* por modos de gestão de múltiplas aplicações por servidor e partilha de recursos entre os mesmos.

Funcionamento da virtualização

Focado em aplicações

A virtualização de uma aplicação significa que a mesma é executada num ou mais servidores, tornando-os os únicos pontos de falha, que possam ter um impacto de elevada importância na realização de negócios. Antigamente, em sistemas de TI, quando o computador central falhasse, a empresa parava de funcionar, o mesmo se coloca aqui em questão, visto que a falha de um *router* ou *switch* poderá causar a falha de inúmeras aplicações vitais ao fornecimento dos clientes. Claro que atualmente, já existem ferramentas de gestão de rede que fornecem notificações instantâneas em casos semelhantes, mas não implicam que o funcionamento das aplicações não seja afetado, apenas servem como aviso à medida de ação rápida à correção.

Soluções de armazenamento e Virtualização

Num ambiente de múltiplos servidores, esta envolve migração de armazenamento, em que os discos são instalados nos servidores da mesma forma que se colocam no ambiente de trabalho

de dispositivos finais para diversos sistemas de armazenamento. Assim os discos trabalham em rede, fazendo parte da própria infraestrutura de rede da organização. Estas migrações detêm várias vantagens em termos de flexibilidade e segurança de informação. A desvantagem que este tipo de funcionamento demonstra, é que o acesso entre aplicações e informação terá de ocorrer pela rede, ao qual poderá causar atrasos que criem problemas às restantes aplicações no mesmo espaço partilhado. Neste ponto, a virtualização é caracterizada por não tolerar falhas de *switches* e sobrecargas de rede.

Aplicações de alta carga de trabalho para Virtualização

As aplicações que atingem altas cargas de trabalham, que colocam mais pressão computacional não provam ser compatíveis com a virtualização de um *hypervisor*, visando que poderá criar sérios problemas no ramo dos serviços. No entanto, identificar quais tais aplicações inadequadamente compatíveis devem permanecer em servidores dedicados, não é tarefa fácil, visto que terá de ser analisado o volume e as características das transmissões dessas aplicações apenas para ter uma ideia/pista para a identificação.

Virtualização e Gestão de recursos

A virtualização promete atender picos de demanda das aplicações com mudanças dinâmicas de recursos de computação e armazenamento, visando aumento da eficiência na alocação de recursos. Os ambientes não-virtualizados, têm como característica a utilização baixa de recursos dado cada aplicação necessitar de energia e memória suficiente para poder lidar com a sua carga de trabalho máxima mais recursos adicionais para eventuais crescimentos futuros. Resumindo, num ambiente não-virtualizado, cada aplicação terá de ter recursos suficientes para atingir os níveis prometidos de serviço para aguentar a demanda, mas isto significa que irá existir o sobre-aprovisionamento de recursos, que na maior parte do tempo não irão ser utilizados, o que irá levar a um tremendo custo não proveitoso. A virtualização vem colocar este problema nos eixos, prometendo a realocação dinâmica de recursos. Devido à elasticidade de serviço, existe a possibilidade de uma aplicação alocar mais recursos de computação, tais recursos de uma aplicação que detêm, de momento, baixa demanda, não necessitando dos mesmos. Em termos práticos, isto tudo só poderá ocorrer dentro de um *data center*, visto que atrasos de transmissão entre os mesmos iriam eventualmente causar problemas de desempenho. Mas se o provedor garantir ligações rápidas e com tolerância a falhas elevada, garantindo bom desempenho, o problema da descentralização da aplicação por vários *data centres*, não fica

longe, denotar que a tecnologia *Cloud*, nesta área, atua sinergeticamente com a virtualização, elevando a virtualização de aplicações a outro patamar.

(Paessler, 2013)

Mas, com isto tudo, para aplicar a virtualização a nível empresarial, é necessário cumprir 5 requisitos característicos de uma infraestrutura para virtualização:

- **Abstração** – Todos os tipos de recursos físicos disponíveis, tais como hiperligações, computacionais e de armazenamento são sistematicamente abstraídos para que sejam manipulados através de interfaces e alocados, para criar, modificar, reclamar e entregar uma "fatia", ou seja certa parte;
- **Isolamento** – Os recursos que formam uma "fatia", são isolados daqueles usados para outras para garantir que as "fatias" não se interfiram umas com as outras, em termos de desempenho, segurança e *namespace,* e ainda que qualquer "fatia" não cause perturbações à infraestrutura;
- **Elasticidade** – Os recursos que constroem uma "fatia" são alocados, reclamados e libertados elasticamente a pedido para que os operadores maximizem a acomodação de várias "fatias", otimizem o uso dos recursos tanto temporalmente quanto espacialmente e ainda permitir uso de recursos instantâneos tal como também o uso de recursos contínuo;
- **"Programabilidade"** – Os recursos que estruturam uma "fatia" podem ser programados para desenvolvimento, implementação e experimentação com novos protocolos de comunicação para disseminação de dados inovadores e para facilitar o processamento de dados eficientes a serem ativos na "fatia";
- **Autenticação, Autorização e Contabilidade** – O uso de recursos para criação de "fatias" deve de usar autenticação e autorização para que se atinga operações seguras de "fatias" que previnam o abuso de recursos e ataques mal-intencionados aos mesmos. É necessário contabilizar os recursos alocados da infraestrutura para que a integridade dos recursos possa ser examinada e monitorizada e o seu próprio uso otimizado.

(NVSG, 2012)

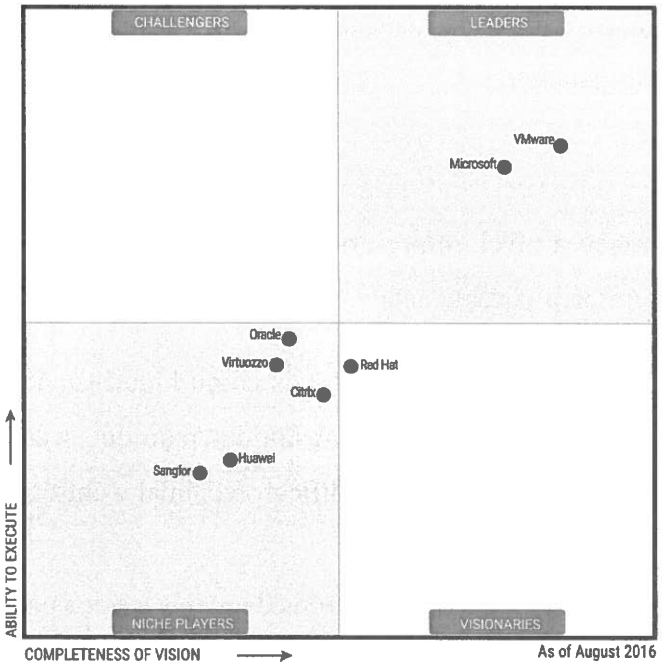

Figura 7 – Quadrante mágico da *Gartner* sobre Infraestruturas de Virtualização de servidores x86. (Bittman, Dawson, & Warrilow, 2016)

Cloud Computing

E finalmente, temos o *Cloud Computing*, considerado por muitos, uma verdadeira revolução industrial, que vem satisfazer da melhor forma as necessidades das organizações em relação ao serviço e custos de tecnologias de informação mais modernas. Este conceito da *Cloud* visa a utilização de recursos de computação que são disponibilizados pelas TI, mas abrange muito mais do que qualquer outra tecnologia específica, como por exemplo a virtualização, que a *Cloud* usa como parte da sua implementação.

Definição

O *Cloud Computing* é um modelo que permite o acesso ubíquo, conveniente e a pedido, através da rede, a um conjunto de recursos de computação partilhados (redes, servidores, armazenamento, aplicações, serviços, entre outros…), que podem ser rapidamente aprovisionados ou libertados, com um mínimo de esforço e sem interação com o fornecedor. (Mell & Grance, 2011)

Composição

O *Cloud Computing* é um paradigma sempre em evolução, sendo que respeita **5 características essenciais** que a compõem, e que de modo algum prescreve ou restringe algum método de implementação, serviço ou operação de negócios:

- **Acesso generalizado à rede** – As capacidades ou recursos estão disponíveis na rede e podem ser acedidos através de mecanismos comuns, tais como o browser ou aplicações, o que promove a utilização de plataformas clientes comuns, como computadores, portáteis, *tablets* e telemóveis. Isto significa que pode aceder aos seus serviços de *Cloud* utilizando qualquer tipo de dispositivo pessoal tecnológico moderno, focando no nível de conveniência ao qual os utilizadores podem aceder aos recursos físicos e virtuais.
- **Serviço medido** – Os sistemas de *Cloud* controlam e otimizam automaticamente a utilização de recursos através de mecanismos de medida, a um determinado nível de abstração de acordo com o tipo de serviço. A utilização de recursos pode ser monitorizada, controlada e reportada, providenciando transparência para o fornecedor e para o utilizador do serviço. Na *Cloud*, os modelos de faturação são pagos por utilização ou baseados em uso e, na maioria dos casos, o que foi usado só pode ser calculado no final de um determinado período. Esta característica foca-se no cobrar de apenas o que o cliente usa oferecendo um modelo de negócios altamente eficiente.
- **Autosserviço a pedido** – Um utilizador pode unilateralmente aprovisionar recursos de computação como, por exemplo, obter tempo de acesso a um servidor ou espaço de armazenamento na rede, sem interagir com outra pessoa, em particular, com um comercial ou técnico do fornecedor de serviços que está a utilizar. Sendo que é enfatizado a redução relativa aos custos, tempo e esforço necessário a tomar medidas.
- **Elasticidade rápida** – Os recursos são disponibilizados ou libertados de forma elástica, nalguns casos automaticamente, para escalar o serviço, consoante a demanda. Para o cliente, os recursos parecem ser ilimitados e podem ser solicitados em qualquer quantidade, em qualquer altura, sendo que esta característica visa que os clientes não necessitam de se preocupar com recursos limitados ou no planeamento de capacidade.
- **Acesso partilhado a recursos** – Os recursos de computação do fornecedor de serviços *Cloud* são partilhados por vários clientes, com recursos físicos ou virtualizados associados ou dissociados dinamicamente aos clientes, de acordo com o pedido dos mesmos. Há um certo sentido de independência da localização física, o cliente habitualmente não tem controlo ou conhecimento sobre a localização exata dos recursos, mas pode especificar a localização a um nível mais abstrato, tal como país, estado ou *data center*. Deduzindo o seu foco na capacidade de "múltiplos inquilinos" enquanto disfarça a complexidade do processo aos clientes.

Para além das características essenciais da *Cloud*, a própria detém a possibilidade de ser utilizada por diferentes **modelos de serviço**, que se dividem em **3 tipos de capacidades** e **7 categorias**. Os tipos de capacidade da *Cloud* classificam a funcionalidade providenciada pelo serviço ao cliente, com base nos recursos utilizados que diferem pelo princípio de separação de preocupações:

- **Tipo de capacidades de aplicação** – O cliente do serviço utiliza as aplicações do provedor;
- **Tipo de capacidades de infraestrutura** – O cliente do serviço pode aprovisionar e usufruir de processamento, armazenamento e recursos de rede;
- **Tipo de capacidades de plataforma** – O cliente do serviço pode implementar, gerir e executar aplicações personalizadas ou de terceiros usando múltiplas linguagens de programação e variados ambientes de execução suportados pelo provedor.

Já as categorias, são um grupo de serviços que possuem certas qualidades comuns. Sendo que uma categoria pode incluir capacidades de múltiplos tipos de capacidades. Eis as categorias:

- **CaaS (Comunicações como serviço)** – Categoria em que o serviço providenciado ao cliente é interação e colaboração em tempo real;
- **CompaaS (Computação como serviço)** - Categoria em que o serviço providenciado ao cliente é o aprovisionamento e uso de recursos de processamento necessários para implementar e executar *software*;
- **DSaaS (Armazenamento de dados como serviço)** - Categoria em que o serviço providenciado ao cliente é o aprovisionamento e uso de armazenamento de dados e capacidades relacionadas;
- **IaaS (Infraestrutura como serviço)** – Categoria em que tipo de capacidades providenciada é do tipo infraestrutura. Este tipo de serviço dá a capacidade ao cliente de instalar recursos fundamentais de computação. Neste caso, o cliente não gere nem controla a infraestrutura, mas tem controlo sobre o S.O, aplicações e possivelmente a rede;
- **NaaS (Rede como serviço)** - Categoria em que o serviço providenciado ao cliente é o transporte de conectividade e capacidades de rede relacionadas;
- **PaaS (Plataforma como serviço)** - Categoria em que tipo de capacidades providenciada é do tipo plataforma. Este tipo de serviço dá a capacidade ao cliente de colocar na *Cloud*, aplicações criadas ou adquiridas pelo próprio, que usam linguagens

de programação, bibliotecas de funções, serviços e ferramentas suportadas pelo provedor. Neste case, o cliente não gere nem controla a infraestrutura, incluindo a rede, servidores, S.Os, armazenamento, mas tem controlo sobre as aplicações instaladas pelo próprio e possivelmente de algumas configurações do ambiente de alojamento;

- **SaaS (*Software* como serviço) -** Categoria em que tipo de capacidades providenciada é do tipo *software*. Este tipo de serviço dá a capacidade ao cliente de aceder a uma aplicação que corre numa infraestrutura de computação do provedor. Neste caso, o cliente não gere nem controla a infraestrutura de computação, incluindo a rede, S.Os, espaço em disco, memória, entre outros...; que se encontrem associados à aplicação, mas pode controlar aspetos relacionados com a configuração de utilizadores da aplicação.

Ainda no tópico da composição da *Cloud*, esta pode-se dividir em **4 modelos de implementação**, que definem a forma como o serviço irá ser prestado, baseando-se no controlo e partilha de recursos físicos ou virtuais. Eis os modelos:

- ***Cloud* pública** – Este modelo caracteriza-se pela infraestrutura ser disponibilizada para uso do público em geral visando utilização aberta. Uma *Cloud* pública pode ser detida, gerida e operada por uma empresa, organização académica, governamental ou uma combinação destas. Esta está fisicamente localizada nas instalações do provedor.
- ***Cloud* privada** – Este modelo caracteriza-se pela infraestrutura ser disponibilizada a uso exclusivo de uma única organização, em que os próprios recursos são controlados pelo cliente. Pode ser detida, gerida e operada pela própria organização ou uma terceira entidade e poderá existir nas instalações da organização ou não.
- ***Cloud* de comunidade** – Este modelo caracteriza-se pela infraestrutura ser disponibilizada para uso exclusivo de uma comunidade de utilizadores que tenham partilhado requisitos e afinidade, e onde os recursos são controlados por apenas um ou mais membros da comunidade. Esta pode ser detida, gerida e operada por uma ou mais organizações envolvidas na comunidade, uma terceira entidade ou alguma combinação entre estas. Assim que pode localizar-se nas instalações afiliadas à comunidade ou num *datacenter* externo.
- ***Cloud* híbrida** – Este modelo caracteriza-se pela infraestrutura usar pelo menos dois tipos de modelos de implementação diferentes. As implementações envolvidas permanecem entidades únicas, mas estão interligadas por tecnologias standard ou

proprietárias, que permitem a portabilidade dos dados ou aplicações. Pode ser detida, gerida e operada pela própria organização ou uma terceira entidade e poderá existir localmente nas instalações ou externamente em *datacenters*.

(Ferreira, 2015; ITU-T, 2014)

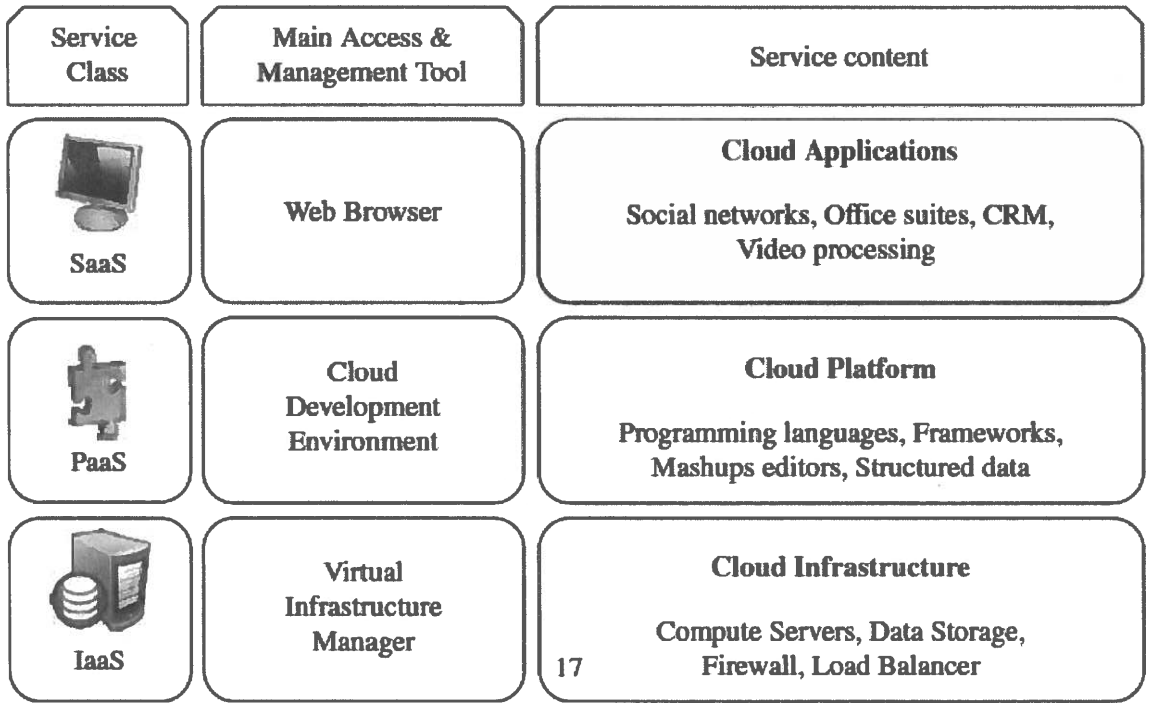

Figura 8 – Modelos de implementação mais importantes e respetivas associações. (Buyya, Broberg, & Goscinski, 2011)

CAPÍTULO I

Contextualização

Virtualização

A Virtualização veio redefinir os limites das Tecnologias de Informação, sendo uma tecnologia recente é bastante dispendiosa no seu investimento, mas a longo prazo esta dá os seus "frutos" económicos e amigáveis do ambiente. Hoje em dia, a maioria das empresas já pensou ou pensa em migrar para um ambiente virtualizado, isto porque no topo das preocupações está as pressões do custo, processo de aprovisionamento de servidores mais rápido e a própria consolidação de servidores. Estas preocupações mencionadas provavelmente são o motivo pelo qual os administradores de Tecnologias de Informação optam por complementar a virtualização com outras infraestruturas já existentes na empresa e pouco a pouco aumentar a sua infraestrutura virtualizada. Mas um dos fatores mais importantes é humano. O mundo das Tecnologias de Informação está sempre em mudança e por contrário à natureza humana, que somos criaturas que criamos hábitos ficando relutantes à aceitação da mudança, não é fácil implementar este tipo de novas tecnologias, falando em específico da Virtualização, este conceito traz grandes mudanças à vida empresarial visto ter um carácter de abstração do hardware físico de uma infraestrutura, possivelmente completa. Contudo esta tecnologia tem um grande grupo de vantagens que justifica o seu investimento, impulsionando em quem nele investe, um grande passo no futuro, já com olhos no Cloud Computing e os serviços que o mesmo oferece.

Projeto Global – Pós-Graduação

Exposto isto, a virtualização está inserida como tema principal na Pós-Graduação em Virtualização e Cloud Computing lecionada no ISTEC (Instituto Superior de Tecnologias Avançadas), onde existe uma parceira em colaboração com a execução deste projeto global, pela coordenação do Profº Pedro Brandão e leccionamento do mesmo, do Profº Henrique Guerreiro, Profº Pedro Crispim e Profº Assistente André João. Na pós-graduação, contamos com o estudo desta tecnologia de forma aprofundada e prática ao qual nos facilita a compreensão e aplicação de conhecimentos no presente projeto.

CAPÍTULO II

Desenvolvimento do Projeto

Especificações da máquina física

Para execução de projetos que lidam com virtualização, é necessário que o equipamento físico em questão cumpra certos requisitos para poder suportar o espetro completo das necessidades do projeto, eis as especificações da máquina utilizada para este projeto:

- Sistema Operativo
  - Windows 10 *Professional.*
- Processador
  - Processador de 64-bit com *Second Level Address Translation* (SLAT);
  - Suporte do processador a *VM Monitor Mode Extension* (VT-c em CPU's da Intel).
- Memória
  - 16 GB RAM.
- Armazenamento
  - 220 GB de espaço (HDD 7200rpm).
- Mudanças na BIOS
  - Tecnologia de Virtualização ativado;
  - *Secure Boot* desativado.
- Software (e Licenças associadas)
  - Imagens ISO
    - *Windows Server 2016 Datacenter*;
    - *Windows Server 2012 R2 Standard*;
    - *Windows 8.1 Pro.*
  - SQL Server 2014 Express with Advanced Services
  - Visual Studio 2017 Community Edition

Esquema Detalhado

Esquema de suporte aquando ao Desenvolvimento do projeto.

Contas

PG\Administrador - P@ssw0rd //DC1 / DC2 / NPS / Routing Server

PG\sql_admin - P@ssw0rd //SQL Server

PG\Cliente1 - P@ssw0rd //Cliente1

Máquinas – Rede/Função/Espaço/SO/Licença

Todas as placas de rede têm IPv6 desativos!

192.168.0.10 - DC1 //32GB W2k12R2 - 3FMX8-NGMWY-M9KW9-R96PT-WW98M

192.168.0.11 - DC2 //32GB W2k12R2 - DBGBW-NPF86-BJVTX-K3WKJ-MTB6V (AVMA)

192.168.0.15 - Routing Server //32GB W2k12R2 - 3FMX8-NGMWY-M9KW9-R96PT-WW98M

192.168.0.20 - SQL Server //32GB W2k12R2 - DBGBW-NPF86-BJVTX-K3WKJ-MTB6V (AVMA)

192.168.0.25 - NPS //32GB W2k12R2 - DBGBW-NPF86-BJVTX-K3WKJ-MTB6V (AVMA)

192.168.0.30 - PC1 //20GB W8.1PRO - XP694-KN43Q-6PXPR-3HBGF-YTB9B

VPN

50.50.50.1 - Placa VPN - NPS Server

50.50.50.2 - Placa VPN - PC1

Pastas Partilhadas

DC1\Pasta PG //Partilhado pelos computadores todos

SQL\AppDados //Partilhado com a máquina cliente

Guia de Desenvolvimento

Preparar a máquina Hypervisor

1. Deve-se instalar o *Hyper-V* na máquina local. Vamos ao **Painel de Controlo** e clicamos em **Programas e Funcionalidades**, na aba esquerda selecionamos em **Ativar ou desativar funcionalidades do Windows** e finalmente instala-se o *Hyper-V* colocando um visto na caixa correspondente à funcionalidade.

Figura 9 – Instalação da componente *Hyper-v*.

2. Reinicia-se o computador aquando pedido.
3. De volta com a máquina ligada vamos ao ambiente de trabalho, abrimos o **Gestor de Hyper-V**, nas **Ações** clicamos em **Novo** -> **Máquina Virtual**.
4. Clicamos em **Seguinte** na primeira página, damos o **Nome** de "**Hypervisor**" e define-se o lugar apropriado no computador para alojar a máquina onde tenha mais de 220GB de espaço livre. Clicamos em **Seguinte** novamente.
5. Definimos a máquina virtual como **Geração 2** e seguimos clicando **Seguinte.**
6. Na atribuição de memória, só mesmo por salvaguarda, definimos o **máximo de memória** do computador no arranque e a utilizar **Memória Dinâmica**. Clicamos em **Seguinte**.
7. Podemos saltar o passo da **Rede** clicando **Seguinte**.
8. Neste passo, deixamos selecionado a opção de **Criar um disco rígido virtual**, definimos o nome de "**PG-HYPER-V-A**", a localização deverá automaticamente apontar para onde se definiu a localização das máquinas virtuais momentos atrás e definimos também o tamanho para **220GB**. Clicamos em **Seguinte**.

9. Selecione a opção para **Instalar um sistema operativo a partir de um ficheiro de imagem de arranque** e indique a localização do ficheiro de imagem para o Sistema Operativo ***Windows Server 2016 Datacenter***. Clicamos em **Seguinte**.
10. Nesta última página de **Resumo**, clicamos em **Concluir.**
11. Ligamos a máquina virtual e acedemos a ela, neste processo de instalação do Sistema Operativo deveremos instalá-la por padrão, mas com os seguintes parâmetros aquando da mesma:
    a. **Linguagem de Sistema e Teclado**: Português (Portugal);
    b. **Password**: P@ssw0rd;
12. Depois dos vários reinícios e de finalizar a instalação, encerramos a máquina virtual.

Fazer o VHDX do Hypervisor bootable

1. Para colocar a máquina *hypervisor* como opção de boot do *Windows*, temos de ir à **Gestão de Discos**, para isso clicamos com botão direito do rato no ícone do ***Start*** e selecionamos **Gestão de Discos**.
2. Na barra de topo, selecionamos **Ação -> Expor VHD** e indicamos o caminho onde foi guardado o VHDX (disco virtual) da máquina *hypervisor*.
3. Após o disco virtual ser reconhecido na **Gestão de Discos** e denotado que a letra do disco é **E:,** abrimos uma **Linha de Comandos** com privilégios de administrador e escrevemos o seguinte comando:
    a. **bcdboot E:\windows**
4. Desseguida, abrimos o **Executar** e escrevemos "**msconfig**" e premimos **Enter**.
5. Vamos até à aba **Arranque (Boot)**, pré-definimos o Sistema Operativo corrente e aumentamos o tempo limite para escolha da opção *Boot* para 60 segundos.

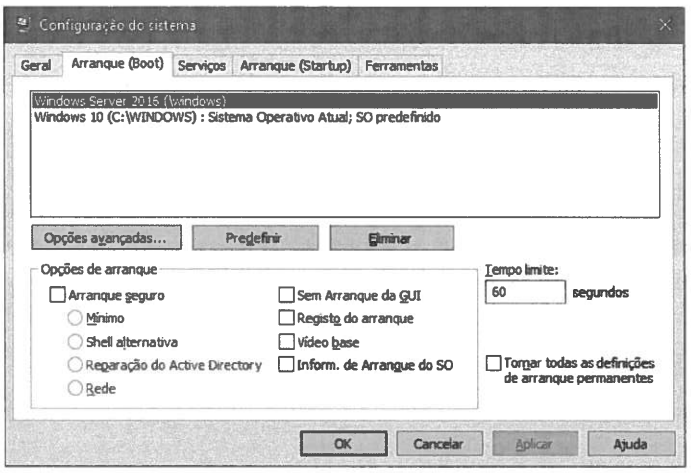

Figura 10 – Alteração das definições de arranque no *msconfig*.

6. Premimos **OK**, e reiniciamos o computador aquando pedido.
7. Quando o computador inicia, agora apresenta um menu de *boot*, possibilitando a escolha entre o vhdx criado e o Sistema Operativo nativo. Escolhemos então o **Windows Server 2016**.

Configuração do Hypervisor

1. Quando entramos com a conta de Administrador local, devemos agora **atualizar completamente o Computador**, isto incluindo ***Windows Updates*** **e** ***Drivers***, no nosso caso de estudo, foi utilizado os drivers para o ASUS GL552VW (Asus Drivers), que é a máquina em questão. (**Repetir a execução de *Windows Updates* até não existir mais atualizações importantes**)
2. Após ter a máquina completamente atualizada, iremos instalar novamente o Hyper-V, mas agora trata-se de uma instalação em servidor. Automaticamente no arranque do Sistema Operativo o **Gestor do Servidor** é iniciado após alguns segundos, quando aberto iremos a **Gerir -> Adicionar Funções e Funcionalidades**.
3. Na primeira página de **Antes de Começar**, clicamos em **Seguinte**.
4. Vamos premindo **Seguinte** até chegar à parte de **Selecionar Funções** e aí escolhemos a opção de **Hyper-V** e desseguida **Adicionar funcionalidades** se for pedido.
5. Clicamos **Seguinte** até aos **Switches Virtuais**, nesta página selecionamos a **placa de rede Ethernet**, passando a **Migração** à frente encontramo-nos no **Armazenamento pré-definido**, aqui devemos definir ambos os campos para guardar as máquinas virtuais e configurações no ***Root*** **do disco**.

6. Avançamos até ao final do Assistente de Configuração e clicamos **Instalar**. Selecionamos o botão de **reinício se necessário**, e aguarda-se o fim da instalação.
7. Reiniciamos o computador se for pedido pelo computador.
8. Após o computador iniciar e fizermos o *login* com a conta de Administrador Local, abrimos o **Gestor de Hyper-V** para configurar os nossos Switches virtuais para a rede empresarial.
9. Vamos a **Ações -> Gestor de Comutadores Virtuais** e abrirá uma nova janela.
10. Primeiro, para criar uma rede privada para o domínio, clicamos em **Novo comutador de rede virtual -> Privado -> Criar Comutador Virtual**.
11. Damos o nome de "**Privado**", selecionamos a opção de **Rede Privada** e por fim clicamos em **Aplicar**.
12. Segundo, para criar uma rede externa para acesso ao exterior, clicamos em **Novo comutador de rede virtual -> Externo -> Criar Comutador Virtual**.
13. Damos o nome de "**Externo**", selecionamos a opção de **Rede Externa** e associamos a **placa de rede Wireless** do computador físico e por fim clicamos em **OK**.

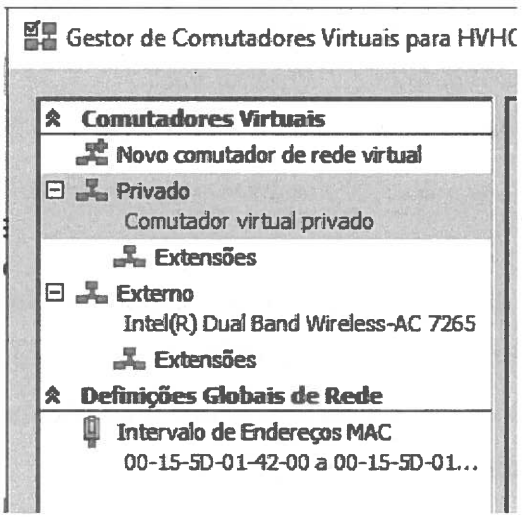

Figura 11 – *Switches virtuais* para o projeto.

14. Pronto, temos o hypervisor pronto para albergar as máquinas virtuais.
15. Mais uma nota para podermos efetuar certas operações, devemos de ir às **Definições do Hyper-v** e ativar a **Política de Modo de Sessão Avançada** no servidor e o **Modo de Sessão Avançada** no utilizador. Denotar ainda que na criação das máquinas virtuais devemos de ativar os **Serviços Convidados**.

Criação das Máquinas Virtuais

Neste conjunto de passos seguintes, que é criar as máquinas virtuais, devemos repeti-los para cobrir as máquinas todas (DC1, DC2, *Routing Server*, *SQL Server*, *NPS Server* e Cliente1 (Máquina cliente)). Eis as características a ter em conta para as mesmas:

- Armazenamento
    - Servidores: 32GB de Disco;
    - Máquina Cliente: 20GB de Disco;
- Placas de Rede
    - Adicionar também uma placa de rede ligada ao *Switch* Externo no *Routing Server*;
    - Adicionar uma placa de rede extra ligada também ao *Switch* Privado ao *NPS Server* e à Máquina Cliente. (Identificação das placas será **Privado – VPN**, enquanto que as placas de rede principais serão identificadas como **Privado - Domínio**).
- Sistema Operativo
    - Servidores: *Windows Server 2012 R2*;
    - Máquina Cliente: *Windows 8.1 Professional*;
- Ativar Serviços Convidados

1. De volta ao **Gestor do Hyper-v**, clicamos em **Ações -> Novo -> Máquina Virtual.**
2. Clicamos em **Seguinte**, damos o nome adequado à máquina mencionado anteriormente (ex. DC1), e confirmar que a máquina virtual é **guardada na mesma pasta que foi definida aquando da instalação** do *Hyper-v*.
3. Escolhemos a **Geração 2** -> **Seguinte -> 2048MB de RAM** incluindo a escolha de **Memória Dinâmica** e fazemos **Seguinte**.
4. Escolhemos o adaptador de rede **Privado**, clicamos em **Seguinte**.
5. Deixamos a opção de **Criar um disco rígido virtual** ativa, damos o mesmo nome que dêmos à máquina para a máquina em questão, e define-se o tamanho para o disco consoante a máquina. Clicamos em **Seguinte**.
6. Selecionamos a opção **Instalar um sistema operativo a partir de um ficheiro de imagem de arranque** e escolhemos o ficheiro de imagem apropriado à máquina virtual em questão. **Seguinte**.

7. E por fim fazemos **Concluir**.

   (Repetimos os passos anteriores para criação das restantes máquinas)

Configuração DC1 / DC2

Devemos repetir os seguintes passos para a configuração do 2º Controlador de Domínio (DC2), mas com as seguintes diferenças:

- Nome da Máquina: DC2
- Configuração IPv4:
  - 192.168.0.11 / 255.255.255.0 / 192.168.0.15 / 192.168.0.10 / 127.0.0.1
- Desativar IPv6.
- Ativar Serviços ICMP na firewall
- No assistente de Promoção para controlador de domínio, indicar que é para se **juntar a um domínio existente** e para funcionar em **modo de replicação.**

1. Pelo **Gestor de Hyper-V**, iniciamos a máquina DC1 e acedemos à mesma. Fazemos a instalação padrão do ***Windows Server*** mas com as características de Linguagem do sistema e teclado em **Português (Portugal)** e a palavra-passe: **P@ssw0rd** para o administrador local.
2. Após a instalação do Sistema Operativo, iniciamos sessão, e devemos de alterar as seguintes informações da Máquina:
   - Nome da Máquina: **DC1**
   - Configuração IPv4:
     - **192.168.0.10 / 255.255.255.0 / 192.168.0.15 / 192.168.0.11 / 127.0.0.1**
   - Desativar IPv6
   - Ativar Serviços ICMP na firewall
3. Vamos ao **Gestor de Servidor** e clicamos em **Gerir -> Adicionar Funções e Funcionalidades**.
4. **Seguinte** nas primeiras três páginas. Nas **Funções de servidor**, vamos selecionar os **Serviços de Domínio do *Active Directory (ADDS)*** e subsequentemente as funcionalidades aquando pedido. Clicamos em **Seguinte** até aparecer o botão **Instalar** ao qual vamos clicar.

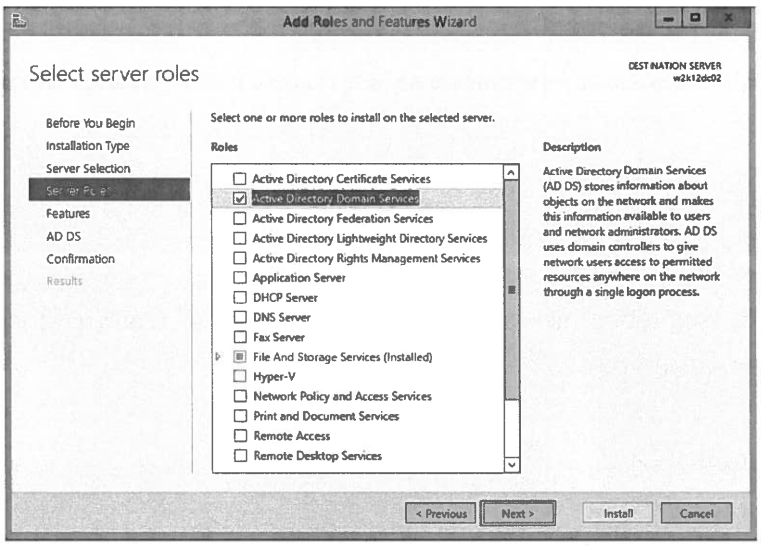

Figura 12 – Instalação do ADDS na máquina virtual DC1.

5. No fim da instalação, reiniciar se necessário. Após isto irá aparecer um símbolo de ***Warning*** no gestor de Servidor em que vamos clicar onde diz "**Promover este servidor a Controlador de Domínio**".
6. No assistente de configuração que aparecerá, escolhemos **Adicionar nova Floresta** e introduzimos o nome do domínio "**PG.local**". Clicamos em **Seguinte**.
7. Confirmamos que a Floresta irá funcionar a nível do ***Windows Server 2012 R2*** e definimos uma **Palavra-passe DSRM**: P@ssw0rd. Clicamos em **Seguinte**.
8. Após isto podemos avançar até ao fim do assistente e clicamos em **Instalar**. **Reiniciar após.**

(Na repetição dos passos para configurar o DC2, saltamos para o passo 11)

9. Depois da máquina ligar, iniciamos sessão já no Domínio (Administrador / P@ssw0rd), vamos ao **Gestor de Servidor -> Ferramentas -> Utilizadores e Computadores do *Active Directory***.
10. Expandimos o domínio, fazemos clique com o botão direito em cima do domínio e **Novo -> Utilizador**. Por este processo **criamos 2 utilizadores**, sql_admin e Cliente1, o sql_admin terá privilégios de administrador enquanto que o Cliente1 não (Palavra-passe é sempre P@ssw0rd).
11. (**Exclusivo DC2**) Depois de executar os passos anteriores, no DC2 iremos aos **Serviços e Locais do *Active Directory*** localizado nas **Ferramentas** do **Gestor de Servidor**. Nos ***Sites* -> *Default-First-Site-Name* -> *Servers* -> DC2 -> *NTDS Settings***, na parcela

direita aparecerá o objeto "**<gerado automaticamente>**", clicamos neste objeto -> **Ação -> Replicar Agora**.

Configuração Routing Server

1. No Hypervisor, pelo **Gestor de Hyper-V**, ligamos e acedemos à máquina de **Routing**, fazemos então a instalação padrão do ***Windows Server*** com a Linguagem do sistema e teclado em **Português (Portugal)** e palavra-passe: **P@ssw0rd**.
2. Após a instalação do Sistema Operativo, iniciamos sessão, e devemos de alterar as seguintes informações da Máquina:
    a. Nome da Máquina: **Routing**
    b. Configuração IPv4 (**Privado – Domínio**):
        i. **192.168.0.15 / 255.255.255.0 / - / 192.168.0.10 / 192.168.0.11**
    c. Desativar IPv6 (**Privado - Domínio**)
    d. Ativar Serviços ICMP na firewall
3. **Reiniciar** a máquina para aplicar as configurações.
4. Após o iniciar do computador, entrando com a sessão de administrador local, vamos a **Propriedades do Sistema -> Aba "Nome do Computador" -> Alterar... -> Domínio** -> escrever **"PG.local" -> OK -> Reiniciar** o computador aquando pedido.
5. Após adicionarmos o computador ao domínio, agora no início de sessão deveremos de inserir "**PG\Administrador**" como utilizador e a password padrão. Após o computador ter iniciado sessão, abrirá automaticamente o **Gestor de Servidor**, devemos então selecionar **Gerir -> Adicionar Funções e Funcionalidades -> Seguinte** até às **Funções do Servidor ->** Selecionar **Acesso Remoto -> Seguinte** até aos **Serviços de Função ->** Selecionar **Encaminhamento -> Adicionar Funcionalidades -> Seguinte** até à pagina de instalação -> **Instalar**.
6. **Reiniciar a máquina virtual**.
7. Iniciar sessão como administrador de domínio, vamos às **Ferramentas Administrativas -> Encaminhamento e Acesso Remoto**.
8. Abrirá uma nova janela, **clicamos com botão direito do rato** no nome do servidor e selecionamos **Configurar e Ativar Encaminhamento e Acesso Remoto**.
9. Um assistente de configuração irá aparecer, clicamos em **Seguinte**.
10. Selecionamos a opção de **NAT, Seguinte.**

11. Deixamos a opção **Usar esta interface pública para conexão à Internet** ativa e selecionamos a placa de rede devida **(Externo)** e clicamos em **Seguinte** e finalmente **Concluir.**

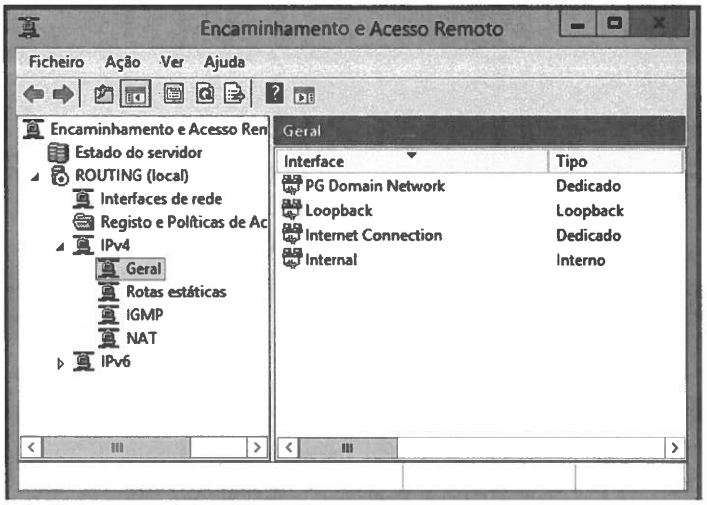

Figura 13 – Serviço de Encaminhamento implementado no Servidor *Routing*.

12. Agora a rede de domínio terá acesso à internet, com isto em conta, deveremos agora instalar todos os ***Windows Updates*** nas três máquinas virtuais até agora instaladas. **(Repetir os *Windows Updates* até não existir mais atualizações importantes)**

Configuração SQL Server

1. No *Hypervisor*, pelo **Gestor de Hyper-V**, ligamos e acedemos à máquina de **SQL Server**, fazemos então a instalação padrão do ***Windows Server*** com a Linguagem do sistema e teclado em **Português (Portugal)** e palavra-passe: **P@ssw0rd**.
2. Após a instalação do Sistema Operativo, iniciamos sessão, e devemos de alterar as seguintes informações da Máquina:
    a. Nome da Máquina: **SQL**
    b. Configuração IPv4:
        i. **192.168.0.20 / 255.255.255.0 / 192.168.0.15 / 192.168.0.10 / 192.168.0.11**
    c. Desativar IPv6
    d. Ativar Serviços ICMP na firewall
3. **Reiniciar** a máquina para aplicar as configurações.

4. Após o iniciar do computador, entrando com a sessão de administrador local, vamos a **Propriedades do Sistema -> Aba "Nome do Computador" -> Alterar... -> Domínio** -> escrever **"PG.local" -> OK -> Reiniciar** o computador aquando pedido.
5. Iniciando agora sempre a sessão com a conta **sql_admin**, deveremos instalar todos os ***Windows Updates***.

   (Repetir os *Windows Updates* até não existir mais atualizações importantes)

6. Agora vamos instalar o **SQL Server 2014**, após o computador ligar e iniciarmos sessão como **sql_admin**, vamos transferir o ficheiro de instalação do SQL para a máquina, simplesmente com um copiar e colar para dentro da máquina virtual irá resultar na sua sucedida transferência do ficheiro que podemos enviar para o seu ambiente de trabalho. Vamos então executar o **setup.exe**.
7. Na janela que aparecerá, iremos clicar em ***Installation*** que aparece de lado esquerdo e em seguida em ***New SQL Server stand-alone installation or add features to an existing installation*** no lado direito da janela.
8. Isto abrirá uma outra nova janela direcionada exclusivamente à instalação, aceitamos os **termos de licença** e clicamos em **Next**.
9. Um processo de verificação de requisitos irá começar automaticamente, dando tudo como ***Passed***, podemos avançar clicando em **Next**.
10. Para deixar que o Windows faça atualizações premimos em ***Use Microsoft Update to check for Updates*** e desseguida **Next**.
11. Agora o programa irá instalar os ficheiros de instalação, assim que completar os passos designados podemos clicar **Next**.
12. Outra verificação irá ocorrer para deteção de problemas potenciais, neste passo é normal que apareça um ***Warning*** na regra da **Firewall,** façamos **Next.**
13. Na ***Feature Selection***, podemos deixar os valores por defeito, clicamos então em **Next**.
14. Na ***Instance Configuration***, selecionamos ***Default instance*** e prosseguimos com **Next**.
15. No **Server Configuration**, atribuímos tanto para ***SQL Server Database Engine*** e ***SQL Server Browser*** a conta do **sql_admin** e a palavra-passe do costume. Clicamos em **Next**.
16. Na ***Database Engine Configuration*** mantemos todos os valores padrões e seguimos em **Next**.
17. O SQL Server 2014 será agora instalado. Aguardamos a conclusão e clicamos em **Close**.
18. Fechamos as janelas todas e **reiniciamos a máquina virtual**.

SQL Server - Criar Base de Dados

1. Na máquina virtual do *SQL Server*, vamos instalar o ***Visual Studio*** para implementação da base de dados. Da mesma forma, iremos transferir o instalador do *Visual Studio* para o ambiente de trabalho da nossa máquina virtual, e clicaremos no **ficheiro de instalação**. A instalação deverá ser por padrão, mas com as seguintes características:
    a. **Cargas de Trabalho – Desenvolvimento com a Plataforma Universal do Windows** e **Desenvolvimento para desktop com o .NET**.
2. Abrimos o ***Visual Studio***, vamos a **File -> New -> File.**
3. Na lista das **Categories** selecionar **General**, premir em **Sql File** e **Open**.
4. Agora devemos de introduzir o código seguinte que será um script para implementação da base de dados:

```
-- Cria Esquema de Base de Dados
CREATE SCHEMA [Sales]
  AUTHORIZATION [dbo];
GO

-- Cria Tabela dos Clientes
CREATE TABLE [Sales].[Customer] (
  [CustomerID]  INT         IDENTITY (1, 1) NOT NULL,
  [CustomerName] NVARCHAR (40) NOT NULL,
  [YTDOrders]   INT         NOT NULL,
  [YTDSales]    INT         NOT NULL
);
GO

-- Cria Tabela dos Pedidos
CREATE TABLE [Sales].[Orders] (
  [CustomerID] INT     NOT NULL,
  [OrderID]   INT     IDENTITY (1, 1) NOT NULL,
  [OrderDate]  DATETIME NOT NULL,
  [FilledDate] DATETIME NULL,
  [Status]    CHAR (1) NOT NULL,
  [Amount]    INT     NOT NULL
);
GO

-- Cria Tabela das Imagens
CREATE TABLE [Sales].[Images] (
  [ID]   INT     IDENTITY (1, 1) NOT NULL,
  [Image]    IMAGE  NULL,
  [Description]    NVARCHAR(50)     NULL
);
GO
```

```
-- Adiciona condições de arranque
ALTER TABLE [Sales].[Customer]
   ADD CONSTRAINT [Def_Customer_YTDOrders] DEFAULT 0 FOR [YTDOrders];
GO
ALTER TABLE [Sales].[Customer]
   ADD CONSTRAINT [Def_Customer_YTDSales] DEFAULT 0 FOR [YTDSales];
GO
ALTER TABLE [Sales].[Orders]
   ADD CONSTRAINT [Def_Orders_OrderDate] DEFAULT GetDate() FOR [OrderDate];
GO
ALTER TABLE [Sales].[Orders]
   ADD CONSTRAINT [Def_Orders_Status] DEFAULT 'Pedido Pendente' FOR [Status];
GO

-- Adiciona restrições de Chave primária e secundária
ALTER TABLE [Sales].[Customer]
   ADD CONSTRAINT [PK_Customer_CustID] PRIMARY KEY CLUSTERED
([CustomerID] ASC) WITH (ALLOW_PAGE_LOCKS = ON, ALLOW_ROW_LOCKS =
ON, PAD_INDEX = OFF, IGNORE_DUP_KEY = OFF, STATISTICS_NORECOMPUTE
= OFF);
GO
ALTER TABLE [Sales].[Orders]
   ADD CONSTRAINT [PK_Orders_OrderID] PRIMARY KEY CLUSTERED ([OrderID]
ASC) WITH (ALLOW_PAGE_LOCKS = ON, ALLOW_ROW_LOCKS = ON,
PAD_INDEX = OFF, IGNORE_DUP_KEY = OFF, STATISTICS_NORECOMPUTE =
OFF);
GO
ALTER TABLE [Sales].[Orders]
   ADD CONSTRAINT [FK_Orders_Customer_CustID] FOREIGN KEY ([CustomerID])
REFERENCES [Sales].[Customer] ([CustomerID]) ON DELETE NO ACTION ON
UPDATE NO ACTION;
GO
ALTER TABLE [Sales].[Images]
   ADD CONSTRAINT [PK_Images_ID] PRIMARY KEY CLUSTERED ([ID] ASC)
WITH (ALLOW_PAGE_LOCKS = ON, ALLOW_ROW_LOCKS = ON, PAD_INDEX =
OFF, IGNORE_DUP_KEY = OFF, STATISTICS_NORECOMPUTE = OFF);
GO


-- Adiciona restrições sobre datas inválidas
ALTER TABLE [Sales].[Orders]
   ADD CONSTRAINT [CK_Orders_FilledDate] CHECK ((FilledDate >= OrderDate)
AND (FilledDate < '01/01/2020'));
GO
ALTER TABLE [Sales].[Orders]
   ADD CONSTRAINT [CK_Orders_OrderDate] CHECK ((OrderDate > '01/01/2005') and
(OrderDate < '01/01/2020'));
GO
```

```
-- Cria um SP para cancelamento de um pedido
CREATE PROCEDURE [Sales].[uspCancelOrder]
@OrderID INT
AS
BEGIN
DECLARE @Delta INT, @CustomerID INT
BEGIN TRANSACTION
   SELECT @Delta = [Amount], @CustomerID = [CustomerID]
    FROM [Sales].[Orders] WHERE [OrderID] = @OrderID;

UPDATE [Sales].[Orders]
  SET [Status] = 'Pedido Cancelado'
WHERE [OrderID] = @OrderID;

UPDATE [Sales].[Customer]
  SET
  YTDOrders = YTDOrders - @Delta
   WHERE [CustomerID] = @CustomerID
COMMIT TRANSACTION
END
GO

-- Cria um SP para preenchimento de um pedido
CREATE PROCEDURE [Sales].[uspFillOrder]
@OrderID INT, @FilledDate DATETIME
AS
BEGIN
DECLARE @Delta INT, @CustomerID INT
BEGIN TRANSACTION
   SELECT @Delta = [Amount], @CustomerID = [CustomerID]
    FROM [Sales].[Orders] WHERE [OrderID] = @OrderID;

UPDATE [Sales].[Orders]
  SET [Status] = 'Pedido Efetuado',
     [FilledDate] = @FilledDate
WHERE [OrderID] = @OrderID;

UPDATE [Sales].[Customer]
  SET
  YTDSales = YTDSales + @Delta
   WHERE [CustomerID] = @CustomerID
COMMIT TRANSACTION
END
GO

-- Cria um SP para um novo Cliente
CREATE PROCEDURE [Sales].[uspNewCustomer]
@CustomerName NVARCHAR (40),
@CustomerID INT OUTPUT
AS
```

```
BEGIN
INSERT INTO [Sales].[Customer] (CustomerName) VALUES (@CustomerName);
SET @CustomerID = SCOPE_IDENTITY();
RETURN @@ERROR
END
GO

-- Cria um SP para novo Pedido
CREATE PROCEDURE [Sales].[uspPlaceNewOrder]
@CustomerID INT, @Amount INT, @OrderDate DATETIME, @Status CHAR
(1)='Pedido Pendente'
AS
BEGIN
DECLARE @RC INT
BEGIN TRANSACTION
INSERT INTO [Sales].[Orders] (CustomerID, OrderDate, FilledDate, Status, Amount)
    VALUES (@CustomerID, @OrderDate, NULL, @Status, @Amount)
SELECT @RC = SCOPE_IDENTITY();
UPDATE [Sales].[Customer]
  SET
  YTDOrders = YTDOrders + @Amount
   WHERE [CustomerID] = @CustomerID
COMMIT TRANSACTION
RETURN @RC
END
GO


-- Cria um SP para mostrar os pedidos na DataGrid
CREATE PROCEDURE [Sales].[uspShowOrderDetails]
@CustomerID INT=0
AS
BEGIN
SELECT [C].[CustomerName], CONVERT(date, [O].[OrderDate]), CONVERT(date,
[O].[FilledDate]), [O].[Status], [O].[Amount]
  FROM [Sales].[Customer] AS C
  INNER JOIN [Sales].[Orders] AS O
    ON [O].[CustomerID] = [C].[CustomerID]
  WHERE [C].[CustomerID] = @CustomerID
END
GO
```

5. Depois de programado o código, vamos à barra de menu e selecionamos ***File*** -> ***Save SqlQuery_1.sql As.***
6. Damos o nome de "**BDAppDadosScript.sql**" e guardamos o ficheiro no **Ambiente de Trabalho**.
7. Outra vez na barra de menu, selecionamos ***File*** -> ***Close Solution***.

8. Assim com o script criado, vamos criar um **novo projeto** para o ***Deployment* da Base de Dados**. Vamos a ***File -> New -> Project***.
9. Na nova janela escolher a categoria de ***SQL Server*** e selecionar desseguida ***SQL Server Database Project***. Vamos dar o nome "**BD.AppDados**" e clicar em **OK**.
10. Vamos novamente à barra do menu e selecionamos ***Project*** -> ***Import*** -> ***Script***. Na página das boas-vindas clicamos em **Next**.
11. Selecionamos **Single File** e associamos o **ficheiro script** que guardámos no ambiente de trabalho. Depois avançamos pelas páginas e clicamos em **Finish**.
12. Vamos agora pressionar a **tecla F5** para a implementação da base de dados ocorrer. **(ATENÇÃO: Teremos de indicar na aba DEBUG nas propriedades do projeto para a implementação ser feita no *SQL Server 2014* que nós instalámos (*Connection String*))**
13. E assim temos a nossa Base de Dados criada.

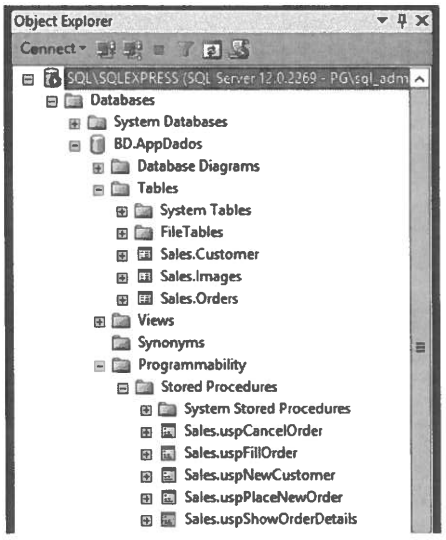

Figura 14 – Esquema da base de dados.

14. Para que a conexão à base de dados seja sucedida através da rede de domínio pelo utilizador, teremos de efetuar as seguintes instruções:
    a. Permitir as portas de ***SQL Server Database Engine*** e ***SQL Browser*** na **Firewall do Windows com Segurança Avançada**;
    b. Ativar os Protocolos **TCP/IP** e ***Named Pipes*** no ***SQL Server Configuration Manager***;

c. Criar um ***SQL Server Login*** associado à conta Windows de Domínio do **Cliente1** e atribuir as seguintes permissões ao mesmo sobre a base de dados criada:
    i. ***db_datareader*;**
    ii. ***db_datawriter*;**
    iii. ***db_owner*.**

Configuração básica NPS Server e Máquina Cliente

1. No Hypervisor, pelo **Gestor de Hyper-V**, ligamos e acedemos à máquina de **NPS Server**, fazemos então a instalação padrão do ***Windows Server*** com a Linguagem do sistema e teclado em **Português (Portugal)** e palavra-passe: **P@ssw0rd**.
2. Após a instalação do Sistema Operativo, iniciamos sessão, e devemos de alterar as seguintes informações da Máquina:
    a. Nome da Máquina: **NPS**
    b. Configuração IPv4(**Privado - Domínio**):
        i. 192.168.0.25 / 255.255.255.0 / 192.168.0.15 / 192.168.0.10 / 192.168.0.11
    c. Configuração IPv4(**Privado - VPN**):
        i. 50.50.50.1 / 255.255.255.0 / 192.168.0.15 / 192.168.0.10 / 50.50.50.1
    d. Desativar IPv6 (**Privado - Domínio** e **Privado - VPN**)
    e. Ativar Serviços ICMP na firewall
3. **Reiniciar** a máquina para aplicar as configurações.
4. Após o iniciar do computador, entrando com a sessão de administrador local, vamos a **Propriedades do Sistema -> Aba "Nome do Computador" -> Alterar... -> Domínio** -> escrever **"PG.local" -> OK -> Reiniciar** o computador aquando pedido.
5. Com a conta de **Administrador de Domínio**, deveremos agora instalar todos os ***Windows Updates***.

    (Repetir os *Windows Updates* até não existir mais atualizações importantes)

6. Após o computador iniciar, iniciamos sessão como **Administrador de Domínio**, e agora iremos providenciar o serviço de VPN para clientes exteriores poderem-se ligar ao Domínio. Mas antes, devemos de preparar a nossa máquina cliente.
7. No Hypervisor, pelo **Gestor de Hyper-V**, ligamos e acedemos à **Máquina Cliente**, fazemos então a instalação padrão do ***Windows 8.1 Professional*** com a Linguagem do sistema e teclado em **Português (Portugal)** e palavra-passe: **P@ssw0rd**.
8. Após a instalação do Sistema Operativo, iniciamos sessão, e devemos de alterar as seguintes informações da Máquina:
    a. Nome da Máquina: **PC1**
    b. Configuração IPv4(**Privado - Domínio**):
        i. 192.168.0.30 / 255.255.255.0 / 192.168.0.15 / 192.168.0.10 / 192.168.0.11
    c. Configuração IPv4(**Privado - VPN**):
        i. 50.50.50.2 / 255.255.255.0 / 50.50.50.1 / 192.168.0.10 / 50.50.50.1
    d. Desativar IPv6 (**Privado - Domínio** e **Privado - VPN**)
    e. Ativar Serviços ICMP na firewall
9. **Reiniciar** a máquina para aplicar as configurações.
10. Após o iniciar do computador, entrando com a sessão de administrador local, vamos a **Propriedades do Sistema -> Aba "Nome do Computador" -> Alterar... -> Domínio** -> escrever **"PG.local" -> OK -> Reiniciar** o computador aquando pedido.
11. Com a conta de **Administrador de Domínio**, deveremos agora instalar todos os ***Windows Updates***.

    (**Repetir os *Windows Updates* até não existir mais atualizações importantes**)

12. Depois de atualizado o Sistema Operativo, faz-se o teste de conexão (**PING**) entre a máquina cliente e o *NPS Server* através da placa de VPN que em princípio irá suceder.

Configuração Avançada NPS Server

1. Na máquina ***NPS Server***, vamos ao **Gestor de Servidor -> Gerir -> Adicionar Funções e Funcionalidades.**

2. A partir da página **Antes de Começar** clicamos em **Seguinte** até à secção de **Funções do Servidor**. Aqui selecionamos **Acesso Remoto** e desseguida clicar em **Seguinte** até à página dos **Serviços de Funções**. Iremos escolher a opção de ***DirectAcess e VPN(RAS)*** e podemos instalar finalmente indo à última página do Assistente clicando no botão **Instalar**.
3. No fim da instalação, clicamos em **Fechar**.
4. Ainda no **Gestor de Servidor**, vamos a **Ferramentas -> Encaminhamento e Acesso Remoto**.
5. Na nova janela apresentada, clicamos com o **botão direito do rato em cima do nome do Servidor** e selecionamos **Configurar e Ativar Encaminhamento e Acesso Remoto**.
6. Um Assistente de instalação irá aparecer, clicamos em **Seguinte** na primeira página, nesta próxima página selecionamos **Configuração personalizada** e desseguida em **Seguinte** novamente.
7. Escolheremos o **Acesso VPN,** clicamos em **Seguinte** e finalmente em **Concluir**.
8. Um aviso aparecerá a dizer que o serviço relacionado está pronto a ser usado, aqui clicamos em **Iniciar Serviço**.
9. Depois de aguardar algum tempo, o processo terminará e o servidor estará agora apto para albergar o sistema de VPN. Vamos às propriedades clicando com o **botão direito do rato no nome do Servidor** e selecionamos **Propriedades.**
10. Abrimos a **aba IPv4**, escolhemos a opção **Conjunto de endereços estático** e fazemos **Adicionar** colocando a seguir as seguintes informações:
    a. **Endereço IP inicial**: 192.168.0.100
    b. **Endereço IP final**: 192.168.0.110

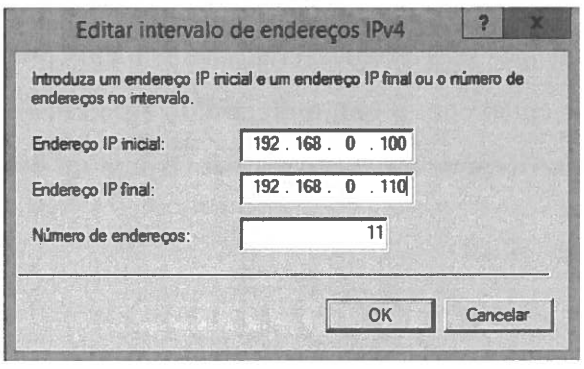

Figura 15 – Âmbito IPv4 das sessões na rede VPN.

11. Clicamos então nos dois botões **OK**, para fechar as janelas das propriedades.

12. Vamos agora de novo ao **Gestor de Servidor -> Ferramentas -> Servidor de Políticas de Rede**.
13. Isto abrirá a janela sobre o NPS, na lateral esquerda expandimos as **Políticas** e selecionamos as **Políticas de Rede**. Mas antes de avançar, vamos criar um grupo de utilizadores que irão utilizar a VPN, isto no **Controlador de Domínio**.
14. Vamos então nos ligar à máquina virtual **DC1**, abrimos o **Gestor de Servidor**, selecionamos **Ferramentas -> Utilizadores e computadores do *Active Directory***.
15. Isto abrirá a janela correspondente à ferramenta que selecionamos, clicamos então com o **botão direito do rato** em cima do **Domínio -> Novo -> Grupo**.
16. Na nova janela, daremos então o nome de "**VPN - Utilizadores**" ao grupo e fazemos **OK**.
17. Clicamos com o **botão direito do rato** em cima de **VPN - Utilizadores -> Propriedades**. Vamos à aba dos **Membros** e adicionamos o **Administrador do Domínio** e o **Cliente1** clicando em **Adicionar...**.
18. Clicamos em **OK**.
19. Voltemos então de novo à máquina virtual ***NPS Server***. Abrimos a janela do **Servidor de Políticas de Rede**, estando então selecionado as **Políticas de Rede** vamos a **Ação -> Novo**.
20. Aparecerá um assistente, aqui daremos o nome de "**Conexão-VPN**" e selecionamos **Servidor de Acesso Remoto (VPN de Acesso Telefónico)**, clicando em **Seguinte** a seguir.
21. Nesta próxima página, vamos a **Adicionar... -> Grupos de Utilizadores** e selecionamos o grupo "**VPN - Utilizadores**". Clicamos em **Seguinte**.
22. Mantemos a opção de **Acesso concedido** selecionada e fazemos **Seguinte** até à última página do assistente finalizando o processo clicando em **Concluir**.
23. Agora devemos de pôr esta política criada com maior prioridade que as outras já existentes. Vamos então com o **botão direito do rato clicar na política** e selecionar **Mover para Cima (Repetir o processo até a Ordem de Processamento ser igual a 1)**.

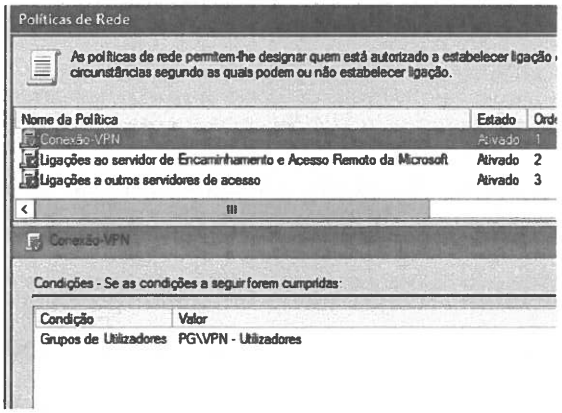

Figura 16 – Priorização de Políticas de rede.

Configuração Avançada Máquina Cliente

1. Agora vamos configurar o Cliente para a ligação VPN. Vamos então nos ligar de novo à máquina virtual **Cliente1 (Máquina Cliente)**.
2. Visto a máquina já estar configurada no Domínio, podemos remover a placa de rede que a liga ao Domínio **(Privado – Domínio)**, mantendo a placa de rede que a liga ao *NPS Server* e que vai fazer a conexão VPN. Para isto vamos ao **Hyper-V** na máquina física (*Hypervisor*), clicamos com botão direito do rato na **Máquina Cliente -> Definições....**
3. Selecionamos a **placa de rede que a liga ao domínio** mostrada na lateral esquerda e clicamos em **Remover** nos detalhes do **Adaptador de Rede**. Clicamos em **OK**.
4. Voltamos para dentro da máquina virtual **Cliente1** e na **Barra de tarefas**, clicar com o **botão direito do rato no ícone de Rede -> Abrir o Centro de Rede e Partilha**.
5. Em **Alterar as definições de rede**, clicamos em **Configurar uma nova ligação de rede.**
6. Selecionamos a opção de **Ligar a uma área de trabalho**, e a seguir em **Seguinte.**
7. Na questão do método de ligação, escolhemos **Utilizar a minha ligação à Internet**.
8. Vamos escolher **Irei configurar uma ligação à Internet mais tarde**.
9. Agora introduzimos os seguintes dados:
    a. **Endereço Internet**: 50.50.50.1;
    b. **Nome Destino**: Ligação PG VPN.
10. Clicamos em **Criar**.
11. No **Centro de Rede e Partilha**, na lateral esquerda vamos clicar em **Alterar Definições da placa**, clicamos com o **botão direito do rato em cima da ligação VPN**

-> Propriedades (Introduzir credenciais de Administrador de Domínio se necessário).

12. Selecionamos a aba **Segurança**, no **Tipo de VPN** escolhemos o **Protocolo PPTP (*Point to Point Tunneling Protocol*)**, ativamos a opção **Permitir estes protocolos** e premimos em **OK**.
13. **Configuração concluída**. Sempre que o utilizador pretender ligar-se ao domínio terá de clicar no **ícone da Rede**, selecionar a **Ligação PG VPN -> Ligar** e introduzir as **credencias do Cliente1**. (Caso não pretenda estar sempre a introduzir as credenciais para se ligar, podemos ativar a opção de **Memorizar as credenciais** na janela relacionada).

Aplicação – Introdução

A aplicação desenvolvida neste projeto visa ser uma aplicação ADO.NET que interliga a própria aplicação a uma base de dados implementada no Servidor SQL. Trata-se de uma aplicação de encomendas de Kits completos da *Lamborghini* sobre um veículo específico, que permite criar um cliente (Oficina) e o pedido associado, definir a quantidade de kits e a data de colocação da encomenda. Permite ainda o preenchimento dos mesmos pedidos (confirmação), a sua consulta, verificação do seu estado e visualização das peças integrantes do KIT.

Aplicação – Código

Antes de começar a desenvolver a aplicação em si, é necessário definir a ***Connection String***. No **Visual Studio** criamos um novo projeto **Windows Forms App** com o nome de "**AppDados**", vamos aos ***Settings.settings*** e adicionamos a seguinte entrada:

	Name	Type	Scope	Value
▶	constr	(Connectio...	Application	Data Source=SQL\SQLEXPRESS;Initial Catalog=BD.AppDados;Integrated Security=True
*				

Figura 17 – Entrada da *Connection String* nos *Settings* do Projeto.

Devemos de adicionar a referência "***System.Configuration.dll***" ao projeto através de ***Project -> Add Reference***.

Desseguida, temos de criar uma classe com o nome "**conexaoBD**" com uma função para obter a ***Connection String***:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Configuration;

namespace AppDados
{
    class conexaoBD
    {
        internal static string ObterConnectionString()
        {
            string conexao = null;
            ConnectionStringSettings settings =
ConfigurationManager.ConnectionStrings["AppDados.Properties.Settings.constr"];
            if(settings != null)
            {
                conexao = settings.ConnectionString;
            }
            return conexao;
        }
    }
}
```

Para confirmar que a ***Connection String*** está acertada, vamos ao ***App.Config*** e verificamos se se encontra tal e qual como neste próximo excerto:

```
<?xml version="1.0" encoding="utf-8" ?>
<configuration>
    <configSections>
    </configSections>
    <connectionStrings>
        <add name="AppDados.Properties.Settings.constr" connectionString="Data
Source=SQL\SQLEXPRESS;Initial Catalog=BD.AppDados;Integrated Security=True"
            providerName="System.Data.SqlClient" />
    </connectionStrings>
    <startup>
        <supportedRuntime version="v4.0" sku=".NETFramework,Version=v4.5.2" />
    </startup>
</configuration>
```

Eliminamos o Form1, criado automaticamente no início do projeto.

Começando pelo **Menu**, no **Solution Explorer** clicamos com o botão direito do rato no nome do projeto e vamos a **Add -> Windows Form...**, damos o nome de "**Menu**" e introduzimos o seguinte código:

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
```

```
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace AppDados
{
    public partial class Menu : Form
    {
        //Inicializa o form do Menu
        public Menu()
        {
            InitializeComponent();
        }

        //Evento de Botão - Abre o Form da Nova Guia
        private void btnNovaConta_Click(object sender, EventArgs e)
        {
            Form form = new NovaGuia();
            form.Show();
        }

        //Evento de Botão - Abre o Form dos Pedidos
        private void btnPedidos_Click(object sender, EventArgs e)
        {
            Form form = new Pedidos();
            form.ShowDialog();
        }


        //Evento de Botão - Fecha a Aplicação
        private void btnSair_Click(object sender, EventArgs e)
        {
            this.Close();
        }

        //Evento de Botão - Abre o Form das Imagens do Produto
        private void btnProduto_Click(object sender, EventArgs e)
        {
            Form form = new Produto();
            form.Show();
        }
    }
}
```

com o seguinte Design(**ANEXO A**):

- ***2 Labels*** e ***4 Buttons***;
- **Dimensão**: 1000x600 (Máximo e Mínimo);
- ***BackgroundImage***: Ver notas.

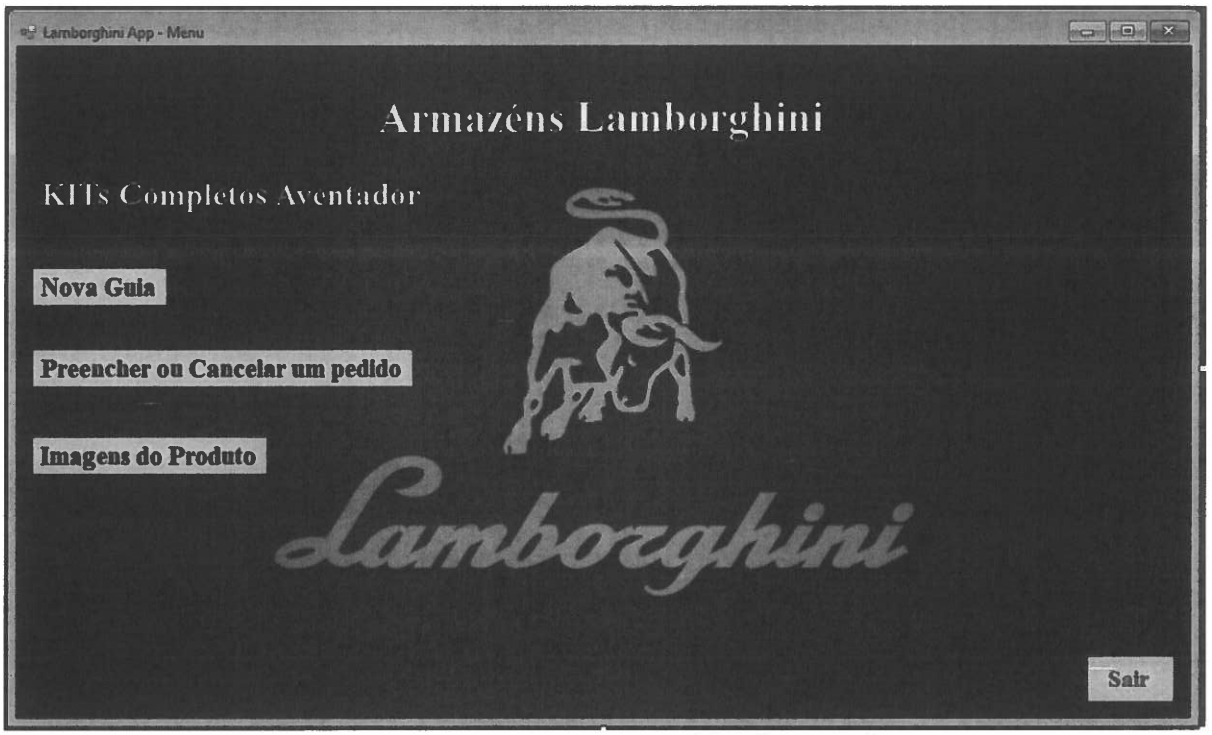

Figura 18 – Interface do menu da aplicação.

A primeira opção do menu é a opção de criar uma nova guia, portanto, da mesma forma que se criou um *Form* para o **Menu**, vamos criar agora um outro com o nome de "**NovaGuia**" com o seguinte código:

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using System.Data.SqlClient;
using System.Configuration;

namespace AppDados
{
    public partial class NovaGuia : Form
    {
        private int IDClienteConvertido;
        private int IDPedido;
        string conn = AppDados.conexaoBD.ObterConnectionString();

        //Inicializa o Form
        public NovaGuia()
        {
            InitializeComponent();
        }

        //Evento de Botão - Cria conta da Oficina
```

```
        private void btnCriarConta_Click(object sender, EventArgs e)
        {
            //Se um nome estiver inserido...
            if (nomeClienteFlag())
            {
                //Cria a conexão
                SqlConnection sqlcon = new SqlConnection(conn);
                //Cria um Comando SQL e identifica-o como Stored Procedure
                SqlCommand cmdNovaConta = new SqlCommand("Sales.uspNewCustomer",
sqlcon);
                cmdNovaConta.CommandType = CommandType.StoredProcedure;
                //Adiciona um parâmetro de entrada do Stored Procedure e especifica
o seu valor
                cmdNovaConta.Parameters.Add(new SqlParameter("@CustomerName",
SqlDbType.NVarChar, 40));
                cmdNovaConta.Parameters["@CustomerName"].Value =
txtNomeOficina.Text;
                //Adiciona um parâmetro de saída
                cmdNovaConta.Parameters.Add(new SqlParameter("@CustomerID",
SqlDbType.Int));
                cmdNovaConta.Parameters["@CustomerID"].Direction =
ParameterDirection.Output;
                try
                {
                    //Abre a Conexão
                    sqlcon.Open();
                    //Executa o SP
                    cmdNovaConta.ExecuteNonQuery();
                    //Obtém o ID do cliente gerado pela base de dados
                    this.IDClienteConvertido =
(int)cmdNovaConta.Parameters["@CustomerID"].Value;
                    this.txtIdOficina.Text = Convert.ToString(IDClienteConvertido);
                }
                catch
                {
                    MessageBox.Show("Mensagem: \n  Algo inesperado ocorreu. \n" +
                        "Nota: \n  Reinicie o programa e tente novamente.");
                }
                finally
                {
                    //Fecha a Conexão
                    sqlcon.Close();
                }
            }
        }

        //Flag para determinar se um nome foi inserido
        private bool nomeClienteFlag()
        {
            //Se a textbox estiver vazia...
            if(txtNomeOficina.Text == "")
            {
                MessageBox.Show("Mensagem: \n  Introduza um Nome!");
                return false;
            }
            else
            {
                return true;
            }
        }

        //Evento de Botão - Adiciona o pedido à base de dados
```

```
        private void btnConfirmarPedido_Click(object sender, EventArgs e)
        {
            //Se existir uma conta e uma quantidade...
            if (contaEQuantidadeFlag())
            {
                //Cria a Conexão
                SqlConnection sqlcon = new SqlConnection(conn);
                //Cria um Comando SQL e identifica-o como Stored Procedure
                SqlCommand cmdNovoPedido = new SqlCommand("Sales.uspPlaceNewOrder",
sqlcon);
                cmdNovoPedido.CommandType = CommandType.StoredProcedure;
                //Adiciona parâmetros de entrada do Stored Procedure e especifica o
seus valores
                cmdNovoPedido.Parameters.Add(new SqlParameter("@CustomerID",
SqlDbType.Int));
                cmdNovoPedido.Parameters["@CustomerID"].Value =
this.IDClienteConvertido;
                cmdNovoPedido.Parameters.Add(new SqlParameter("@OrderDate",
SqlDbType.DateTime, 8));
                cmdNovoPedido.Parameters["@OrderDate"].Value = dtpData.Value;
                cmdNovoPedido.Parameters.Add(new SqlParameter("@Amount",
SqlDbType.Int));
                cmdNovoPedido.Parameters["@Amount"].Value = numQuantidade.Value;
                cmdNovoPedido.Parameters.Add(new SqlParameter("@Status",
SqlDbType.Char, 50));
                cmdNovoPedido.Parameters["@Status"].Value = "Pedido Pendente";
                //Valor de retorno para o SP
                cmdNovoPedido.Parameters.Add(new SqlParameter("@RC",
SqlDbType.Int));
                cmdNovoPedido.Parameters["@RC"].Direction =
ParameterDirection.ReturnValue;
                try
                {
                    //Abre a Conexão
                    sqlcon.Open();
                    //Executa o SP
                    cmdNovoPedido.ExecuteNonQuery();
                    //Demonstra o ID do pedido
                    this.IDPedido = (int)cmdNovoPedido.Parameters["@RC"].Value;
                    MessageBox.Show("Mensagem: \n  O Pedido com o número " +
this.IDPedido + " foi submetido!" +
                            "\nNota: \n  (ID Pedido = " + this.IDPedido + ")");
                }
                catch
                {
                    MessageBox.Show("Mensagem: \n  O Pedido não pôde ser feito!");
                }
                finally
                {
                    //Fecha a Conexão
                    sqlcon.Close();
                }
            }
        }
        //Flag para determinar se foi criada uma conta e escolhido uma quantidade
        private bool contaEQuantidadeFlag()
        {
            //Se a textbox estiver vazia...
            if(txtIdOficina.Text == "")
            {
                MessageBox.Show("Mensagem: \n  Crie uma conta antes de confirmar
pedido!");
```

```
                return false;
            }
            //Se a quantidade for menor que 1...
            else if((numQuantidade.Value < 1))
            {
                MessageBox.Show("Mensagem: \n  Especifique a Quantidade!");
                return false;
            }
            else
            {
                return true;
            }
        }

        //Evento de Botão - Faz reset ao Form
        private void btnCriarNovaConta_Click(object sender, EventArgs e)
        {
            this.LimparForm();
        }

        //Limpa os campos do Form
        private void LimparForm()
        {
            txtNomeOficina.Clear();
            txtIdOficina.Clear();
            dtpData.Value = DateTime.Now;
            numQuantidade.Value = 0;
            this.IDClienteConvertido = 0;
        }

        //Fecha o Form da Nova Guia
        private void btnFinalizar_Click(object sender, EventArgs e)
        {
            this.Close();
        }
    }
}
```

Com o seguinte Design(**ANEXO B**):

- ***4 Labels, 2 GroupBoxes, 4 Buttons, 2 TextBoxes, 1 NumericUpDown*** e ***1 DateTimePicker***;
- **Dimensão**: 1000x600 (Máximo e Mínimo);
- ***BackgroundImage***: Ver notas.

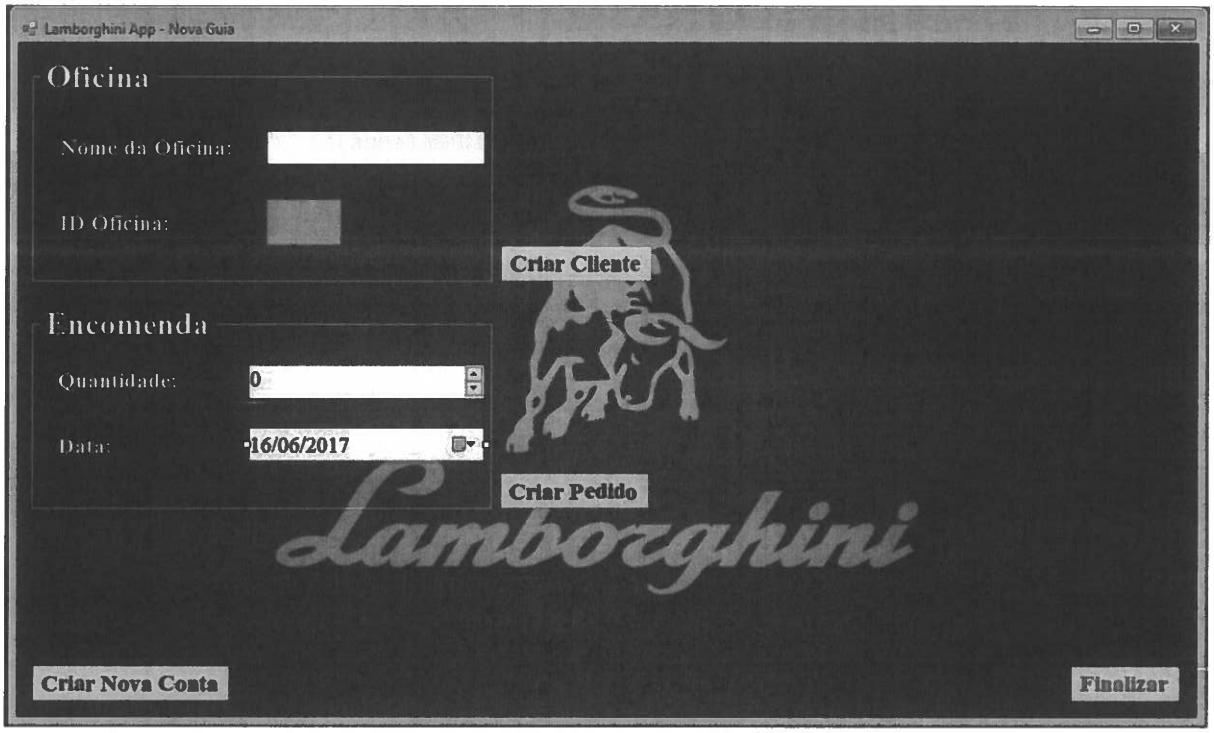

Figura 19 – Interface da nova guia na aplicação.

A próxima janela será para o Preenchimento e o Cancelar de pedidos, criamos então um novo *Form* com o nome de "**Pedidos**" com o seguinte código:

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using System.Text.RegularExpressions;
using System.Data.SqlClient;
using System.Configuration;

namespace AppDados
{
    public partial class Pedidos : Form
    {
        private int IDPedidoConvertido;
        string conn = AppDados.conexaoBD.ObterConnectionString();

        //Inicializa o Form
        public Pedidos()
        {
            InitializeComponent();
        }

        //Evento de Botão - Procura o pedido pelo ID Pedido
        private void btnProcurarPorIdPedido_Click(object sender, EventArgs e)
        {
```

```
            //Se o ID Pedido for válido...
            if (idPedidoFlag())
            {
                //Cria a Conexão
                SqlConnection sqlcon = new SqlConnection(conn);
                //Define a String de SQL Query que contém o ID Pedido como parâmetro
                string cmdsql = "select * from Sales.Orders where orderID =
@orderID";
                //Cria um comando SQL
                SqlCommand cmdIDPedido = new SqlCommand(cmdsql, sqlcon);
                //Adiciona o parâmetro ID Pedido e o seu valor
                cmdIDPedido.Parameters.Add(new SqlParameter("@orderID",
SqlDbType.Int));
                cmdIDPedido.Parameters["@orderID"].Value = IDPedidoConvertido;
                try
                {
                    //Abre a Conexão
                    sqlcon.Open();
                    //Lê os dados contidos da base de dados via Comando SQL
                    SqlDataReader sqldr = cmdIDPedido.ExecuteReader();
                    //Cria uma tabela de dados para receber os dados lidos
                    DataTable tabelaDados = new DataTable();
                    //Carrega os dados lidos para a tabela de dados
                    tabelaDados.Load(sqldr);
                    //Expõe os dados da tabela na dataGridView
                    this.dgvPedidosClientes.DataSource = tabelaDados;
                    //Fecha a Conexão
                    sqldr.Close();
                }
                catch
                {
                    MessageBox.Show("Mensagem: \n  O Pedido não pôde ser carregado
no formulário!");
                }
                finally
                {
                    //Fecha a Conexão
                    sqlcon.Close();
                }
            }
        }

        //Evento de Botão - Cancela um Pedido
        private void btnCancelarPedido_Click(object sender, EventArgs e)
        {
            //Se o ID Pedido for válido...
            if (idPedidoFlag())
            {
                //Cria a Conexão
                SqlConnection sqlcon = new SqlConnection(conn);
                // Cria um Comando SQL e identifica-o como Stored Procedure
                SqlCommand cmdCancelarPedido = new
SqlCommand("Sales.uspCancelOrder", sqlcon);
                cmdCancelarPedido.CommandType = CommandType.StoredProcedure;
                //Adiciona o parâmetro ID Pedido e o seu valor
                cmdCancelarPedido.Parameters.Add(new SqlParameter("@orderID",
SqlDbType.Int));
                cmdCancelarPedido.Parameters["@orderID"].Value = IDPedidoConvertido;
                try
                {
                    //Abre a Conexão
                    sqlcon.Open();
```

```
                    //Executa o SP
                    cmdCancelarPedido.ExecuteNonQuery();
                }
                catch
                {
                    MessageBox.Show("Mensagem: \n  O Cancelamento não foi
sucedido!");
                }
                finally
                {
                    //Fecha a Conexão
                    sqlcon.Close();
                }
            }
        }

        //Evento de Botão - Preenche (Confirma) um Pedido
        private void btnPreencherPedido_Click(object sender, EventArgs e)
        {
            //Se o ID Pedido for válido...
            if (idPedidoFlag())
            {
                //Cria a Conexão
                SqlConnection sqlcon = new SqlConnection(conn);
                // Cria um Comando SQL e identifica-o como Stored Procedure
                SqlCommand cmdPreencherPedido = new SqlCommand("Sales.uspFillOrder",
sqlcon);
                cmdPreencherPedido.CommandType = CommandType.StoredProcedure;
                //Adiciona o parâmetro ID Pedido e o seu valor
                cmdPreencherPedido.Parameters.Add(new SqlParameter("@orderID",
SqlDbType.Int));
                cmdPreencherPedido.Parameters["@orderID"].Value =
IDPedidoConvertido;
                //Adiciona o parâmetro da Data de preenchimento e o seu valor
                cmdPreencherPedido.Parameters.Add(new SqlParameter("@FilledDate",
SqlDbType.DateTime, 8));
                cmdPreencherPedido.Parameters["@FilledDate"].Value =
dtpPreencherData.Value;
                try
                {
                    //Abre a Conexão
                    sqlcon.Open();
                    //Executa o SP
                    cmdPreencherPedido.ExecuteNonQuery();
                }
                catch
                {
                    MessageBox.Show("Mensagem: \n  A confirmação do pedido não foi
sucedido!" +
                        "\nNota: \n  Verifique a Data!");
                }
                finally
                {
                    //Fecha a Conexão
                    sqlcon.Close();
                }
            }
        }

        //Flag que determina se o ID Pedido é válido
        private bool idPedidoFlag()
        {
```

```
            //Se a textbox estiver vazia
            if(txtIdPedido.Text == "")
            {
                MessageBox.Show("Mensagem: \n  Especifique o ID Pedido!");
                return false;
            }
            //Se o texto da textbox tiver caracteres sem ser números...
            else if(Regex.IsMatch(txtIdPedido.Text, @"^\D*$"))
            {
                MessageBox.Show("Mensagem \n  Introduza apenas números inteiros!");
                //Limpa o campo do ID Pedido
                txtIdPedido.Clear();
                return false;
            }
            //Senão...
            else
            {
                //Converte o ID de texto para Inteiro
                IDPedidoConvertido = Int32.Parse(txtIdPedido.Text);
                return true;
            }
        }

        //Evento de Botão - Fecha o Form
        private void btnFinalizarPedidos_Click(object sender, EventArgs e)
        {
            this.Close();
        }

        //Evento de Botão - Abre o Form dos Clientes
        private void btnIdentificarCliente_Click(object sender, EventArgs e)
        {
            Form form = new Clientes();
            form.Show();
        }
    }
}
```

Com o seguinte Design (**ANEXO C**):

- ***1 TextBox, 2 Labels, 5 Buttons, 1 DateTimePicker*** e ***1 DataGridView***;
- **Dimensão**: 1000x600 (Máximo e Mínimo);
- ***BackgroundImage***: Ver notas.

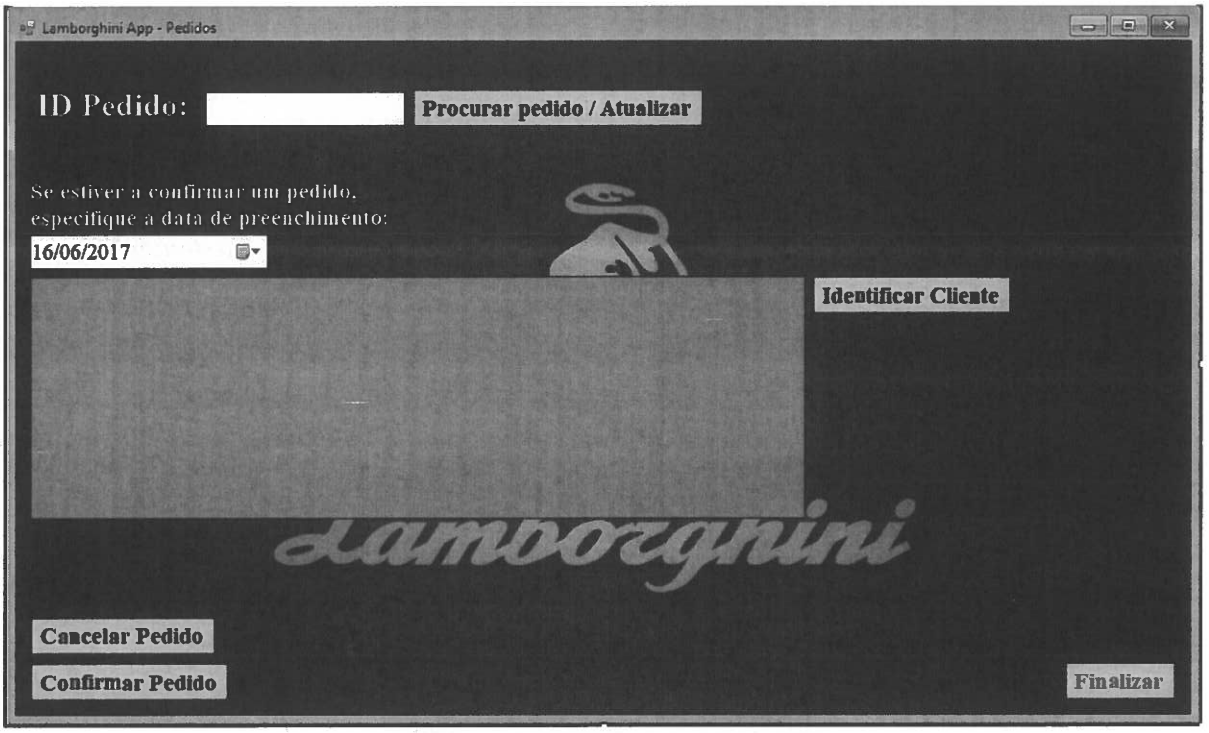

Figura 20 – Interface dos pedidos na aplicação.

Na janela dos **Pedidos**, temos o botão "**Identificar Cliente**", isto abrirá um novo *Form* que iremos criar com o nome "**Clientes**" e com o seguinte código:

```
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace AppDados
{
    public partial class Clientes : Form
    {
        //Inicializa o Form
        public Clientes()
        {
            InitializeComponent();
        }
        //Evento de Botão - Fecha o Form
        private void btnFinalizar_Click(object sender, EventArgs e)
        {
            this.Close();
        }
        //Carrega o dataGridView com o DataSource ClientesDS
        private void Clientes_Load(object sender, EventArgs e)
        {
            this.customerTableAdapter.Fill(this.clientesDS.Customer);
        }
    }
}
```

O último excerto de código (***Clientes_Load***) é criado automaticamente, este refere-se ao carregamento de informação através de um ***Data Source*** criado através do designer. Para o criar vamos ao modo de ***Design***, clicamos no ***dataGridView*** -> **Seta** no campo superior direito -> ***Choose Data Source*** -> ***Add Project Data Source*** -> ***Database (Next)*** -> ***Dataset (Next)*** -> Selecionar **constr** ***(Next)*** -> Selecionamos os campos ***CustomerID*** e ***CustomerName*** da tabela dos Clientes e fazemos ***Finish***.

E completamos o Form com o seguinte Design(**ANEXO D**):

- ***1 dataGridView*** e ***1 Button***;
- **Dimensão**: 250x600 (Máximo e Mínimo);
- ***BackgroundImage***: Ver notas.

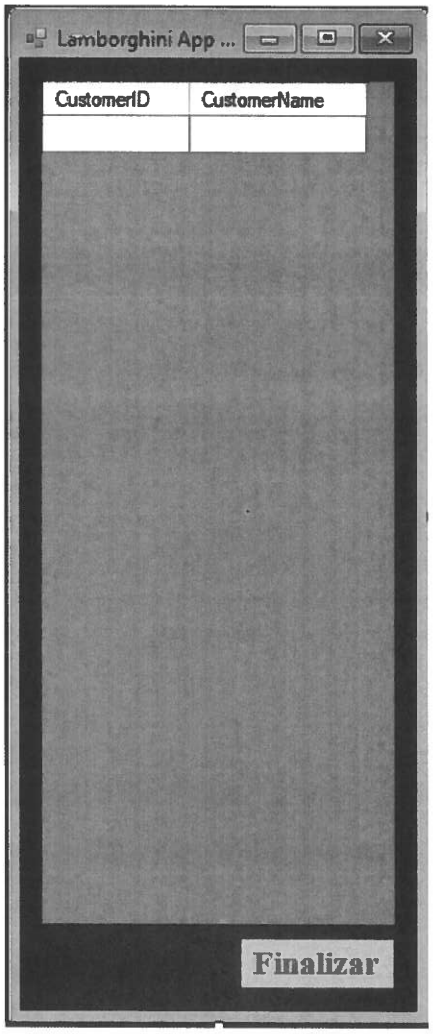

Figura 21 – Interface dos clientes na aplicação.

E finalmente, temos o último *Form* que servirá para mostrar imagens das peças incluídas no KIT. Criamos então um novo *Form* com o nome "**Produto**" e com o seguinte código:

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using System.Data.SqlClient;
using System.IO;

namespace AppDados
{
    public partial class Produto : Form
    {
        //Cria a Conexão
        SqlConnection conn = new SqlConnection(@"Data Source=SQL\SQLEXPRESS;Initial
Catalog=BD.AppDados;Integrated Security=True");
        SqlCommand sqlcmd;
        string fonteImg = "";

        //Inicializa o Form
        public Produto()
        {
            InitializeComponent();
        }

        //Evento de Botão - Faz Upload de uma Imagem
        private void btnPesquisar_Click(object sender, EventArgs e)
        {
            try
            {
                //Abre uma janela Windows para upload de uma imagem com preferência
a ficheiros JPG e GIF
                OpenFileDialog dlg = new OpenFileDialog();
                dlg.Filter = "JPG Files (*.jpg)|*.jpg|GIF Files (*.gif)|*.gif|All
Files (*.*)|*.*";
                dlg.Title = "Carregue Imagem do Produto";
                //Se a janela receber o comando do Clique em OK...
                if(dlg.ShowDialog() == DialogResult.OK)
                {
                    //Carrega o caminho da imagem
                    fonteImg = dlg.FileName.ToString();
                    pbImagem.ImageLocation = fonteImg;
                }
            }
            catch(Exception erro)
            {
                MessageBox.Show(erro.Message);
            }
        }

        //Evento de Botão - Fechar o Form
        private void btnFinalizar_Click(object sender, EventArgs e)
        {
            this.Close();
        }
```

```
        //Evento de Botão - Guarda uma entrada
        private void btnGuardar_Click(object sender, EventArgs e)
        {
            try
            {
                byte[] imagem = null;
                //Guarda a informação da imagem de forma sequencial
                FileStream fs = new FileStream(fonteImg, FileMode.Open,
FileAccess.Read);
                //Lê a informação primitiva do filestream
                BinaryReader br = new BinaryReader(fs);
                //Preenche um array de bytes com a informação da imagem
                imagem = br.ReadBytes((int)fs.Length);
                //Define umaa String de SQL Query para introdução de informação na
tabela da base de dados
                string sql = "INSERT INTO
Sales.Images(ID,Image,Description)VALUES('"+txtID.Text+"',@img,'"+txtDescricao.Text+
"')";
                //Se a conexão ainda não estiver aberta...
                if(conn.State != ConnectionState.Open)
                {
                    //Abre a Conexão
                    conn.Open();
                    //Cria um comando SQL
                    sqlcmd = new SqlCommand(sql, conn);
                    //Adiciona o parâmetro da imagem
                    sqlcmd.Parameters.Add(new SqlParameter("@img", imagem));
                    //Executa o Comando SQL
                    sqlcmd.ExecuteNonQuery();
                    //Fecha a Conexão
                    conn.Close();
                    MessageBox.Show("Mensagem: \n  Imagem guardada!");
                    //Limpeza dos campos
                    txtID.Text = "";
                    txtDescricao.Text = "";
                    pbImagem.Image = null;
                }
            }
            catch (Exception erro)
            {
                //Fecha a Conexão
                conn.Close();
                MessageBox.Show(erro.Message);
            }
        }

        //Evento de Botão - Mostra uma entrada
        private void btnMostrar_Click(object sender, EventArgs e)
        {
            try
            {
                //Define uma string de SQL Query para ir buscar informação à tabela
através do ID
                string sql = "SELECT Image, Description FROM Sales.Images WHERE
ID='"+txtID.Text+"'";
                //Se a conexão ainda não estiver aberta...
                if (conn.State != ConnectionState.Open)
                {
                    //Abre a Conexão
                    conn.Open();
                    //Cria um Comando SQL
```

```
                    sqlcmd = new SqlCommand(sql, conn);
                    //Lê a informação das linhas da tabela
                    SqlDataReader rdr = sqlcmd.ExecuteReader();
                    rdr.Read();
                    //Se a tabela tem uma ou mais linhas...
                    if (rdr.HasRows)
                    {
                        //Guarda a imagem e a descrição
                        txtDescricao.Text = rdr[1].ToString();
                        byte[] imagem = (byte[])(rdr[0]);
                        //Se não houver imagem...
                        if(imagem == null)
                        {
                            //Não mostra nada na PictureBox
                            pbImagem.Image = null;
                        }
                        else
                        {
                            //Guarda o fluxo de dados da iamgem em memória
                            MemoryStream ms = new MemoryStream(imagem);
                            //Carrega e mostra a iamgem na PictureBox
                            pbImagem.Image = Image.FromStream(ms);
                        }
                    }
                    else
                    {
                        //Limpa os campos
                        txtID.Text = "";
                        txtDescricao.Text = "";
                        pbImagem.Image = null;
                        MessageBox.Show("Mensagem: \n  O ID não existe!");
                    }
                    //Fecha a Conexão
                    conn.Close();
                }
            }
            catch(Exception erro)
            {
                conn.Close();
                MessageBox.Show(erro.Message);
            }
        }

        //Carrega o dataGridView com o DataSet ImagensDS1 na inicialização do Form
        private void Produto_Load(object sender, EventArgs e)
        {
            this.imagesTableAdapter.Fill(this.imagensDS1.Images);
        }

        //Evento de Botão - Carrega o dataGridView com o DataSet ImagensDS1
        private void btnAtualizaLista_Click(object sender, EventArgs e)
        {
            this.imagesTableAdapter.Fill(this.imagensDS1.Images);
        }

        //Evento de Botão - Elimina uma entrada
        private void btnEliminar_Click(object sender, EventArgs e)
        {
            try
            {
                //Define uma string de SQL Query para eliminar uma linha através do
ID
```

```
                string sql = "DELETE FROM Sales.Images WHERE ID = '" + txtID.Text +
"'";
                //Se a conexão ainda não estiver aberta...
                if (conn.State != ConnectionState.Open)
                {
                    //Abre a Conexão
                    conn.Open();
                    //Cria um Comando SQL
                    sqlcmd = new SqlCommand(sql, conn);
                    //Executa o Comando SQL
                    sqlcmd.ExecuteNonQuery();
                    //Fecha a Conexão
                    conn.Close();
                    MessageBox.Show("Mensagem: \n  A imagem com o ID = " +
txtID.Text + " foi eliminada!");
                    //Limpa os campos
                    txtID.Text = "";
                    txtDescricao.Text = "";
                }
            }
            catch (Exception erro)
            {
                conn.Close();
                MessageBox.Show(erro.Message);
            }
        }
    }
}
```

O excerto de código da função ***Produto_Load*** é criado automaticamente, este refere-se ao carregamento de informação através de um ***Data Source*** criado através do designer. Para o criar vamos ao modo de ***Design***, clicamos no ***dataGridView*** -> **Seta** no campo superior direito -> ***Choose Data Source*** -> ***Add Project Data Source*** -> ***Database (Next)*** -> ***Dataset (Next)*** -> Selecionar **constr** ***(Next)*** -> Selecionamos os campos ***ID*** e ***Description*** da tabela das Imagens e fazemos ***Finish***.

E completamos o *Form* com o seguinte Design(**ANEXO E**):

- ***4 Labels, 2 textBoxes, 6 Buttons, 1 pictureBox*** e ***1 dataGridView***;
- **Dimensões**: 1000x600 (Máximo e Mínimo);
- ***BackgroundImage***: Ver notas.

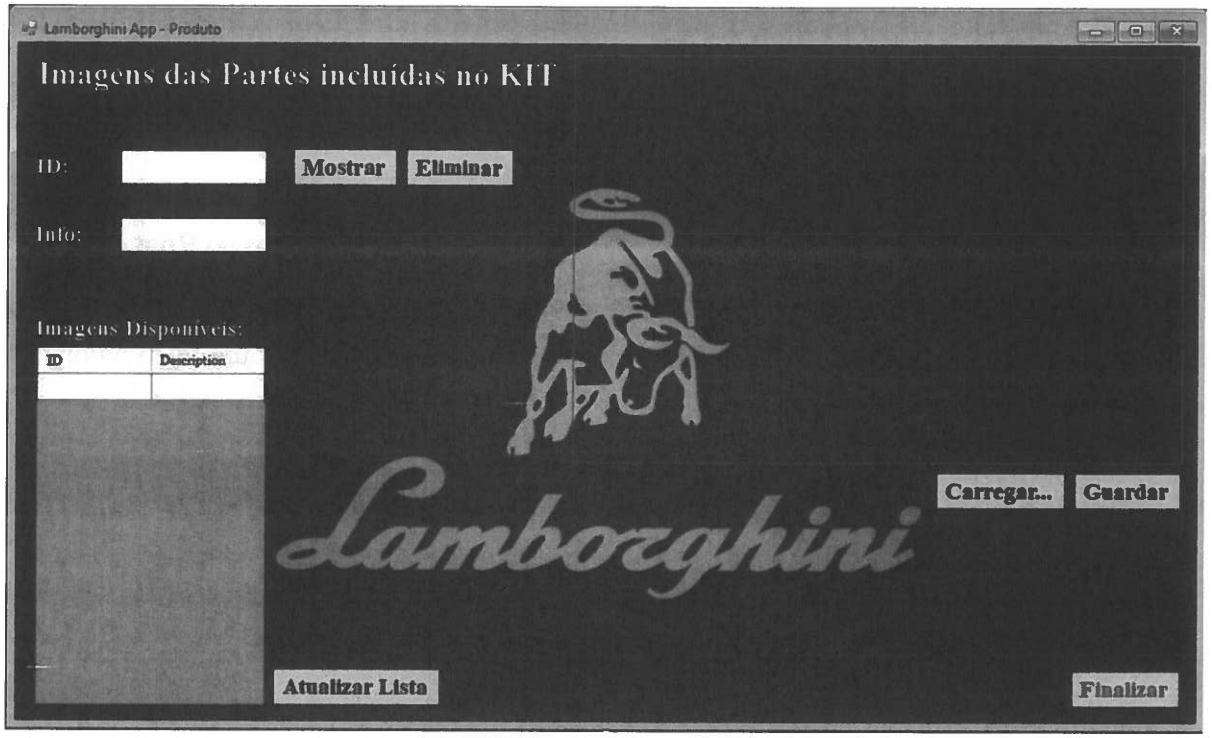

Figura 22 – Interface do produto na aplicação.

Aplicação – Notas

BackgroundImage

Para inserir uma imagem tal como a mostrada nas imagens como ***BackgroundImage*** nos forms, clicamos no **Form**, vamos às **Propriedades** do mesmo, no campo ***BackgroundImage*** clicamos nas reticências e fazemos o **Import...** da imagem. (**Imagem em ANEXO F**)

Aplicação

Devemos no fim do desenvolvimento do programa, compilar o projeto, portanto vamos à **Barra de menu -> Build -> Build AppDados**.

Aplicação – Implementação

Ainda no Visual Studio, vamos publicar a aplicação para que a máquina cliente possa aceder à mesma.

1. Vamos então à **Barra de menu -> Build -> Publish AppDados**.
2. Isto abrirá o **Publish Wizard**, na caixa de texto colocamos "**C:\AppDados**" para criar uma nova pasta no root do Disco. Clicamos em **Next**.

3. Escolhemos a opção **From a UNC path or file share** e introduzimos o mesmo caminho escrito na janela anterior (**C:\AppDados**). Clicamos em **Next**.
4. Selecionamos a opção **No, this application is only available online** e seguimos para a próxima página clicando em **Next**. E finalmente em **Finish**.
5. Fechamos o Visual Studio. Abrimos o ***Explorer*** do *Windows* e vamos ao **Root** do disco, com o **botão direito do rato** clicamos na **pasta AppDados -> Propriedades**.
6. Vamos à aba "**Partilhar**" -> **Partilha Avançada... ->** Colocamos um visto na opção de **Partilhar esta pasta** e desseguida selecionamos **Permissões**.
7. Adicionamos então o **Cliente1** com permissões **apenas de Leitura** e o **Administrador** de Domínio com **Controlo Total**.
8. Fazemos **OK** e fechamos então as janelas do explorador.
9. Mudamos agora para a máquina virtual **Cliente1** para instalação da aplicação. Iniciamos sessão na máquina com a conta de **Cliente1** e após isso **autenticar-se com a mesma conta na ligação VPN** se ainda não estiver ligado. Abrimos o explorador do *Windows*, e na aba **Computador** vamos a **Adicionar local na rede**.
10. O assistente irá aparecer, clicamos logo em **Seguinte**, selecionamos a opção de **Escolher uma localização de rede personalizada** e **Seguinte.** Na nova página introduzimos "**\\SQL\AppDados**" na caixa de texto e fazemos **Enter -> Seguinte -> Concluir**.
11. Abrimos então a pasta partilhada e executamos o ficheiro **setup.exe (é uma instalação simples, basta fazer seguinte e aceitar os termos para instalar a *.NET Framework*).**

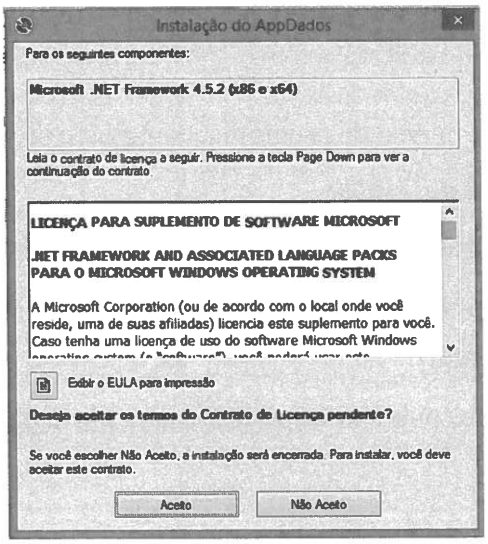

Figura 23 – Requisito de *.NET Framework* para instalação da aplicação.

12. Após a instalação, deveremos voltar à pasta partilhada e com o **botão direito do rato** em cima de **AppDados (Application Manifest)**, selecionamos **Enviar para... -> Ambiente de Trabalho (criar atalho)** para criação de um atalho à aplicação.
13. Implementação Concluída.

Conclusão

Com o laboratório finalmente implementado, só dá para ficar maravilhado ao ver até que ponto a tecnologia *Hyper-V* (Virtualização) nos permite chegar. No nosso caso, um computador devidamente bem preparado consegue colocar em execução o seu próprio Sistema Operativo mais 6 máquinas virtuais em completo funcionamento ao mesmo tempo, sendo que um servidor dedicado à virtualização com o *Windows Server 2012 R2* está preparado, teoricamente, para suportar até 1.000 Anfitriões e 25.000 Máquinas virtuais. Dá para comprovar que uma simples rede empresarial pode muito facilmente correr apenas em "cima" de uma máquina física, devidamente preparada para o efeito, visando poupar imensos custos, recursos e também bem importante, tempo. Explicando em mais pormenor o nosso caso de estudo, a infraestrutura que foi criada simula uma rede empresarial que tem como objetivo agrupar um conjunto de recursos e utilizadores formando um domínio, papel desempenhado pelos controladores de domínio (existindo 2, forma-se a funcionalidade de replicação ao qual irá aumentar a redundância de funcionamento da rede). Tendo um domínio formado, existem os servidores que pretendem fornecer serviços aos demais que na rede habitam, temos o Servidor de *Routing* que têm 2 placas de rede incorporadas para estar ligado ao domínio e ao exterior ao mesmo tempo, de forma a fornecer acesso à internet, temos o servidor NPS que irá adicionar uma camada de segurança entre o domínio e tudo o que está à sua volta, criando um túnel (VPN) para a quem for devido poder aceder através de um método de autenticação. À volta da aplicação temos a máquina cliente que acede ao domínio pelo anteriormente mencionado túnel (VPN) que irá permitir aceder à aplicação e interagir com a mesma, simbolizando uma interação cliente-servidor totalmente virtual sem sequer sair da rede do domínio, e por fim temos o Servidor SQL, obtendo o *Visual Studio*, neste é possível criar uma aplicação *ADO.NET* e testar constantemente a ligação à base de dados no seu próprio sistema, ao fim da criação de tal software, basta publicá-la para o cliente poder usá-lo. Finalizando, podemos tirar a conclusão de que a virtualização é um grande pilar para o futuro das Tecnologias de Informações, vendo já desde cedo o enorme suporte que dá ao conceito do *Cloud Computing*, observando a sua evolução sobre si próprio e crescendo até ao nível de ... *Fog Computing*.

Referências

Bittman, T., Dawson, P., & Warrilow, M. (2016). Magic Quadrant for x86 Server Virtualization Infrastructure. Retrieved January 21, 2017, from https://www.gartner.com/doc/reprints?ct=160707&id=1-3B9FAM0&st=sb

Buyya, R., Broberg, J., & Goscinski, A. (2011). *Cloud Computing: Principles and Paradigms. Cloud Computing: Principles and Paradigms*. https://doi.org/10.1002/9780470940105

DANRESA. (2017). O ADO.NET Entity Framework no .NET 4. Retrieved January 19, 2017, from http://www.danresa.com.br/fabrica-de-software/index.php/o-ado-net-entity-framework-no-net-4/

Fallis, A. . (2013). *Microsoft ADO.NET Entity Framework. Journal of Chemical Information and Modeling* (Vol. 53). https://doi.org/10.1017/CBO9781107415324.004

Ferreira, A. (2015). *Introdução ao Cloud Computing*. (FCA, Ed.). Lisboa: FCA.

ITU-T. (2014). ITU-T. *SERIES Y: GLOBAL INFORMATION INFRASTRUCTURE, INTERNET PROTOCOL ASPECTS AND NEXT-GENERATION NETWORKS*, (Y.3500).

Johnson, E. B. (2015). Microsoft System Center Virtual Machine Manager 2012 R2 storage automation with IBM XIV Storage System Gen3, (February). Retrieved from http://www-03.ibm.com/support/techdocs/atsmastr.nsf/5cb5ed706d254a8186256c71006d2e0a/c58838106cfb921186257de6001ae749/$FILE/Microsoft SCVMM 2012 R2 XIV Gen3.pdf

Mell, P., & Grance, T. (2011). The NIST definition of cloud computing. *NIST Special Publication*, *145*, 7. https://doi.org/10.1136/emj.2010.096966

Microsoft. (2016). Windows Server 2012 R2 Server Virtualization White Paper, 1–116. Retrieved from http://download.microsoft.com/download/A/2/7/A27F60C3-5113-494A-9215-D02A8ABCFD6B/Windows_Server_2012_R2_Server_Virtualization_White_Paper.pdf

Microsoft Press. Windows Server 2012 R2 Overview White Paper (2014). Retrieved from https://download.microsoft.com/download/0/2/1/021BE527-A882-41E6-A83B-8072BF58721E/Windows_Server_2012_R2_Overview_White_Paper.pdf

Microsoft Technet. (2014). What's New in Active Directory in Windows Server. Retrieved

January 16, 2017, from https://technet.microsoft.com/en-us/library/dn268294.aspx

Minasi, M., Greene, K., Booth, C., Butler, R., McCabe, J., Panek, R., … Roth, S. (2014). *Mastering Windows 2012 R2*. John Wiley & Sons, Inc. Retrieved from https://books.google.pt/books?id=b4AKWdifLYYC&lpg=PA31&ots=DJ8adgh0Sn&dq=mastering windows server 2012&lr&hl=pt-PT&pg=PA1#v=onepage&q&f=false

Mistry, R., & Misner, S. (2014). *Introducing Microsoft SQL Server 2014 Technical Overview*. (D. Musgrave, C. Dillingham, & J. Pierce, Eds.). Redmond, Washington. Retrieved from http://download.microsoft.com/download/D/F/2/DF25A191-1FA4-4BC2-925C-492D616CF7FA/Microsoft_Press_ebook_Introducing_Microsoft_SQL_Server_2014_PDF.pdf

NVSG. (2012). Advanced Network Virtualization : Definition , Benefits , Applications , and Technical Challenges, (January), 1–35. Retrieved from https://nvlab.nakao-lab.org/nv-study-group-white-paper.v1.0.pdf

Paessler, D. (2013). Virtualização de Servidor e Gerenciamento de Rede, 8. Retrieved from https://assets.paessler.com/common/files/pdf/whitepaper/server_virtualization_br.pdf

Patrick, T. (2010). *Microsoft ADO.NET 4 Step by Step*. (R. Jones & K. Borg, Eds.). California: O'Riley Media, Inc. Retrieved from http://pdf.th7.cn/down/files/1411/Microsoft ADO.NET 4 Step by Step.pdf

Salem, K. (2008). State of the Art and Research Challenges What is Virtualization ? Example : Virtual Machines Example : Virtual Storage, 1–45. Retrieved from https://ce2fe2e5-a-62cb3a1a-s-sites.googlegroups.com/site/cloudcomputingsystem/research/data-intensive-computing/DBVirtTutorial_EDBT2008.pdf?attachauth=ANoY7crKerZVSvfEMPZuakKh4J5BuuU-MoSPaNuY55IRVp4ffRpRZznMgoO0DkjM3iTXII2adxsc9SKPnu_NzfPmw-StBRmknlvMNdz

Schnackenburg, P. (2011). Systems Center Virtual Machine Manager 2012 review - Part 1: What's new, installation - 4sysops. Retrieved January 17, 2017, from https://4sysops.com/archives/systems-center-virtual-machine-manager-2012-review-part-1-whats-new-installation/

Spilker, K. (2014). From the MVPs: SQL Server 2014 Performance Enhancements – Microsoft

Press blog. Retrieved January 18, 2017, from https://blogs.msdn.microsoft.com/microsoft_press/2014/05/19/from-the-mvps-sql-server-2014-performance-enhancements/

Tulloch, M., & Team, W. S. (2013). *Introducing Windows Server 2012 R2* (Vol. 15). Retrieved from https://books.google.com/books?id=IlxuAwAAQBAJ&pgis=1

Ziembicki, D., & Tulloch, M. (2014). *Microsoft System Center integrated Cloud Platform*. (A. Hamilton, K. Szall, & A. Jones, Eds.). Redmond, Washington: S4Carlisle Publishing Services. Retrieved from https://download.microsoft.com/DOWNLOAD/3/B/2/3B27DCBA-A35C-4A0B-87A7-98B956AE98BB/MICROSOFT_SYSTEM_CENTER_INTEGRATED_CLOUD_PLATFORM_PDF.PDF

ANEXOS

ANEXO A

```
namespace AppDados
{
    partial class Menu
    {
        /// <summary>
        /// Required designer variable.
        /// </summary>
        private System.ComponentModel.IContainer components = null;

        /// <summary>
        /// Clean up any resources being used.
        /// </summary>
        /// <param name="disposing">true if managed resources should be disposed;
otherwise, false.</param>
        protected override void Dispose(bool disposing)
        {
            if (disposing && (components != null))
            {
                components.Dispose();
            }
            base.Dispose(disposing);
        }

        #region Windows Form Designer generated code

        /// <summary>
        /// Required method for Designer support - do not modify
        /// the contents of this method with the code editor.
        /// </summary>
        private void InitializeComponent()
        {
            this.lblbemvindo = new System.Windows.Forms.Label();
            this.btnNovaGuia = new System.Windows.Forms.Button();
            this.btnPedidos = new System.Windows.Forms.Button();
            this.btnSair = new System.Windows.Forms.Button();
            this.btnProduto = new System.Windows.Forms.Button();
            this.label1 = new System.Windows.Forms.Label();
            this.SuspendLayout();
            //
            // lblbemvindo
            //
            this.lblbemvindo.BackColor = System.Drawing.Color.Transparent;
            this.lblbemvindo.Font = new System.Drawing.Font("Times New Roman",
27.75F, System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point,
((byte)(0)));
            this.lblbemvindo.ForeColor = System.Drawing.Color.White;
            this.lblbemvindo.Location = new System.Drawing.Point(12, 38);
            this.lblbemvindo.Name = "lblbemvindo";
            this.lblbemvindo.Size = new System.Drawing.Size(960, 48);
            this.lblbemvindo.TabIndex = 0;
            this.lblbemvindo.Text = "Armazéns Lamborghini\r\n\r\n´\r\n";
            this.lblbemvindo.TextAlign =
System.Drawing.ContentAlignment.MiddleCenter;
            //
            // btnNovaGuia
```

```
            //
            this.btnNovaGuia.AccessibleName = "";
            this.btnNovaGuia.AutoSize = true;
            this.btnNovaGuia.AutoSizeMode =
System.Windows.Forms.AutoSizeMode.GrowAndShrink;
            this.btnNovaGuia.BackColor = System.Drawing.Color.Transparent;
            this.btnNovaGuia.Font = new System.Drawing.Font("Times New Roman",
15.75F, System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point,
((byte)(0)));
            this.btnNovaGuia.ForeColor = System.Drawing.Color.Black;
            this.btnNovaGuia.Location = new System.Drawing.Point(12, 184);
            this.btnNovaGuia.Name = "btnNovaGuia";
            this.btnNovaGuia.Size = new System.Drawing.Size(116, 34);
            this.btnNovaGuia.TabIndex = 1;
            this.btnNovaGuia.Text = "Nova Guia";
            this.btnNovaGuia.UseVisualStyleBackColor = false;
            this.btnNovaGuia.Click += new
System.EventHandler(this.btnNovaConta_Click);
            //
            // btnPedidos
            //
            this.btnPedidos.AutoSize = true;
            this.btnPedidos.AutoSizeMode =
System.Windows.Forms.AutoSizeMode.GrowAndShrink;
            this.btnPedidos.BackColor = System.Drawing.Color.Transparent;
            this.btnPedidos.Font = new System.Drawing.Font("Times New Roman",
15.75F, System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point,
((byte)(0)));
            this.btnPedidos.ForeColor = System.Drawing.Color.Black;
            this.btnPedidos.Location = new System.Drawing.Point(12, 252);
            this.btnPedidos.Name = "btnPedidos";
            this.btnPedidos.Size = new System.Drawing.Size(322, 34);
            this.btnPedidos.TabIndex = 2;
            this.btnPedidos.Text = "Preencher ou Cancelar um pedido";
            this.btnPedidos.UseVisualStyleBackColor = false;
            this.btnPedidos.Click += new System.EventHandler(this.btnPedidos_Click);
            //
            // btnSair
            //
            this.btnSair.AutoSize = true;
            this.btnSair.BackColor = System.Drawing.Color.Transparent;
            this.btnSair.Font = new System.Drawing.Font("Times New Roman", 15F,
System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point, ((byte)(0)));
            this.btnSair.ForeColor = System.Drawing.Color.Red;
            this.btnSair.Location = new System.Drawing.Point(897, 508);
            this.btnSair.Name = "btnSair";
            this.btnSair.Size = new System.Drawing.Size(75, 41);
            this.btnSair.TabIndex = 3;
            this.btnSair.Text = "Sair";
            this.btnSair.UseVisualStyleBackColor = false;
            this.btnSair.Click += new System.EventHandler(this.btnSair_Click);
            //
            // btnProduto
            //
            this.btnProduto.AutoSize = true;
            this.btnProduto.AutoSizeMode =
System.Windows.Forms.AutoSizeMode.GrowAndShrink;
            this.btnProduto.BackColor = System.Drawing.Color.Transparent;
            this.btnProduto.Font = new System.Drawing.Font("Times New Roman",
15.75F, System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point,
((byte)(0)));
            this.btnProduto.ForeColor = System.Drawing.Color.Black;
```

```
            this.btnProduto.Location = new System.Drawing.Point(12, 325);
            this.btnProduto.Name = "btnProduto";
            this.btnProduto.Size = new System.Drawing.Size(200, 34);
            this.btnProduto.TabIndex = 4;
            this.btnProduto.Text = "Imagens do Produto";
            this.btnProduto.UseVisualStyleBackColor = false;
            this.btnProduto.Click += new System.EventHandler(this.btnProduto_Click);
            //
            // label1
            //
            this.label1.BackColor = System.Drawing.Color.Transparent;
            this.label1.Font = new System.Drawing.Font("Times New Roman", 20.25F,
System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point, ((byte)(0)));
            this.label1.ForeColor = System.Drawing.Color.White;
            this.label1.Location = new System.Drawing.Point(12, 101);
            this.label1.Name = "label1";
            this.label1.Size = new System.Drawing.Size(338, 42);
            this.label1.TabIndex = 5;
            this.label1.Text = "KITs Completos Aventador";
            this.label1.TextAlign = System.Drawing.ContentAlignment.MiddleCenter;
            //
            // Menu
            //
            this.AutoScaleDimensions = new System.Drawing.SizeF(6F, 14F);
            this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font;
            this.BackgroundImage = global::AppDados.Properties.Resources.bg2;
            this.BackgroundImageLayout = System.Windows.Forms.ImageLayout.Stretch;
            this.ClientSize = new System.Drawing.Size(984, 561);
            this.Controls.Add(this.label1);
            this.Controls.Add(this.btnProduto);
            this.Controls.Add(this.btnSair);
            this.Controls.Add(this.btnPedidos);
            this.Controls.Add(this.btnNovaGuia);
            this.Controls.Add(this.lblbemvindo);
            this.Font = new System.Drawing.Font("Times New Roman", 8.25F,
System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point, ((byte)(0)));
            this.MaximumSize = new System.Drawing.Size(1000, 600);
            this.MinimumSize = new System.Drawing.Size(1000, 600);
            this.Name = "Menu";
            this.Text = "Lamborghini App - Menu";
            this.ResumeLayout(false);
            this.PerformLayout();

        }

        #endregion

        private System.Windows.Forms.Label lblbemvindo;
        private System.Windows.Forms.Button btnNovaGuia;
        private System.Windows.Forms.Button btnPedidos;
        private System.Windows.Forms.Button btnSair;
        private System.Windows.Forms.Button btnProduto;
        private System.Windows.Forms.Label label1;
    }
}
```

ANEXO B

```
namespace AppDados
{
   partial class NovaGuia
   {
      /// <summary>
      /// Required designer variable.
      /// </summary>
      private System.ComponentModel.IContainer components = null;

      /// <summary>
      /// Clean up any resources being used.
      /// </summary>
      /// <param name="disposing">true if managed resources should be disposed; otherwise,
false.</param>
      protected override void Dispose(bool disposing)
      {
         if (disposing && (components != null))
         {
            components.Dispose();
         }
         base.Dispose(disposing);
      }

      #region Windows Form Designer generated code

      /// <summary>
      /// Required method for Designer support - do not modify
      /// the contents of this method with the code editor.
      /// </summary>
      private void InitializeComponent()
      {
         this.gbOficina = new System.Windows.Forms.GroupBox();
         this.txtIdOficina = new System.Windows.Forms.TextBox();
         this.txtNomeOficina = new System.Windows.Forms.TextBox();
         this.lblIdCliente = new System.Windows.Forms.Label();
         this.lblNomeOficina = new System.Windows.Forms.Label();
         this.btnCriarCliente = new System.Windows.Forms.Button();
         this.textBox3 = new System.Windows.Forms.TextBox();
         this.textBox4 = new System.Windows.Forms.TextBox();
         this.label1 = new System.Windows.Forms.Label();
         this.label2 = new System.Windows.Forms.Label();
         this.gbEncomenda = new System.Windows.Forms.GroupBox();
         this.dtpData = new System.Windows.Forms.DateTimePicker();
         this.numQuantidade = new System.Windows.Forms.NumericUpDown();
         this.lblData = new System.Windows.Forms.Label();
         this.lblQuantidade = new System.Windows.Forms.Label();
         this.btnCriarPedido = new System.Windows.Forms.Button();
         this.btnFechar = new System.Windows.Forms.Button();
```

```
        this.btnCriarNovaConta = new System.Windows.Forms.Button();
        this.gbOficina.SuspendLayout();
        this.gbEncomenda.SuspendLayout();
        ((System.ComponentModel.ISupportInitialize)(this.numQuantidade)).BeginInit();
        this.SuspendLayout();
        //
        // gbOficina
        //
        this.gbOficina.BackColor = System.Drawing.Color.Transparent;
        this.gbOficina.Controls.Add(this.txtIdOficina);
        this.gbOficina.Controls.Add(this.txtNomeOficina);
        this.gbOficina.Controls.Add(this.lblIdCliente);
        this.gbOficina.Controls.Add(this.lblNomeOficina);
        this.gbOficina.Font = new System.Drawing.Font("Times New Roman", 20.25F,
System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point, ((byte)(0)));
        this.gbOficina.ForeColor = System.Drawing.Color.White;
        this.gbOficina.Location = new System.Drawing.Point(12, 12);
        this.gbOficina.Name = "gbOficina";
        this.gbOficina.Size = new System.Drawing.Size(385, 188);
        this.gbOficina.TabIndex = 0;
        this.gbOficina.TabStop = false;
        this.gbOficina.Text = "Oficina";
        //
        // txtIdOficina
        //
        this.txtIdOficina.BackColor = System.Drawing.SystemColors.ActiveBorder;
        this.txtIdOficina.BorderStyle = System.Windows.Forms.BorderStyle.FixedSingle;
        this.txtIdOficina.Font = new System.Drawing.Font("Times New Roman", 20.25F,
System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point, ((byte)(0)));
        this.txtIdOficina.Location = new System.Drawing.Point(196, 118);
        this.txtIdOficina.Name = "txtIdOficina";
        this.txtIdOficina.ReadOnly = true;
        this.txtIdOficina.Size = new System.Drawing.Size(63, 39);
        this.txtIdOficina.TabIndex = 3;
        this.txtIdOficina.TextAlign                                              =
System.Windows.Forms.HorizontalAlignment.Center;
        //
        // txtNomeOficina
        //
        this.txtNomeOficina.Font = new System.Drawing.Font("Times New Roman",
14.25F,      System.Drawing.FontStyle.Regular,      System.Drawing.GraphicsUnit.Point,
((byte)(0)));
        this.txtNomeOficina.Location = new System.Drawing.Point(196, 61);
        this.txtNomeOficina.Name = "txtNomeOficina";
        this.txtNomeOficina.Size = new System.Drawing.Size(183, 29);
        this.txtNomeOficina.TabIndex = 2;
        //
        // lblIdCliente
        //
        this.lblIdCliente.AutoSize = true;
```

```
        this.lblIdCliente.Font = new System.Drawing.Font("Times New Roman", 14.25F,
System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point, ((byte)(0)));
        this.lblIdCliente.Location = new System.Drawing.Point(20, 127);
        this.lblIdCliente.Name = "lblIdCliente";
        this.lblIdCliente.Size = new System.Drawing.Size(101, 22);
        this.lblIdCliente.TabIndex = 1;
        this.lblIdCliente.Text = "ID Oficina:";
        //
        // lblNomeOficina
        //
        this.lblNomeOficina.AutoSize = true;
        this.lblNomeOficina.Font = new System.Drawing.Font("Times New Roman",
14.25F, System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point,
((byte)(0)));
        this.lblNomeOficina.Location = new System.Drawing.Point(20, 64);
        this.lblNomeOficina.Name = "lblNomeOficina";
        this.lblNomeOficina.Size = new System.Drawing.Size(158, 22);
        this.lblNomeOficina.TabIndex = 0;
        this.lblNomeOficina.Text = "Nome da Oficina: ";
        //
        // btnCriarCliente
        //
        this.btnCriarCliente.AutoSize = true;
        this.btnCriarCliente.AutoSizeMode =
System.Windows.Forms.AutoSizeMode.GrowAndShrink;
        this.btnCriarCliente.Font = new System.Drawing.Font("Times New Roman",
14.25F, System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point,
((byte)(0)));
        this.btnCriarCliente.Location = new System.Drawing.Point(403, 168);
        this.btnCriarCliente.Name = "btnCriarCliente";
        this.btnCriarCliente.Size = new System.Drawing.Size(128, 32);
        this.btnCriarCliente.TabIndex = 1;
        this.btnCriarCliente.Text = "Criar Cliente";
        this.btnCriarCliente.UseVisualStyleBackColor = true;
        this.btnCriarCliente.Click += new System.EventHandler(this.btnCriarConta_Click);
        //
        // textBox3
        //
        this.textBox3.Location = new System.Drawing.Point(182, 116);
        this.textBox3.Name = "textBox3";
        this.textBox3.Size = new System.Drawing.Size(197, 20);
        this.textBox3.TabIndex = 3;
        //
        // textBox4
        //
        this.textBox4.Location = new System.Drawing.Point(182, 53);
        this.textBox4.Name = "textBox4";
        this.textBox4.Size = new System.Drawing.Size(197, 20);
        this.textBox4.TabIndex = 2;
        //
```

```
        // label1
        //
        this.label1.AutoSize = true;
        this.label1.Font = new System.Drawing.Font("Times New Roman", 14.25F,
System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point, ((byte)(0)));
        this.label1.Location = new System.Drawing.Point(20, 127);
        this.label1.Name = "label1";
        this.label1.Size = new System.Drawing.Size(99, 22);
        this.label1.TabIndex = 1;
        this.label1.Text = "ID Cliente:";
        //
        // label2
        //
        this.label2.AutoSize = true;
        this.label2.Font = new System.Drawing.Font("Times New Roman", 14.25F,
System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point, ((byte)(0)));
        this.label2.Location = new System.Drawing.Point(20, 64);
        this.label2.Name = "label2";
        this.label2.Size = new System.Drawing.Size(156, 22);
        this.label2.TabIndex = 0;
        this.label2.Text = "Nome do Cliente: ";
        //
        // gbEncomenda
        //
        this.gbEncomenda.BackColor = System.Drawing.Color.Transparent;
        this.gbEncomenda.Controls.Add(this.dtpData);
        this.gbEncomenda.Controls.Add(this.numQuantidade);
        this.gbEncomenda.Controls.Add(this.lblData);
        this.gbEncomenda.Controls.Add(this.lblQuantidade);
        this.gbEncomenda.Font = new System.Drawing.Font("Times New Roman", 20.25F,
System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point, ((byte)(0)));
        this.gbEncomenda.ForeColor = System.Drawing.Color.White;
        this.gbEncomenda.Location = new System.Drawing.Point(12, 218);
        this.gbEncomenda.Name = "gbEncomenda";
        this.gbEncomenda.Size = new System.Drawing.Size(385, 171);
        this.gbEncomenda.TabIndex = 4;
        this.gbEncomenda.TabStop = false;
        this.gbEncomenda.Text = "Encomenda";
        //
        // dtpData
        //
        this.dtpData.AllowDrop = true;
        this.dtpData.Font = new System.Drawing.Font("Times New Roman", 14.25F,
System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point, ((byte)(0)));
        this.dtpData.Format = System.Windows.Forms.DateTimePickerFormat.Short;
        this.dtpData.Location = new System.Drawing.Point(182, 102);
        this.dtpData.MinDate = System.DateTime.Today;
        this.dtpData.Name = "dtpData";
        this.dtpData.Size = new System.Drawing.Size(197, 29);
        this.dtpData.TabIndex = 3;
```

```
        this.dtpData.Value = System.DateTime.Today;
        //
        // numQuantidade
        //
        this.numQuantidade.Font = new System.Drawing.Font("Times New Roman",
14.25F, System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point,
((byte)(0)));
        this.numQuantidade.Location = new System.Drawing.Point(182, 50);
        this.numQuantidade.Maximum = new decimal(new int[] {
        5000,
        0,
        0,
        0});
        this.numQuantidade.Name = "numQuantidade";
        this.numQuantidade.Size = new System.Drawing.Size(197, 29);
        this.numQuantidade.TabIndex = 2;
        //
        // lblData
        //
        this.lblData.AutoSize = true;
        this.lblData.Font = new System.Drawing.Font("Times New Roman", 14.25F,
System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point, ((byte)(0)));
        this.lblData.Location = new System.Drawing.Point(18, 107);
        this.lblData.Name = "lblData";
        this.lblData.Size = new System.Drawing.Size(55, 22);
        this.lblData.TabIndex = 1;
        this.lblData.Text = "Data:";
        //
        // lblQuantidade
        //
        this.lblQuantidade.AutoSize = true;
        this.lblQuantidade.Font = new System.Drawing.Font("Times New Roman", 14.25F,
System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point, ((byte)(0)));
        this.lblQuantidade.Location = new System.Drawing.Point(18, 52);
        this.lblQuantidade.Name = "lblQuantidade";
        this.lblQuantidade.Size = new System.Drawing.Size(110, 22);
        this.lblQuantidade.TabIndex = 0;
        this.lblQuantidade.Text = "Quantidade:";
        //
        // btnCriarPedido
        //
        this.btnCriarPedido.AutoSize = true;
        this.btnCriarPedido.AutoSizeMode =
System.Windows.Forms.AutoSizeMode.GrowAndShrink;
        this.btnCriarPedido.Font = new System.Drawing.Font("Times New Roman",
14.25F, System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point,
((byte)(0)));
        this.btnCriarPedido.Location = new System.Drawing.Point(403, 357);
        this.btnCriarPedido.Name = "btnCriarPedido";
        this.btnCriarPedido.Size = new System.Drawing.Size(126, 32);
```

```
        this.btnCriarPedido.TabIndex = 5;
        this.btnCriarPedido.Text = "Criar Pedido";
        this.btnCriarPedido.UseVisualStyleBackColor = true;
        this.btnCriarPedido.Click                    +=                    new
System.EventHandler(this.btnConfirmarPedido_Click);
        //
        // btnFechar
        //
        this.btnFechar.AutoSize = true;
        this.btnFechar.AutoSizeMode                                        =
System.Windows.Forms.AutoSizeMode.GrowAndShrink;
        this.btnFechar.Font = new System.Drawing.Font("Times New Roman", 14.25F,
System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point, ((byte)(0)));
        this.btnFechar.ForeColor = System.Drawing.Color.Red;
        this.btnFechar.Location = new System.Drawing.Point(879, 517);
        this.btnFechar.Name = "btnFechar";
        this.btnFechar.Size = new System.Drawing.Size(93, 32);
        this.btnFechar.TabIndex = 6;
        this.btnFechar.Text = "Finalizar";
        this.btnFechar.UseVisualStyleBackColor = true;
        this.btnFechar.Click += new System.EventHandler(this.btnFinalizar_Click);
        //
        // btnCriarNovaConta
        //
        this.btnCriarNovaConta.AutoSize = true;
        this.btnCriarNovaConta.AutoSizeMode                                =
System.Windows.Forms.AutoSizeMode.GrowAndShrink;
        this.btnCriarNovaConta.Font = new System.Drawing.Font("Times New Roman",
14.25F,        System.Drawing.FontStyle.Bold,        System.Drawing.GraphicsUnit.Point,
((byte)(0)));
        this.btnCriarNovaConta.Location = new System.Drawing.Point(12, 517);
        this.btnCriarNovaConta.Name = "btnCriarNovaConta";
        this.btnCriarNovaConta.Size = new System.Drawing.Size(168, 32);
        this.btnCriarNovaConta.TabIndex = 7;
        this.btnCriarNovaConta.Text = "Criar Nova Conta";
        this.btnCriarNovaConta.UseVisualStyleBackColor = true;
        this.btnCriarNovaConta.Click                 +=                    new
System.EventHandler(this.btnCriarNovaConta_Click);
        //
        // NovaGuia
        //
        this.AutoScaleDimensions = new System.Drawing.SizeF(6F, 14F);
        this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font;
        this.BackgroundImage = global::AppDados.Properties.Resources.bg2;
        this.BackgroundImageLayout = System.Windows.Forms.ImageLayout.Stretch;
        this.ClientSize = new System.Drawing.Size(984, 561);
        this.Controls.Add(this.btnCriarNovaConta);
        this.Controls.Add(this.btnFechar);
        this.Controls.Add(this.btnCriarPedido);
        this.Controls.Add(this.gbEncomenda);
```

```
            this.Controls.Add(this.btnCriarCliente);
            this.Controls.Add(this.gbOficina);
            this.Font = new System.Drawing.Font("Times New Roman", 8.25F,
System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point, ((byte)(0)));
            this.MaximumSize = new System.Drawing.Size(1000, 600);
            this.MinimumSize = new System.Drawing.Size(1000, 600);
            this.Name = "NovaGuia";
            this.Text = "Lamborghini App - Nova Guia";
            this.gbOficina.ResumeLayout(false);
            this.gbOficina.PerformLayout();
            this.gbEncomenda.ResumeLayout(false);
            this.gbEncomenda.PerformLayout();
            ((System.ComponentModel.ISupportInitialize)(this.numQuantidade)).EndInit();
            this.ResumeLayout(false);
            this.PerformLayout();

        }

        #endregion

        private System.Windows.Forms.GroupBox gbOficina;
        private System.Windows.Forms.Button btnCriarCliente;
        private System.Windows.Forms.Label lblIdCliente;
        private System.Windows.Forms.Label lblNomeOficina;
        private System.Windows.Forms.TextBox textBox3;
        private System.Windows.Forms.TextBox textBox4;
        private System.Windows.Forms.Label label1;
        private System.Windows.Forms.Label label2;
        private System.Windows.Forms.TextBox txtIdOficina;
        private System.Windows.Forms.TextBox txtNomeOficina;
        private System.Windows.Forms.GroupBox gbEncomenda;
        private System.Windows.Forms.Label lblData;
        private System.Windows.Forms.Label lblQuantidade;
        private System.Windows.Forms.Button btnCriarPedido;
        private System.Windows.Forms.Button btnFechar;
        private System.Windows.Forms.Button btnCriarNovaConta;
        private System.Windows.Forms.DateTimePicker dtpData;
        private System.Windows.Forms.NumericUpDown numQuantidade;
    }
}
```

ANEXO C

```
namespace AppDados
{
    partial class Pedidos
    {
        /// <summary>
```

```
        /// Required designer variable.
        /// </summary>
        private System.ComponentModel.IContainer components = null;

        /// <summary>
        /// Clean up any resources being used.
        /// </summary>
        /// <param name="disposing">true if managed resources should be disposed; otherwise,
false.</param>
        protected override void Dispose(bool disposing)
        {
            if (disposing && (components != null))
            {
                components.Dispose();
            }
            base.Dispose(disposing);
        }

        #region Windows Form Designer generated code

        /// <summary>
        /// Required method for Designer support - do not modify
        /// the contents of this method with the code editor.
        /// </summary>
        private void InitializeComponent()
        {
            this.lblIdPedido = new System.Windows.Forms.Label();
            this.txtIdPedido = new System.Windows.Forms.TextBox();
            this.btnProcurarPorIdPedido = new System.Windows.Forms.Button();
            this.label1 = new System.Windows.Forms.Label();
            this.dtpPreencherData = new System.Windows.Forms.DateTimePicker();
            this.dgvPedidosClientes = new System.Windows.Forms.DataGridView();
            this.btnCancelarPedido = new System.Windows.Forms.Button();
            this.btnConfirmarPedido = new System.Windows.Forms.Button();
            this.btnFinalizarPedidos = new System.Windows.Forms.Button();
            this.btnIdentificarCliente = new System.Windows.Forms.Button();

((System.ComponentModel.ISupportInitialize)(this.dgvPedidosClientes)).BeginInit();
            this.SuspendLayout();
            //
            // lblIdPedido
            //
            this.lblIdPedido.AutoSize = true;
            this.lblIdPedido.BackColor = System.Drawing.Color.Transparent;
            this.lblIdPedido.Font = new System.Drawing.Font("Times New Roman", 20.25F,
System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point, ((byte)(0)));
            this.lblIdPedido.ForeColor = System.Drawing.Color.White;
            this.lblIdPedido.Location = new System.Drawing.Point(12, 37);
            this.lblIdPedido.Name = "lblIdPedido";
            this.lblIdPedido.Size = new System.Drawing.Size(141, 31);
```

```
            this.lblIdPedido.TabIndex = 0;
            this.lblIdPedido.Text = "ID Pedido:";
            //
            // txtIdPedido
            //
            this.txtIdPedido.Font = new System.Drawing.Font("Times New Roman", 14.25F,
System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point, ((byte)(0)));
            this.txtIdPedido.Location = new System.Drawing.Point(159, 42);
            this.txtIdPedido.Name = "txtIdPedido";
            this.txtIdPedido.Size = new System.Drawing.Size(167, 29);
            this.txtIdPedido.TabIndex = 1;
            //
            // btnProcurarPorIdPedido
            //
            this.btnProcurarPorIdPedido.AutoSize = true;
            this.btnProcurarPorIdPedido.AutoSizeMode                                    =
System.Windows.Forms.AutoSizeMode.GrowAndShrink;
            this.btnProcurarPorIdPedido.Font = new System.Drawing.Font("Times New
Roman", 14.25F, System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point,
((byte)(0)));
            this.btnProcurarPorIdPedido.Location = new System.Drawing.Point(332, 40);
            this.btnProcurarPorIdPedido.Name = "btnProcurarPorIdPedido";
            this.btnProcurarPorIdPedido.Size = new System.Drawing.Size(243, 32);
            this.btnProcurarPorIdPedido.TabIndex = 2;
            this.btnProcurarPorIdPedido.Text = "Procurar pedido / Atualizar";
            this.btnProcurarPorIdPedido.UseVisualStyleBackColor = true;
            this.btnProcurarPorIdPedido.Click                    +=                    new
System.EventHandler(this.btnProcurarPorIdPedido_Click);
            //
            // label1
            //
            this.label1.AutoSize = true;
            this.label1.BackColor = System.Drawing.Color.Transparent;
            this.label1.Font = new System.Drawing.Font("Times New Roman", 14.25F,
System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point, ((byte)(0)));
            this.label1.ForeColor = System.Drawing.Color.White;
            this.label1.Location = new System.Drawing.Point(8, 114);
            this.label1.Name = "label1";
            this.label1.Size = new System.Drawing.Size(310, 44);
            this.label1.TabIndex = 3;
            this.label1.Text = "Se estiver a confirmar um pedido, \r\nespecifique a data de
preenchimento:";
            //
            // dtpPreencherData
            //
            this.dtpPreencherData.Font = new System.Drawing.Font("Times New Roman",
14.25F,      System.Drawing.FontStyle.Regular,      System.Drawing.GraphicsUnit.Point,
((byte)(0)));
            this.dtpPreencherData.Format                                                =
System.Windows.Forms.DateTimePickerFormat.Short;
```

```
        this.dtpPreencherData.Location = new System.Drawing.Point(12, 161);
        this.dtpPreencherData.Name = "dtpPreencherData";
        this.dtpPreencherData.Value = System.DateTime.Today;
        this.dtpPreencherData.MinDate = System.DateTime.Today;
        this.dtpPreencherData.Size = new System.Drawing.Size(200, 29);
        this.dtpPreencherData.TabIndex = 4;
        //
        // dgvPedidosClientes
        //
        this.dgvPedidosClientes.ColumnHeadersHeightSizeMode =
System.Windows.Forms.DataGridViewColumnHeadersHeightSizeMode.AutoSize;
        this.dgvPedidosClientes.Location = new System.Drawing.Point(12, 196);
        this.dgvPedidosClientes.Name = "dgvPedidosClientes";
        this.dgvPedidosClientes.ReadOnly = true;
        this.dgvPedidosClientes.RowHeadersVisible = false;
        this.dgvPedidosClientes.Size = new System.Drawing.Size(648, 202);
        this.dgvPedidosClientes.TabIndex = 5;
        //
        // btnCancelarPedido
        //
        this.btnCancelarPedido.AutoSize = true;
        this.btnCancelarPedido.AutoSizeMode =
System.Windows.Forms.AutoSizeMode.GrowAndShrink;
        this.btnCancelarPedido.Font = new System.Drawing.Font("Times New Roman",
14.25F, System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point,
((byte)(0)));
        this.btnCancelarPedido.Location = new System.Drawing.Point(12, 479);
        this.btnCancelarPedido.Name = "btnCancelarPedido";
        this.btnCancelarPedido.Size = new System.Drawing.Size(156, 32);
        this.btnCancelarPedido.TabIndex = 6;
        this.btnCancelarPedido.Text = "Cancelar Pedido";
        this.btnCancelarPedido.UseVisualStyleBackColor = true;
        this.btnCancelarPedido.Click += new
System.EventHandler(this.btnCancelarPedido_Click);
        //
        // btnConfirmarPedido
        //
        this.btnConfirmarPedido.AutoSize = true;
        this.btnConfirmarPedido.AutoSizeMode =
System.Windows.Forms.AutoSizeMode.GrowAndShrink;
        this.btnConfirmarPedido.Font = new System.Drawing.Font("Times New Roman",
14.25F, System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point,
((byte)(0)));
        this.btnConfirmarPedido.Location = new System.Drawing.Point(12, 517);
        this.btnConfirmarPedido.Name = "btnConfirmarPedido";
        this.btnConfirmarPedido.Size = new System.Drawing.Size(167, 32);
        this.btnConfirmarPedido.TabIndex = 7;
        this.btnConfirmarPedido.Text = "Confirmar Pedido";
        this.btnConfirmarPedido.UseVisualStyleBackColor = true;
```

```
            this.btnConfirmarPedido.Click += new
System.EventHandler(this.btnPreencherPedido_Click);
            //
            // btnFinalizarPedidos
            //
            this.btnFinalizarPedidos.AutoSize = true;
            this.btnFinalizarPedidos.AutoSizeMode =
System.Windows.Forms.AutoSizeMode.GrowAndShrink;
            this.btnFinalizarPedidos.Font = new System.Drawing.Font("Times New Roman",
14.25F, System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point,
((byte)(0)));
            this.btnFinalizarPedidos.ForeColor = System.Drawing.Color.Red;
            this.btnFinalizarPedidos.Location = new System.Drawing.Point(879, 517);
            this.btnFinalizarPedidos.Name = "btnFinalizarPedidos";
            this.btnFinalizarPedidos.Size = new System.Drawing.Size(93, 32);
            this.btnFinalizarPedidos.TabIndex = 8;
            this.btnFinalizarPedidos.Text = "Finalizar";
            this.btnFinalizarPedidos.UseVisualStyleBackColor = true;
            this.btnFinalizarPedidos.Click += new
System.EventHandler(this.btnFinalizarPedidos_Click);
            //
            // btnIdentificarCliente
            //
            this.btnIdentificarCliente.AutoSize = true;
            this.btnIdentificarCliente.AutoSizeMode =
System.Windows.Forms.AutoSizeMode.GrowAndShrink;
            this.btnIdentificarCliente.Font = new System.Drawing.Font("Times New Roman",
14.25F, System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point,
((byte)(0)));
            this.btnIdentificarCliente.Location = new System.Drawing.Point(666, 196);
            this.btnIdentificarCliente.Name = "btnIdentificarCliente";
            this.btnIdentificarCliente.Size = new System.Drawing.Size(168, 32);
            this.btnIdentificarCliente.TabIndex = 9;
            this.btnIdentificarCliente.Text = "Identificar Cliente";
            this.btnIdentificarCliente.UseVisualStyleBackColor = true;
            this.btnIdentificarCliente.Click += new
System.EventHandler(this.btnIdentificarCliente_Click);
            //
            // Pedidos
            //
            this.AutoScaleDimensions = new System.Drawing.SizeF(6F, 14F);
            this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font;
            this.BackgroundImage = global::AppDados.Properties.Resources.bg2;
            this.BackgroundImageLayout = System.Windows.Forms.ImageLayout.Stretch;
            this.ClientSize = new System.Drawing.Size(984, 561);
            this.Controls.Add(this.btnIdentificarCliente);
            this.Controls.Add(this.btnFinalizarPedidos);
            this.Controls.Add(this.btnConfirmarPedido);
            this.Controls.Add(this.btnCancelarPedido);
            this.Controls.Add(this.dgvPedidosClientes);
```

```
        this.Controls.Add(this.dtpPreencherData);
        this.Controls.Add(this.label1);
        this.Controls.Add(this.btnProcurarPorIdPedido);
        this.Controls.Add(this.txtIdPedido);
        this.Controls.Add(this.lblIdPedido);
        this.Font = new System.Drawing.Font("Times New Roman", 8.25F,
System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point, ((byte)(0)));
        this.MaximumSize = new System.Drawing.Size(1000, 600);
        this.MinimumSize = new System.Drawing.Size(1000, 600);
        this.Name = "Pedidos";
        this.Text = "Lamborghini App - Pedidos";
        ((System.ComponentModel.ISupportInitialize)(this.dgvPedidosClientes)).EndInit();
        this.ResumeLayout(false);
        this.PerformLayout();

    }

    #endregion

    private System.Windows.Forms.Label lblIdPedido;
    private System.Windows.Forms.TextBox txtIdPedido;
    private System.Windows.Forms.Button btnProcurarPorIdPedido;
    private System.Windows.Forms.Label label1;
    private System.Windows.Forms.DateTimePicker dtpPreencherData;
    private System.Windows.Forms.DataGridView dgvPedidosClientes;
    private System.Windows.Forms.Button btnCancelarPedido;
    private System.Windows.Forms.Button btnConfirmarPedido;
    private System.Windows.Forms.Button btnFinalizarPedidos;
    private System.Windows.Forms.Button btnIdentificarCliente;
  }
}
```

ANEXO D

```
namespace AppDados
{
    partial class Clientes
    {
        /// <summary>
        /// Required designer variable.
        /// </summary>
        private System.ComponentModel.IContainer components = null;

        /// <summary>
        /// Clean up any resources being used.
        /// </summary>
        /// <param name="disposing">true if managed resources should be disposed;
otherwise, false.</param>
        protected override void Dispose(bool disposing)
        {
            if (disposing && (components != null))
            {
```

```
                components.Dispose();
            }
            base.Dispose(disposing);
        }

        #region Windows Form Designer generated code

        /// <summary>
        /// Required method for Designer support - do not modify
        /// the contents of this method with the code editor.
        /// </summary>
        private void InitializeComponent()
        {
            this.components = new System.ComponentModel.Container();
            this.customerBindingSource = new
System.Windows.Forms.BindingSource(this.components);
            this.btnFechar = new System.Windows.Forms.Button();
            this.btnFinalizar = new System.Windows.Forms.Button();
            this.clientesDS1BindingSource = new
System.Windows.Forms.BindingSource(this.components);
            this.dgvClientes = new System.Windows.Forms.DataGridView();
            this.dataGridViewTextBoxColumn1 = new
System.Windows.Forms.DataGridViewTextBoxColumn();
            this.dataGridViewTextBoxColumn2 = new
System.Windows.Forms.DataGridViewTextBoxColumn();
            this.customerBindingSource1 = new
System.Windows.Forms.BindingSource(this.components);
            this.clientesDS = new AppDados.ClientesDS();
            this.customerTableAdapter = new
AppDados.ClientesDSTableAdapters.CustomerTableAdapter();

((System.ComponentModel.ISupportInitialize)(this.customerBindingSource)).BeginInit()
;

((System.ComponentModel.ISupportInitialize)(this.clientesDS1BindingSource)).BeginIni
t();

((System.ComponentModel.ISupportInitialize)(this.dgvClientes)).BeginInit();

((System.ComponentModel.ISupportInitialize)(this.customerBindingSource1)).BeginInit(
);

((System.ComponentModel.ISupportInitialize)(this.clientesDS)).BeginInit();
            this.SuspendLayout();
            //
            // customerBindingSource
            //
            this.customerBindingSource.DataMember = "Customer";
            //
            // btnFechar
            //
            this.btnFechar.AutoSizeMode =
System.Windows.Forms.AutoSizeMode.GrowAndShrink;
            this.btnFechar.Font = new System.Drawing.Font("Times New Roman", 14.25F,
System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point, ((byte)(0)));
            this.btnFechar.ForeColor = System.Drawing.Color.Red;
            this.btnFechar.Location = new System.Drawing.Point(129, 517);
            this.btnFechar.Name = "btnFechar";
            this.btnFechar.Size = new System.Drawing.Size(93, 32);
            this.btnFechar.TabIndex = 1;
            this.btnFechar.Text = "Finalizar";
            this.btnFechar.UseVisualStyleBackColor = true;
```

```
            this.btnFechar.Click += new
System.EventHandler(this.btnFinalizar_Click);
            //
            // btnFinalizar
            //
            this.btnFinalizar.Cursor = System.Windows.Forms.Cursors.Arrow;
            this.btnFinalizar.ForeColor = System.Drawing.SystemColors.Desktop;
            this.btnFinalizar.Location = new System.Drawing.Point(12, 526);
            this.btnFinalizar.Name = "btnFinalizar";
            this.btnFinalizar.Size = new System.Drawing.Size(210, 23);
            this.btnFinalizar.TabIndex = 1;
            this.btnFinalizar.Text = "Fechar";
            this.btnFinalizar.UseVisualStyleBackColor = true;
            this.btnFinalizar.Click += new
System.EventHandler(this.btnFinalizar_Click);
            //
            // dgvClientes
            //
            this.dgvClientes.AutoGenerateColumns = false;
            this.dgvClientes.AutoSizeColumnsMode =
System.Windows.Forms.DataGridViewAutoSizeColumnsMode.AllCells;
            this.dgvClientes.ColumnHeadersHeightSizeMode =
System.Windows.Forms.DataGridViewColumnHeadersHeightSizeMode.AutoSize;
            this.dgvClientes.Columns.AddRange(new
System.Windows.Forms.DataGridViewColumn[] {
            this.dataGridViewTextBoxColumn1,
            this.dataGridViewTextBoxColumn2});
            this.dgvClientes.DataSource = this.customerBindingSource1;
            this.dgvClientes.Location = new System.Drawing.Point(12, 12);
            this.dgvClientes.Name = "dgvClientes";
            this.dgvClientes.RowHeadersVisible = false;
            this.dgvClientes.Size = new System.Drawing.Size(210, 499);
            this.dgvClientes.TabIndex = 2;
            //
            // dataGridViewTextBoxColumn1
            //
            this.dataGridViewTextBoxColumn1.DataPropertyName = "CustomerID";
            this.dataGridViewTextBoxColumn1.HeaderText = "CustomerID";
            this.dataGridViewTextBoxColumn1.Name = "dataGridViewTextBoxColumn1";
            this.dataGridViewTextBoxColumn1.ReadOnly = true;
            this.dataGridViewTextBoxColumn1.Width = 85;
            //
            // dataGridViewTextBoxColumn2
            //
            this.dataGridViewTextBoxColumn2.DataPropertyName = "CustomerName";
            this.dataGridViewTextBoxColumn2.HeaderText = "CustomerName";
            this.dataGridViewTextBoxColumn2.Name = "dataGridViewTextBoxColumn2";
            this.dataGridViewTextBoxColumn2.Width = 102;
            //
            // customerBindingSource1
            //
            this.customerBindingSource1.DataMember = "Customer";
            this.customerBindingSource1.DataSource = this.clientesDS;
            //
            // clientesDS
            //
            this.clientesDS.DataSetName = "ClientesDS";
            this.clientesDS.SchemaSerializationMode =
System.Data.SchemaSerializationMode.IncludeSchema;
            //
            // customerTableAdapter
            //
```

```
            this.customerTableAdapter.ClearBeforeFill = true;
            //
            // Clientes
            //
            this.AutoScaleDimensions = new System.Drawing.SizeF(6F, 13F);
            this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font;
            this.AutoSize = true;
            this.BackgroundImage = global::AppDados.Properties.Resources.bg2;
            this.BackgroundImageLayout = System.Windows.Forms.ImageLayout.Stretch;
            this.ClientSize = new System.Drawing.Size(234, 561);
            this.Controls.Add(this.dgvClientes);
            this.Controls.Add(this.btnFechar);
            this.MaximumSize = new System.Drawing.Size(250, 600);
            this.MinimumSize = new System.Drawing.Size(250, 600);
            this.Name = "Clientes";
            this.Text = "Lamborghini App - Clientes";
            this.Load += new System.EventHandler(this.Clientes_Load);

((System.ComponentModel.ISupportInitialize)(this.customerBindingSource)).EndInit();

((System.ComponentModel.ISupportInitialize)(this.clientesDS1BindingSource)).EndInit(
);

((System.ComponentModel.ISupportInitialize)(this.dgvClientes)).EndInit();

((System.ComponentModel.ISupportInitialize)(this.customerBindingSource1)).EndInit();
            ((System.ComponentModel.ISupportInitialize)(this.clientesDS)).EndInit();
            this.ResumeLayout(false);

        }

        #endregion
        private System.Windows.Forms.Button btnFechar;
        private System.Windows.Forms.BindingSource customerBindingSource;
        private System.Windows.Forms.Button btnFinalizar;
        private System.Windows.Forms.BindingSource clientesDS1BindingSource;
        private System.Windows.Forms.DataGridView dgvClientes;
        private ClientesDS clientesDS;
        private System.Windows.Forms.BindingSource customerBindingSource1;
        private ClientesDSTableAdapters.CustomerTableAdapter customerTableAdapter;
        private System.Windows.Forms.DataGridViewTextBoxColumn
dataGridViewTextBoxColumn1;
        private System.Windows.Forms.DataGridViewTextBoxColumn
dataGridViewTextBoxColumn2;
    }
}
```

ANEXO E

```
namespace AppDados
{
   partial class Produto
   {
      /// <summary>
      /// Required designer variable.
      /// </summary>
      private System.ComponentModel.IContainer components = null;
```

```
        /// <summary>
        /// Clean up any resources being used.
        /// </summary>
        /// <param name="disposing">true if managed resources should be disposed; otherwise,
false.</param>
        protected override void Dispose(bool disposing)
        {
            if (disposing && (components != null))
            {
                components.Dispose();
            }
            base.Dispose(disposing);
        }

        #region Windows Form Designer generated code

        /// <summary>
        /// Required method for Designer support - do not modify
        /// the contents of this method with the code editor.
        /// </summary>
        private void InitializeComponent()
        {
            this.components = new System.ComponentModel.Container();
            this.txtID = new System.Windows.Forms.TextBox();
            this.pbImagem = new System.Windows.Forms.PictureBox();
            this.label1 = new System.Windows.Forms.Label();
            this.btnPesquisar = new System.Windows.Forms.Button();
            this.btnMostrar = new System.Windows.Forms.Button();
            this.btnGuardar = new System.Windows.Forms.Button();
            this.btnFinalizar = new System.Windows.Forms.Button();
            this.txtDescricao = new System.Windows.Forms.TextBox();
            this.label2 = new System.Windows.Forms.Label();
            this.label3 = new System.Windows.Forms.Label();
            this.dgvImagensDisp = new System.Windows.Forms.DataGridView();
            this.imagensBindingSource1                    =                    new
System.Windows.Forms.BindingSource(this.components);
            this.imagensBindingSource                     =                    new
System.Windows.Forms.BindingSource(this.components);
            this.label4 = new System.Windows.Forms.Label();
            this.btnAtualizaLista = new System.Windows.Forms.Button();
            this.bDAppDadosDataSetBindingSource           =                    new
System.Windows.Forms.BindingSource(this.components);
            this.btnEliminar = new System.Windows.Forms.Button();
            this.imagensDS1 = new AppDados.ImagensDS1();
            this.imagesBindingSource                      =                    new
System.Windows.Forms.BindingSource(this.components);
            this.imagesTableAdapter                       =                    new
AppDados.ImagensDS1TableAdapters.ImagesTableAdapter();
```

```
        this.iDDataGridViewTextBoxColumn = new
System.Windows.Forms.DataGridViewTextBoxColumn();
        this.descriptionDataGridViewTextBoxColumn = new
System.Windows.Forms.DataGridViewTextBoxColumn();
        ((System.ComponentModel.ISupportInitialize)(this.pbImagem)).BeginInit();
        ((System.ComponentModel.ISupportInitialize)(this.dgvImagensDisp)).BeginInit();

((System.ComponentModel.ISupportInitialize)(this.imagensBindingSource1)).BeginInit();

((System.ComponentModel.ISupportInitialize)(this.imagensBindingSource)).BeginInit();

((System.ComponentModel.ISupportInitialize)(this.bDAppDadosDataSetBindingSource)).
BeginInit();
        ((System.ComponentModel.ISupportInitialize)(this.imagensDS1)).BeginInit();

((System.ComponentModel.ISupportInitialize)(this.imagesBindingSource)).BeginInit();
        this.SuspendLayout();
        //
        // txtID
        //
        this.txtID.Font = new System.Drawing.Font("Times New Roman", 14.25F,
System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point, ((byte)(0)));
        this.txtID.Location = new System.Drawing.Point(86, 89);
        this.txtID.Name = "txtID";
        this.txtID.Size = new System.Drawing.Size(122, 29);
        this.txtID.TabIndex = 0;
        //
        // pbImagem
        //
        this.pbImagem.BackColor = System.Drawing.Color.Transparent;
        this.pbImagem.BorderStyle = System.Windows.Forms.BorderStyle.FixedSingle;
        this.pbImagem.Location = new System.Drawing.Point(463, 9);
        this.pbImagem.Name = "pbImagem";
        this.pbImagem.Size = new System.Drawing.Size(509, 338);
        this.pbImagem.SizeMode =
System.Windows.Forms.PictureBoxSizeMode.StretchImage;
        this.pbImagem.TabIndex = 1;
        this.pbImagem.TabStop = false;
        //
        // label1
        //
        this.label1.AutoSize = true;
        this.label1.BackColor = System.Drawing.Color.Transparent;
        this.label1.Font = new System.Drawing.Font("Times New Roman", 14.25F,
System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point, ((byte)(0)));
        this.label1.ForeColor = System.Drawing.Color.White;
        this.label1.Location = new System.Drawing.Point(12, 92);
        this.label1.Name = "label1";
        this.label1.Size = new System.Drawing.Size(36, 22);
        this.label1.TabIndex = 2;
```

```
        this.label1.Text = "ID:";
        //
        // btnPesquisar
        //
        this.btnPesquisar.AutoSize = true;
        this.btnPesquisar.AutoSizeMode                                    =
System.Windows.Forms.AutoSizeMode.GrowAndShrink;
        this.btnPesquisar.Font = new System.Drawing.Font("Times New Roman", 14.25F,
System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point, ((byte)(0)));
        this.btnPesquisar.Location = new System.Drawing.Point(765, 353);
        this.btnPesquisar.Name = "btnPesquisar";
        this.btnPesquisar.Size = new System.Drawing.Size(110, 32);
        this.btnPesquisar.TabIndex = 3;
        this.btnPesquisar.Text = "Carregar...";
        this.btnPesquisar.UseVisualStyleBackColor = true;
        this.btnPesquisar.Click += new System.EventHandler(this.btnPesquisar_Click);
        //
        // btnMostrar
        //
        this.btnMostrar.AutoSize = true;
        this.btnMostrar.AutoSizeMode                                      =
System.Windows.Forms.AutoSizeMode.GrowAndShrink;
        this.btnMostrar.Font = new System.Drawing.Font("Times New Roman", 14.25F,
System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point, ((byte)(0)));
        this.btnMostrar.Location = new System.Drawing.Point(229, 87);
        this.btnMostrar.Name = "btnMostrar";
        this.btnMostrar.Size = new System.Drawing.Size(88, 32);
        this.btnMostrar.TabIndex = 4;
        this.btnMostrar.Text = "Mostrar";
        this.btnMostrar.UseVisualStyleBackColor = true;
        this.btnMostrar.Click += new System.EventHandler(this.btnMostrar_Click);
        //
        // btnGuardar
        //
        this.btnGuardar.AutoSize = true;
        this.btnGuardar.AutoSizeMode                                      =
System.Windows.Forms.AutoSizeMode.GrowAndShrink;
        this.btnGuardar.Font = new System.Drawing.Font("Times New Roman", 14.25F,
System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point, ((byte)(0)));
        this.btnGuardar.Location = new System.Drawing.Point(881, 353);
        this.btnGuardar.Name = "btnGuardar";
        this.btnGuardar.Size = new System.Drawing.Size(91, 32);
        this.btnGuardar.TabIndex = 5;
        this.btnGuardar.Text = "Guardar";
        this.btnGuardar.UseVisualStyleBackColor = true;
        this.btnGuardar.Click += new System.EventHandler(this.btnGuardar_Click);
        //
        // btnFinalizar
        //
        this.btnFinalizar.AutoSize = true;
```

```
        this.btnFinalizar.AutoSizeMode                                    =
System.Windows.Forms.AutoSizeMode.GrowAndShrink;
        this.btnFinalizar.Font = new System.Drawing.Font("Times New Roman", 14.25F,
System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point, ((byte)(0)));
        this.btnFinalizar.ForeColor = System.Drawing.Color.Red;
        this.btnFinalizar.Location = new System.Drawing.Point(879, 517);
        this.btnFinalizar.Name = "btnFinalizar";
        this.btnFinalizar.Size = new System.Drawing.Size(93, 32);
        this.btnFinalizar.TabIndex = 6;
        this.btnFinalizar.Text = "Finalizar";
        this.btnFinalizar.UseVisualStyleBackColor = true;
        this.btnFinalizar.Click += new System.EventHandler(this.btnFinalizar_Click);
        //
        // txtDescricao
        //
        this.txtDescricao.Font = new System.Drawing.Font("Times New Roman", 14.25F,
System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point, ((byte)(0)));
        this.txtDescricao.Location = new System.Drawing.Point(86, 145);
        this.txtDescricao.Name = "txtDescricao";
        this.txtDescricao.Size = new System.Drawing.Size(122, 29);
        this.txtDescricao.TabIndex = 7;
        //
        // label2
        //
        this.label2.AutoSize = true;
        this.label2.BackColor = System.Drawing.Color.Transparent;
        this.label2.Font = new System.Drawing.Font("Times New Roman", 14.25F,
System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point, ((byte)(0)));
        this.label2.ForeColor = System.Drawing.Color.White;
        this.label2.Location = new System.Drawing.Point(12, 148);
        this.label2.Name = "label2";
        this.label2.Size = new System.Drawing.Size(48, 22);
        this.label2.TabIndex = 8;
        this.label2.Text = "Info:";
        //
        // label3
        //
        this.label3.AutoSize = true;
        this.label3.BackColor = System.Drawing.Color.Transparent;
        this.label3.Font = new System.Drawing.Font("Times New Roman", 20.25F,
System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point, ((byte)(0)));
        this.label3.ForeColor = System.Drawing.Color.White;
        this.label3.Location = new System.Drawing.Point(12, 9);
        this.label3.Name = "label3";
        this.label3.Size = new System.Drawing.Size(445, 31);
        this.label3.TabIndex = 9;
        this.label3.Text = "Imagens das Partes incluídas no KIT";
        //
        // dgvImagensDisp
        //
```

```
        this.dgvImagensDisp.AllowUserToOrderColumns = true;
        this.dgvImagensDisp.AutoGenerateColumns = false;
        this.dgvImagensDisp.AutoSizeColumnsMode =
System.Windows.Forms.DataGridViewAutoSizeColumnsMode.Fill;
        this.dgvImagensDisp.Columns.AddRange(new
System.Windows.Forms.DataGridViewColumn[] {
        this.iDDataGridViewTextBoxColumn,
        this.descriptionDataGridViewTextBoxColumn});
        this.dgvImagensDisp.DataSource = this.imagesBindingSource;
        this.dgvImagensDisp.GridColor = System.Drawing.SystemColors.ActiveBorder;
        this.dgvImagensDisp.Location = new System.Drawing.Point(16, 251);
        this.dgvImagensDisp.Name = "dgvImagensDisp";
        this.dgvImagensDisp.ReadOnly = true;
        this.dgvImagensDisp.RowHeadersVisible = false;
        this.dgvImagensDisp.Size = new System.Drawing.Size(192, 298);
        this.dgvImagensDisp.TabIndex = 10;
        //
        // label4
        //
        this.label4.AutoSize = true;
        this.label4.BackColor = System.Drawing.Color.Transparent;
        this.label4.Font = new System.Drawing.Font("Times New Roman", 14.25F,
System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point, ((byte)(0)));
        this.label4.ForeColor = System.Drawing.Color.White;
        this.label4.Location = new System.Drawing.Point(12, 226);
        this.label4.Name = "label4";
        this.label4.Size = new System.Drawing.Size(182, 22);
        this.label4.TabIndex = 11;
        this.label4.Text = "Imagens Disponíveis:";
        //
        // btnAtualizaLista
        //
        this.btnAtualizaLista.AutoSize = true;
        this.btnAtualizaLista.AutoSizeMode =
System.Windows.Forms.AutoSizeMode.GrowAndShrink;
        this.btnAtualizaLista.Font = new System.Drawing.Font("Times New Roman",
14.25F, System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point,
((byte)(0)));
        this.btnAtualizaLista.Location = new System.Drawing.Point(214, 517);
        this.btnAtualizaLista.Name = "btnAtualizaLista";
        this.btnAtualizaLista.Size = new System.Drawing.Size(141, 32);
        this.btnAtualizaLista.TabIndex = 12;
        this.btnAtualizaLista.Text = "Atualizar Lista";
        this.btnAtualizaLista.UseVisualStyleBackColor = true;
        this.btnAtualizaLista.Click += new
System.EventHandler(this.btnAtualizaLista_Click);
        //
        // btnEliminar
        //
        this.btnEliminar.AutoSize = true;
```

```
        this.btnEliminar.AutoSizeMode =
System.Windows.Forms.AutoSizeMode.GrowAndShrink;
        this.btnEliminar.Font = new System.Drawing.Font("Times New Roman", 14.25F,
System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point, ((byte)(0)));
        this.btnEliminar.Location = new System.Drawing.Point(323, 87);
        this.btnEliminar.Name = "btnEliminar";
        this.btnEliminar.Size = new System.Drawing.Size(91, 32);
        this.btnEliminar.TabIndex = 13;
        this.btnEliminar.Text = "Eliminar";
        this.btnEliminar.UseVisualStyleBackColor = true;
        this.btnEliminar.Click += new System.EventHandler(this.btnEliminar_Click);
        //
        // imagensDS1
        //
        this.imagensDS1.DataSetName = "ImagensDS1";
        this.imagensDS1.SchemaSerializationMode =
System.Data.SchemaSerializationMode.IncludeSchema;
        //
        // imagesBindingSource
        //
        this.imagesBindingSource.DataMember = "Images";
        this.imagesBindingSource.DataSource = this.imagensDS1;
        //
        // imagesTableAdapter
        //
        this.imagesTableAdapter.ClearBeforeFill = true;
        //
        // iDDataGridViewTextBoxColumn
        //
        this.iDDataGridViewTextBoxColumn.DataPropertyName = "ID";
        this.iDDataGridViewTextBoxColumn.HeaderText = "ID";
        this.iDDataGridViewTextBoxColumn.Name = "iDDataGridViewTextBoxColumn";
        this.iDDataGridViewTextBoxColumn.ReadOnly = true;
        //
        // descriptionDataGridViewTextBoxColumn
        //
        this.descriptionDataGridViewTextBoxColumn.DataPropertyName = "Description";
        this.descriptionDataGridViewTextBoxColumn.HeaderText = "Description";
        this.descriptionDataGridViewTextBoxColumn.Name =
"descriptionDataGridViewTextBoxColumn";
        this.descriptionDataGridViewTextBoxColumn.ReadOnly = true;
        //
        // Produto
        //
        this.AutoScaleDimensions = new System.Drawing.SizeF(6F, 14F);
        this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font;
        this.BackgroundImage = global::AppDados.Properties.Resources.bg2;
        this.BackgroundImageLayout = System.Windows.Forms.ImageLayout.Stretch;
        this.ClientSize = new System.Drawing.Size(984, 561);
        this.Controls.Add(this.btnEliminar);
```

```
        this.Controls.Add(this.btnAtualizaLista);
        this.Controls.Add(this.label4);
        this.Controls.Add(this.dgvImagensDisp);
        this.Controls.Add(this.label3);
        this.Controls.Add(this.label2);
        this.Controls.Add(this.txtDescricao);
        this.Controls.Add(this.btnFinalizar);
        this.Controls.Add(this.btnGuardar);
        this.Controls.Add(this.btnMostrar);
        this.Controls.Add(this.btnPesquisar);
        this.Controls.Add(this.label1);
        this.Controls.Add(this.pbImagem);
        this.Controls.Add(this.txtID);
        this.Font = new System.Drawing.Font("Times New Roman", 8.25F,
System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point, ((byte)(0)));
        this.MaximumSize = new System.Drawing.Size(1000, 600);
        this.MinimumSize = new System.Drawing.Size(1000, 600);
        this.Name = "Produto";
        this.Text = "Lamborghini App - Produto";
        this.Load += new System.EventHandler(this.Produto_Load);
        ((System.ComponentModel.ISupportInitialize)(this.pbImagem)).EndInit();
        ((System.ComponentModel.ISupportInitialize)(this.dgvImagensDisp)).EndInit();

((System.ComponentModel.ISupportInitialize)(this.imagensBindingSource1)).EndInit();

((System.ComponentModel.ISupportInitialize)(this.imagensBindingSource)).EndInit();

((System.ComponentModel.ISupportInitialize)(this.bDAppDadosDataSetBindingSource)).
EndInit();
        ((System.ComponentModel.ISupportInitialize)(this.imagensDS1)).EndInit();

((System.ComponentModel.ISupportInitialize)(this.imagesBindingSource)).EndInit();
        this.ResumeLayout(false);
        this.PerformLayout();
    }

    #endregion

    private System.Windows.Forms.TextBox txtID;
    private System.Windows.Forms.PictureBox pbImagem;
    private System.Windows.Forms.Label label1;
    private System.Windows.Forms.Button btnPesquisar;
    private System.Windows.Forms.Button btnMostrar;
    private System.Windows.Forms.Button btnGuardar;
    private System.Windows.Forms.Button btnFinalizar;
    private System.Windows.Forms.TextBox txtDescricao;
    private System.Windows.Forms.Label label2;
    private System.Windows.Forms.Label label3;
    private System.Windows.Forms.DataGridView dgvImagensDisp;
    private System.Windows.Forms.Label label4;
```

```
        private System.Windows.Forms.BindingSource imagensBindingSource;
        private System.Windows.Forms.Button btnAtualizaLista;
        private System.Windows.Forms.BindingSource bDAppDadosDataSetBindingSource;
        private System.Windows.Forms.Button btnEliminar;
        private System.Windows.Forms.BindingSource imagensBindingSource1;
        private                    System.Windows.Forms.DataGridViewTextBoxColumn
dataGridViewTextBoxColumn1;
        private                    System.Windows.Forms.DataGridViewTextBoxColumn
dataGridViewTextBoxColumn2;
        private ImagensDS1 imagensDS1;
        private System.Windows.Forms.BindingSource imagesBindingSource;
        private ImagensDS1TableAdapters.ImagesTableAdapter imagesTableAdapter;
        private                    System.Windows.Forms.DataGridViewTextBoxColumn
iDDataGridViewTextBoxColumn;
        private                    System.Windows.Forms.DataGridViewTextBoxColumn
descriptionDataGridViewTextBoxColumn;
    }
}
```

ANEXO F