INSTITUTO SUPERIOR DE ENGENHARIA DO PORTO
MESTRADO EM ENGENHARIA INFORMÁTICA




Visão computacional aplicada a processos de recuperação de animais desaparecidos

BRUNO MIGUEL PEREIRA GOMES
Outubro de 2017

POLITÉCNICO
DO PORTO

Instituto Superior de Engenharia do Porto

Visão computacional aplicada a processos de recuperação de animais desaparecidos

Bruno Miguel Pereira Gomes

Dissertação para obtenção do Grau de Mestre em Engenharia Informática, Área de Especialização em Engenharia de Software

Orientador: António José Rocha de Oliveira

Júri:

Presidente:

[Nome do Presidente, Categoria, Escola]

Vogais:

[Nome do Vogal1, Categoria, Escola]

[Nome do Vogal2, Categoria, Escola] (até 4 vogais)

Porto, outubro de 2017

Dedicatória

À minha família e namorada por todo o apoio e compreensão.

Resumo

Com base em estudos recentes foi possível concluir-se que existem cerca de 6,7 milhões de animais de companhia em Portugal, estando estes distribuídos por cerca de 2 milhões de habitações, o que representa cerca de 54% dos lares portugueses. A curiosidade, a busca por alimento ou os estados de recetividade sexual são alguns exemplos das causas que levam ao desaparecimento de milhares de animais todos os anos. Isto é algo que provoca dor e desespero aos seus detentores, uma vez que estes animais são muitas vezes considerados como membros da família.

Nesta era digital, muitos são os que cada vez mais recorrem à internet em busca de soluções para os seus problemas, pelo que é frequente a utilização das redes sociais e outras plataformas para a partilha e a procura de ajuda nestes casos. As plataformas existentes têm o foco muito centrado nas publicações de anúncios e pouco na aplicação de tecnologia que possa efetivamente apoiar os cidadãos de forma mais eficiente.

A criação de uma plataforma *online* que permitisse a colocação de anúncios de animais desaparecidos e encontrados de forma gratuita, aliada à utilização de componentes externos que pudessem auxiliar o processo de recuperação de animais, foi um dos principais objetivos deste projeto. A inclusão destes componentes, com diferentes formas de comunicação exigia uma solução de integração que permitisse a interoperabilidade entre ambientes heterogéneos. Pelo que a utilização de um Enterprise Service Bus como meio preferencial de comunicação entre os diversos componentes que constituem a solução permitiu dar resposta à integração de diversos sistemas e aos requisitos de fiabilidade definidos para a entrega de mensagens.

Com a utilização de um sistema de visão computacional externo para a avaliação das fotografias dos animais constantes nos anúncios, juntamente com o conteúdo dos mesmos, pretendeu-se fazer corresponder os anúncios de forma automática e fornecer *feedback* aos cidadãos de forma mais célere e eficiente.

Palavras-chave: Google Cloud Vision API, ESB, .NET Core

Abstract

It is estimated that there are about 6.7 million pet animals in Portugal based on recent studies, being these distributed for about 2 million houses, which represents around 54% of the Portuguese households. The curiosity, the search for food or the periodic state of sexual excitement in the female of most mammals are some examples of reasons leading to disappearance of animal species every year. This is something that causes pain and despair to the animal keepers, given that they often consider these animals as family.

In this digital era, many people use the internet looking for solutions to their problems, thus, in such cases they frequently use the social media – as well as other platforms – to share and get help. The existing platforms focus much on posting advertisements and less on applying technology that can effectively support the citizens in a more efficient manner.

The creation of an online platform, which would allow posting free ads for missing and found animals, together with the use of external components that could help the recovery rate in lost animals, was one of the main goals of this project. Including these components, with different forms of communication would demand for an integration solution, which would allow an interoperability between heterogeneous environments. Thus, using an Enterprise Service Bus as a way of flowing information between the several components that constitute the solution, it allowed integrating the several systems and the reliability requirements defined for the messages delivery.

With the use of an external computer vision system for the assessment of the photographs of animals present in the advertisements, together with the content of the ads, the aim was both to get a match in an automatic manner, as well as to give feedback to the citizens in a faster and more efficient manner.

Keywords: Google Cloud Vision API, ESB, .NET Core

Agradecimentos

Cada ser é único, distinto e concreto. Desde o seu nascimento absorve, com todos os seus sentidos, informação que lhe é transmitida e que utiliza, na maior parte dos casos, com o intuito de se tornar um ser melhor. É com esse pensamento que vivo todos os dias, tentando sempre dar o devido valor às pessoas que me ajudam a dar um passo em frente e àquelas que me ajudam a dar um passo atrás para poder dar dois em frente.

Muito agradeço aos meus pais por me terem educado, incutido valores e feito tudo o que podiam, e especialmente o que não podiam, pela minha educação e pelo meu crescimento enquanto pessoa.

Aquando da execução de um trabalho, no decurso do mestrado, o meu amigo Ricardo Costa recordou-me de um conhecido provérbio africano: "Se quiseres chegar depressa vai sozinho, se quiseres chegar longe vai acompanhado". Aprendi então, a dar ainda mais valor ao grupo que me acompanhou e a quem de seguida agradeço.

Aos bravos e corajosos companheiros de curso que tive o prazer de conhecer e acompanhar durante os dois anos do mestrado. Ao núcleo que se formou, mesmo de diferentes ramos de especialidade e que partilharam as suas experiências e conhecimentos sem querer nada em troca. O meu mais sincero obrigado à Ana Almeida, ao Mário Leite, ao Nuno Lima, ao Ricardo Costa e à Viviana Baptista.

Agradeço à Anabela Castro pela pessoa que é, dedicando-se a todos os projetos em que acredita, pela sua entrega e apoio incondicional a este projeto o meu muito obrigado.

Ao meu professor e orientador António Rocha por toda a atenção, tempo dedicado e disponibilidade demonstrada durante todo o projeto. Por todos os conselhos e direções dadas, mas também por acreditar em mim e neste trabalho.

Agradeço ainda à minha equipa de trabalho e em especial à minha chefe, Gabriela Magalhães, pela compreensão e pelo tempo que não pude estar presente em prol da minha formação.

Agradeço à minha namorada, Sofia Alves, pela compreensão, força e coragem que me deu para percorrer este tão desafiante caminho. As suas palavras e gestos de apoio muito contribuíram para este trabalho, pelo que também lhe estou profundamente grato.

Como ouvi de um peregrino, nos caminhos de Santiago: "O caminho faz-se caminhando". Aos meus amigos, àqueles que mencionei e também a muitos outros que não mencionei e que foram preciosos neste caminho, o meu mais sincero obrigado por terem feito parte dele.

Índice

Lista de Figuras

Lista de Tabelas

Acrónimos e Símbolos

Lista de Acrónimos

AHP	Analytic Hierarchy Process
API	Application Programming Interface
BPMN	Business Process Model and Notation
BSD	Berkeley Software Distribution
CD	Continuous Delivery
CI	Continuous Integration
CRO	Centro de Recolha Oficial
CSP	Content Security Policy
CSS	Cascading Style Sheets
DGAV	Direção-Geral de Alimentação e Veterinária
EF	Entity Framework
EIP	Enterprise Integration Patterns
ESB	Enterprise Service Bus
GPS	Global Positioning System
GSM	Global System for Mobile Communications
HTTP	Hypertext Transfer Protocol
ICAM	International Companion Animal Management Coalition
JSON	JavaScript Object Notation.
MIME	Multipurpose Internet Mail Extensions
ML	Machine Learning
NCD	New concept development
OCR	Optical Character Recognition (OCR)
OMV	Ordem dos Médicos Veterinários

ORM	Object-relational mapper
OWASP	Open Web Application Security Project
POCO	Plain Old CLR Object
REST	Representational State Transfer
SGBD	Sistema de Gestão de Base de Dados
SIRA	Sistema de Identificação e Recuperação Animal
SICAFE	Sistema de Identificação de Canídeos e Felídeos
SOAP	Simple Object Access Protocol
STP	Synchronizer Token Pattern
UML	Unified Modeling Language
URI	Uniform Resource Identifier
WSPA	World Society for the Protection of Animals
XSS	Cross-site Scripting

Glossário

MIME sniffing Capacidade dos *browsers* de detetarem qual o tipo de conteúdo existente na página a visualizar

on-premises Software que é instalado e que corre no computador ou computadores do cliente final

stakeholder Parte interessada ou interveniente

1 Introdução

Desde muito cedo que o Homem olhou para os animais como sendo uma fonte diversificada de recursos, tendo recorrido à sua domesticação para seu próprio benefício. Sabe-se hoje que o primeiro animal a ser domesticado terá sido o cão, há cerca de 13.000 – 17.000 anos, que se mostrava especialmente útil como vigia (avisando da presença de outros animais), na caça ou ainda como ajudante no processo de domesticação de outras espécies (Driscoll et al. 2009).

Mais tarde, na transição entre o Homem nómada, essencialmente caçador-recolector, para o período em que se torna sedentário – construindo aldeias e começando a cultivar sementes e plantas – surge a domesticação de outros animais, tais como: as ovelhas, as cabras e o gado. Isso deveu-se essencialmente aos recursos que forneciam, nomeadamente a carne, a lã, o couro ou o leite (Driscoll et al. 2009; Crabtree 1992). Esta evolução veio então trazer ao cão outro papel importante ajudando na pastorícia.

Os gatos por outro lado surgem mais tarde, há cerca de 10.000 anos, quando o Homem já havia construído as suas habitações, quintas e povoações, tornando-se um dos principais aliados no extermínio de roedores. No entanto, a origem da domesticação dos gatos não é de modo algum fácil de explicar, devido à improbabilidade deste acontecimento. O facto de os gatos selvagens serem exclusivamente carnívoros, demasiado solitários ou o facto de serem defensores de um determinado território faziam dele um candidato pouco provável à domesticação por parte do Homem. Contudo, defende-se que os gatos foram sendo tolerados pelos humanos que frequentemente os tinham por perto, tendo evoluído ao longo de milhares de anos desde os seus antecessores selvagens (Driscoll et al. 2009).

Nos dias de hoje, os cães, pelas suas características, continuam a ter papéis funcionais na sociedade sendo utilizados ainda para fins militares, caça, policiais ou de segurança pública, mas também como cães-guia ou auxiliares de terapia. Contudo, cada vez mais, cães e gatos desempenham outras funções que não as anteriormente referidas, sendo em grande parte vistos como animais de companhia vivendo dentro das casas de muitos dos portugueses.

A ligação emocional para com estes animais é hoje em dia de tal ordem que muitas famílias os consideram parte dela, e que no caso de desaparecimento o mesmo é sentido com grande dor, desespero e sofrimento. Urge, então, uma resposta mais eficaz dos agentes ligados à proteção do bem-estar animal para com estas situações. O aparecimento de ferramentas tecnológicas que possam apoiar as famílias nestas situações é de extrema importância.

Neste capítulo introduzir-se-á o contexto em que o problema se encontra, o problema, quais os objetivos que se pretende que sejam atingidos e por fim quais os contributos deste projeto para os seus *stakeholders*.

1.1 Contexto

Um estudo recente (GFK 2015) veio divulgar dados importantes relativamente ao panorama nacional, estimando a existência de cerca de 6,7 milhões de animais de companhia, estando estes distribuídos por cerca de 2 milhões de habitações, o que representa cerca de 54% dos lares portugueses. Este estudo vem ainda constatar que já existem mais cães e gatos do que crianças no seio das famílias portuguesas, sendo considerados na maioria em grande parte dos casos parte integrante das famílias, o que demonstra uma forte ligação emocional criada com os animais.

Tal como se pode verificar na Figura 1, entre 2011 e 2015, houve inclusive um aumento do tipo de ligação emocional, tanto para com os cães como para com os gatos. As ligações do tipo funcional, embora continuem a existir, têm, contudo, uma menor expressão.

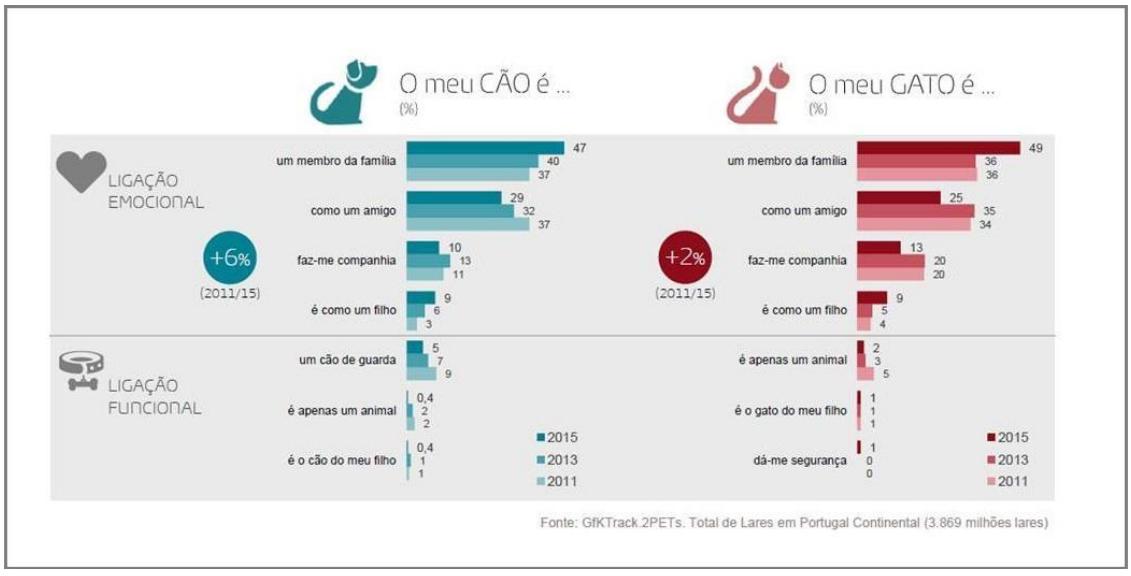

Figura 1 – Ligação emocional e funcional de cães e gatos [1]

[1] Fonte: http://www.gfk.com/pt/insights/press-release/portugal-e-um-pais-pet-friendly/

Em todo o país existem diversos agentes associados à proteção e bem-estar animal, entre eles diversas associações, clínicas e hospitais veterinários, abrigos, centros de recolha oficial (CRO), entre outros.

Felizmente, muito se tem feito em Portugal para uma maior consciencialização da importância e valorização da vida dos animais, tendo surgido nos últimos anos alguma legislação nesse sentido. De realçar a Lei n.º 69/2014 de 29 de agosto onde se estabelece a criminalização contra abandono e maus-tratos a animais de companhia, a Lei n.º 27/2016 de 23 de agosto que vem estabelecer a proibição do abate compulsivo de animais como meio de controlo da população e mais recentemente a Lei n.º 8/2017 de 3 de março que altera o código civil, onde os animais deixam de ser considerados "coisas" para terem uma classificação idêntica à atribuída às crianças.

Apesar dos avanços, há ainda fatores que contribuem para a não responsabilização dos detentores dos animais, essencialmente pelo facto de estes não possuírem qualquer meio de identificação, como coleiras, medalhas, marcas, tatuagens ou *microchip*. Embora a aplicação deste último seja obrigatória a cães com mais de 6 meses considerados perigosos ou potencialmente perigosos, a cães de caça ou cães em exposição para fins comerciais ou lucrativos desde 1 de Julho de 2004 e para todos os outros cães após 1 de Julho de 2008 (Assembleia da República 2003a) sabe-se que nem todos os detentores cumprem a lei.

O mesmo Decreto-Lei define ainda que também para os gatos seria obrigatória a aplicação de *microchip* como meio de identificação, no entanto, é remetida para um despacho posterior que nunca terá sido elaborado.

1.2 Problema

Verifica-se na sociedade atual uma reduzida empatia relativamente ao direito a uma vida digna de todos os seres sencientes, sendo ainda comum assistir-se a situações de risco, abandono, negligência e maus-tratos, apesar de toda a legislação em vigor.

No âmbito da atuação das associações de proteção animal, existem diversas problemáticas associadas que se pretendem ver minoradas ou resolvidas, contudo, tentar-se-á dar respostas a essas problemáticas de forma gradual.

Um dos problemas mais comuns é o desaparecimento de animais, que por razões ligadas essencialmente à curiosidade, mas também à busca de alimento, ou até a estados de recetividade sexual como o cio fogem ou perdem-se. Isto é causado pelo descuido por parte dos detentores, que ao deixarem uma porta ou uma janela aberta despertam a curiosidade dos animais. É ainda do conhecimento público a existência de casos em que os detentores dão liberdade aos seus animais permitindo que andem à solta, sem grandes limitações, pelo que nestes casos também é comum o seu desaparecimento.

Colocados nestas situações, e em desespero, os detentores são levados a procurar todas as ferramentas que possam ser úteis na recuperação dos seus animais de companhia. Sendo comum o pedido de apoio a clínicas veterinárias, aos CRO, a publicação de anúncios em papel espalhados pela zona de residência ou colocação de anúncios nas redes sociais e plataformas dedicadas ao auxílio em caso de perda de animais.

Por outro lado, muito devido à falta de meios de identificação visual, quem encontra estes animais opta muitas vezes por levá-los a clínicas veterinárias para obtenção dos dados dos seus detentores na esperança de esses possuírem *microchip* de identificação. Acontece, porém, que apenas cerca de um milhão de animais se encontra registado na plataforma Sistema de Identificação e Recuperação Animal (SIRA) do Sindicato Nacional de Médicos Veterinários (SNMV) (SNMV 2016a). Quanto ao Sistema de Identificação de Canídeos e Felídeos (SICAFE) não se conhecem os números, o que significa que provavelmente poder-se-á estimar que cerca de 5 milhões de animais não terão *microchip* de identificação, tornando o processo de identificação dos detentores difícil.

Nestes casos, as pessoas que encontram ou acolhem estes animais optam então por utilizar os meios ao alcance para encontrar os detentores dos mesmos. Recorrem habitualmente a meios semelhantes aos utilizados pelos detentores de animais desaparecidos, nomeadamente as publicações de anúncios em papel, nas redes sociais e diversas plataformas online.

No entanto, nem sempre se consegue fazer corresponder um animal perdido com um encontrado, fazendo com que estes fiquem na rua, entregues em abrigos ou nos CRO que acabam por os conduzir a processos de adoção.

Sente-se então uma falta de mecanismos mais eficazes no processo de estabelecimento de correspondências entre os animais perdidos e encontrados, bem como, a falta da divulgação de informações úteis para quem se encontre nas situações referidas.

Outro dos problemas existentes é a falta de elementos estatísticos que possam apoiar na tomada de consciência, sensibilização e atuação para estes casos. Por exemplo, embora exista uma perceção de um maior número de abandonos de animais em alturas do ano como o verão, não existem ferramentas que o quantifiquem.

1.3 Objetivos

Um dos objetivos claros deste trabalho é a construção de um sistema de informação que auxilie os cidadãos detentores de animais de companhia, na sua recuperação em casos de desaparecimento por perda ou fuga.

Este sistema deverá ser disponibilizado publicamente sob a forma de um *website* e/ou aplicação *mobile*, que através da disponibilização de um serviço de inserção de anúncios de animais desaparecidos e encontrados possa fazer corresponder de forma automática os casos. Para o

efeito, pretende-se fazer uso de um sistema de *Machine Learning* que analise as fotografias enviadas pelos anunciantes e que enriqueça a informação fornecida nos anúncios.

Ambas as informações serão então unidas e cruzadas de forma a tentar fazer-se corresponder os anúncios de animais encontrados com os anúncios de animais perdidos.

Através da aplicação destes automatismos espera-se produzir uma ferramenta mais eficaz e mais eficiente do que as existentes.

Pretende-se que esta ferramenta seja um apoio para os cidadãos, não só no meio digital como fora dele. Nesse sentido é objetivo deste projeto auxiliar o cidadão com o recurso a sistemas de geolocalização, marcando num mapa alguns pontos de interesse, tais como: clínicas veterinárias, hospitais e CRO onde o cidadão poderá procurar ajuda suplementar.

Sendo óbvio que tudo se passa no meio ambiente, é importante também que haja uma forma de comunicação nesse meio, pelo que a impressão de folhetos com imagens dos animais desaparecidos/encontrados será uma ferramenta muito útil a disponibilizar pela solução.

Como objetivos secundários pretende-se obter um conjunto de indicadores importantes para a atuação das associações de proteção animal em Portugal, tais como:

- Taxa de animais abandonados por altura do ano;
- Estatísticas sobre animais feridos e atropelados;
- Animais não esterilizados com acesso à via pública.

1.4 Contributos

Com a elaboração desta dissertação pretende-se apoiar o processo de recuperação de animais desaparecidos, fazendo-os retornar o mais rapidamente possível às suas casas. Para isso, pretende-se construir um sistema de informação de cariz social que possa receber anúncios sobre o desaparecimento e sobre o aparecimento de animais, e que com o auxílio das fotografias enviadas, possa efetuar a correspondência entre os dois.

Através da informação submetida pelo cidadão e da aplicação de algoritmos de visão computacional às fotografias enviadas, pretende-se auxiliar o preenchimento de informação bem como o seu enriquecimento, facilitando o processo de correspondência de anúncios, tornando-o assim mais célere, mais eficaz e mais eficiente do que os existentes.

A ligação deste projeto a outros, que funcionarão em paralelo, permitirá ainda a utilização de uma base de dados partilhada para apoiar o processo de informação aos intervenientes, fornecendo informação orientada para cada caso concreto, tal como as entidades geograficamente próximas ou o modo de agir em cada situação.

Com isto, pretende-se evitar que os animais fiquem muito tempo na rua, onde estão sujeitos aos elementos da natureza, aos ataques de outros animais, a causarem acidentes de viação, mas também a causarem distúrbios ou a fazerem aumentar a população da espécie.

2 Contexto

"*A problem well stated is a problem half-solved.*" – Charles Kettering

O projeto descrito nessa dissertação tem o propósito de ser incluído numa plataforma central de gestão de vários dados da vida dos animais de companhia. Nesta, pretende-se que haja interação de diversas entidades da sociedade ligadas à proteção animal, nomeadamente: associações, abrigos, médicos veterinários, famílias de acolhimento temporário e o próprio cidadão detentor de animais. Com a integração destas entidades do panorama nacional pretende-se poder vir a atuar em conformidade em função dos problemas detetados no âmbito desta plataforma. Este capítulo tem como objetivo contextualizar o problema proposto para resolução, de forma clara e inteligível, de modo a que o leitor consiga percecionar o enquadramento na plataforma referida.

2.1 Descrição do problema

Nas próximas secções é descrito o problema do qual nasceu a necessidade da realização deste projeto. Para uma melhor perceção por parte do leitor de todos os aspetos do problema foi utilizada uma metodologia de raciocínio crítico, abordada no livro *The Thinker's Guide to Engineering Reasoning* (Paul et al. 2007).

2.1.1 Propósito

Por todo o mundo há registo do desaparecimento de animais domésticos. É comum que uma porta ou janela deixada aberta faça com que um animal doméstico fuja ou se perca. São várias as razões para estes desaparecimentos, estando muitas vezes associados às épocas de acasalamento, do cio nas fêmeas, mas também por desorientação ou mesmo até por roubo. É pretendido com este projeto a disponibilização de uma plataforma informática que faculte um serviço de apoio à recuperação de animais desaparecidos.

Propósito do *design*

O propósito essencial deste projeto é a criação de uma ferramenta que auxilie detentores de animais de estimação a encontrá-los no caso de estes se perderem no meio ambiente. Sendo também intuito a recolha de indicadores estatísticos para um sistema centralizado, onde outros subprojetos surgirão associados, e onde os indicadores serão analisados em conjunto.

Oportunidades de mercado

De acordo com o Decreto-Lei 315/2003 compete às câmaras municipais nacionais a recolha, a captura e o abate compulsivo de animais de companhia sempre que indispensável, quer por razões de saúde pública, quer por razões de segurança, tranquilidade das pessoas e ainda pela segurança de bens (Assembleia da República 2003b). Estes animais são então dirigidos para CRO de animais – anteriormente denominados canis e gatis – onde são vistos e no caso da não reclamação encaminhados para adoção ou abate.

Com a aprovação da Lei 27/2016, que estabelece a proibição do abate de animais errantes como meio de controlo da população (Assembleia da República 2016), é esperado o aumento do número destes animais por todo o país, pelo que se pretende adicionar ao mercado uma ferramenta que auxilie o cidadão detentor de um animal de estimação a encontrá-lo no caso do seu desaparecimento.

Clientes

Os potenciais clientes desta plataforma serão em primeiro lugar, todos os cidadãos que sejam detentores de animais de estimação, e que, por motivo de desaparecimento se vejam em situação de procura do seu animal. Por outro lado, os cidadãos que encontrem animais e que procurem a plataforma para tentarem encontrar o seu detentor.

Satisfação dos requisitos do cliente

Com a utilização de dados de diferentes fontes e com a correspondência automática de características dos animais pretende-se que, no menor tempo possível, o cidadão tenha notícias sobre o estado e localização do seu animal.

Definição de valor para o cliente

Sabe-se que os animais de companhia, vistos num ambiente desconhecido ficam desorientados sendo comum causarem distúrbios, serem magoados por outros animais ou até mesmo serem atropelados. Pelo facto de se tratarem de animais de companhia de elevada estima para o cidadão é de extrema importância que, em caso de perda ou fuga, rapidamente seja possível localizar o animal.

As ferramentas existentes em Portugal não fazem a correspondência automática de características dos animais pelo que muitas vezes há informação duplicada entre os animais que estão identificados como perdidos e os que são identificados como encontrados. Tal como

referido anteriormente, a solução aqui desenvolvida será integrada numa plataforma central onde existirá informação que poderá ser útil para os clientes deste projeto. É exemplo disso a informação baseada na localização geográfica de clínicas veterinárias, CRO, associações de proteção animal ou outros para onde os animais são muitas vezes levados ou recolhidos.

Necessidade de novas tecnologias

Em vez de se fazer uma simples correspondência de características entre os animais perdidos e encontrados pretende-se que as imagens dos animais, submetidas para a plataforma, sejam analisadas por uma ferramenta automática que possa identificar mais informação além da introduzida pelo cidadão, como por exemplo: espécie ou raça. Em vez de se criar uma ferramenta nova, recorrer-se-á a serviços já existentes que o fazem devolvendo o resultado dessa análise de forma estruturada para a plataforma em causa.

Adaptação de projetos existentes

Existe no nosso país uma solução amplamente utilizada com fins semelhantes aos propostos por este projeto, no entanto para que fosse possível adaptar a solução à plataforma existente teria a associação, que solicitou este projeto, que abdicar do controlo da informação. Não é também intuito da associação ter o mesmo modo de atuação para com o cidadão pelo que não se coloca então a possibilidade de se associar à plataforma em questão.

Importância do *time-to-market*

No imediato, esta plataforma pretende ajudar a solucionar o problema associado ao desaparecimento por perda, fuga ou outras situações, de animais que acabam por chegar aos CRO e são adotados ou dados para abate num curto período de tempo.

2.1.2 Ponto de vista pessoal do problema

Pretende-se que o sistema a desenvolver possa fazer corresponder os casos de animais encontrados com os casos de animais desaparecidos de forma automática. Para isso, espera-se que através da informação fornecida pelo cidadão, conciliada com a informação oriunda de ferramentas de visão computacional, sejam reconhecidas características dos animais que possam fazer corresponder os casos. Além desta capacidade essencial à realização deste projeto, o sistema deverá ainda possuir um conjunto de características que são de seguida enumeradas.

- Através da integração com as redes sociais, permitir a autenticação no sistema, bem como a partilha dos casos;
- Auxiliar o cidadão com recurso a sistemas de geolocalização, marcando em mapas pontos de interesse, como clínicas veterinárias, hospitais, CRO ou outros;
- Devido ao cariz social do projeto, e pelo facto de se pretender financiar também através de doações espontâneas, seria de interesse ter integração com um serviço como o PayPal de modo a tornar este processo mais cómodo.

Ponto de vista pessoal dos vendedores/fornecedores

Os fornecedores das API e outros serviços de *machine learning* têm interesse no aparecimento de produtos semelhantes ao aqui descrito, essencialmente por duas razões: a económica, por se tratar de um serviço pago a partir de um número de pedidos e a relacionada com a própria imagem que passam para o exterior de um maior número de projetos a utilizarem as suas ferramentas.

Ponto de vista do cidadão

O cidadão comum irá utilizar a plataforma essencialmente em duas situações: no desaparecimento do seu animal de companhia ou no caso de encontrar um animal abandonado. Em ambos os casos o que o utilizador pretende do sistema é obter informação orientada para a sua situação num menor número de interações possível.

2.1.3 Pressupostos

Nesta subsecção são descritos os pressupostos assumidos nas diversas áreas às quais o projeto estará ligado.

Pressupostos ambientais e condições de operação

Um dos pressupostos de atuação da plataforma é o seu campo de ação, prevendo-se a sua utilização apenas no território continental nacional e ilhas.

Animais como aves, répteis ou roedores, apesar de serem animais de companhia menos comuns, ainda assim também desaparecem. É então de esperar que procurem a plataforma a para a colocação de anúncios do seu desaparecimento. Numa fase inicial do projeto estes serão excluídos pelo que serão considerados apenas cães e gatos.

A plataforma, bem como a respetiva base de dados serão instalados num *datacenter* nacional e serão feitos *backups* descentralizados de forma a ser possível recuperar de possíveis desastres.

Nível de aceitação dos pressupostos

Os pressupostos assumidos no decorrer deste projeto foram validados pelos seus mentores, sendo considerados aceitáveis e exequíveis.

Nível de maturidade das tecnologias emergentes

A evolução da tecnologia é uma constante, sendo frequente o surgimento de novas soluções que vêm complementar outras ou até mesmos substituí-las. Pretende-se que neste projeto sejam utilizadas tecnologias amplamente aceites e com um estado de maturação e previsão de escalabilidade também aceitáveis.

Pressupostos assumidos para uma solução ótima

Para uma solução ótima deveriam ser reunidas um conjunto de características desejáveis, como:

- Software tolerante a falhas;
- Reduzido *downtime*;
- Taxas de *upload*/*download* suficientes para um *delay* reduzido;
- Redundância de servidores;
- Utilização de *backups* descentralizados.

Pressupostos na disponibilidade dos sistemas

A dependência da plataforma de sistemas externos é evidente, no entanto, pressupõe-se a utilização de sistemas ou ferramentas com um histórico de elevada fiabilidade e disponibilidade.

Pressupostos das competências técnicas

As competências técnicas necessárias à elaboração deste projeto foram sendo adquiridas ao longo dos últimos anos, tanto a nível profissional como a nível académico. No entanto, assume-se que existirá ainda um conjunto de competências que terão de ser adquiridas ao longo da elaboração deste projeto.

O cidadão, utilizador do sistema informático a desenvolver, terá de possuir competências básicas de utilização de um computador ou *smartphone*, tais como a utilização de um *browser* e familiarização com partilha de conteúdos na internet.

2.1.4 Informação de Engenharia

As subsecções seguintes têm o propósito de identificar quais as fontes de informação a serem utilizadas, bem como identificar a sua utilidade e validade.

Fontes de informação

Para a realização deste projeto serão tidas em conta diversas fontes de informação para diferentes fins. No caso de aspetos legais será considerada a consulta do Diário da República onde se pode aceder à legislação em vigor. Recorrer-se-á também a documentação técnica fornecida pelos diversos fornecedores de serviços online de reconhecimento de imagem, tais como manuais, *how-to* entre outros. Sempre que se justifique serão ainda utilizados manuais de ajuda de ferramentas ou tecnologias de desenvolvimento utilizadas na solução. Materiais disponibilizados no âmbito da lecionação do mestrado serão também tidos em consideração devido à sua relevância neste projeto.

Informação em falta

Terão de ser consideradas diferentes tecnologias de análise de imagens, pelo que será necessário avaliar quais as que se adequam melhor e que são mais eficazes e/ou eficientes na avaliação.

Obtenção da informação em falta

Para obter esta informação será necessário proceder ao estudo dos serviços recorrendo a análises estatísticas para verificar qual das ferramentas será a mais indicada para o projeto em questão.

Simulação

A simulação da operação da plataforma terá de ser executada num ambiente diferente do ambiente de produção[2], ou seja, num ambiente de desenvolvimento[3], onde poderão ser submetidos pedidos de procura de animais encontrados ou de animais encontrados. A plataforma terá de analisar os dados e imagens de forma a poder dar *feedback* ao programador sobre a conformidade da solução.

Testes de componentes

A aplicação de boas práticas no desenvolvimento de *software* passa também por testar as soluções desenvolvidas. Nesse sentido serão efetuados testes, tanto aos componentes que fazem parte da solução, como também aos componentes externos para validar a sua funcionalidade.

Experiências a realizar

Um dos pontos críticos da aplicação desta solução é a dependência existente das API externas fornecidas por terceiros. Nesse sentido espera-se produzir uma experiência com cada um dos serviços externos de forma a validar a sua eficácia e eficiência na avaliação de fotografias.

Por forma a que a solução não fique dependente de apenas uma ferramenta é de elevada importância que através destas experiências se selecionem as duas melhores ferramentas.

Estudos, soluções anteriores e problemas

A necessidade do estudo de soluções anteriores prende-se com o facto de ser uma boa prática aprender-se com os erros do passado. Nesse sentido, necessitar-se-á de fazer uma abordagem crítica às soluções existentes, com o intuito de perceber as razões da existência de determinadas características dos sistemas. Em termos de construção da aplicação há uma especial preocupação com a validação dos dados e fotografias enviadas pelos utilizadores.

[2] Ambiente de produção: Ambiente onde os utilizadores terão acesso ao produto final
[3] Ambiente de desenvolvimento: Ambiente isolado onde se desenvolve um determinado produto

Sabe-se que outras plataformas, inclusive de outro tipo de atividades, possuem processos manuais de moderação que são executados para garantir que apenas anúncios com credibilidade sejam aceites.

A tentativa de exploração do sistema através do preenchimento de formulário através de *bots*[4] deve também ser considerada devendo aplicar-se mecanismos que o evitem, como por exemplo a utilização do serviço reCAPTCHA.

A submissão de imagens para um sistema acarreta também um nível de preocupação considerável, sendo que deve ser considerada a moderação de imagens inseridas pelos utilizadores. Imagens com conteúdos impróprios, como zoofilia, violência contra animais ou outras de cariz semelhante devem ser completamente vedadas de publicação. A utilização de funcionalidades de reconhecimento de conteúdo impróprio nas imagens, deverá ser ponderada, passando depois a um sistema de aprovação de anúncios caso necessário.

Disponibilidade da informação

A informação necessária à elaboração da solução para o problema proposto passa por documentos legislativos, técnicos e científicos estando disponíveis na internet, na sua maioria, de forma gratuita.

Necessidade de informação

Essencialmente será necessária a consulta de documentação sobre protocolos de comunicação com as API externas de forma a possibilitar a integração dos sistemas.

Melhor forma para recolher a informação

A informação necessária ao desenvolvimento da solução é encontrada sob a forma de artigos científicos, estando acessível através de ligação do Instituto Superior de Engenharia do Porto. Outros materiais e documentação técnica está disponível *online* pelo que basta ter um acesso à Internet.

2.1.5 Conceitos e teorias

Nas subsecções seguintes são apresentados os conceitos e teorias relevantes para a definição do problema.

Conceitos e teorias aplicáveis ao problema

Embora não se trate de um sistema crítico, deve o desenvolvimento desta solução basear-se na utilização de processos e programação confiável sendo sempre que possível tolerante a

[4] Programas criados para, de forma automática, preencher e submeter formulários em *websites*

algumas falhas. A utilização da modularidade dos componentes que vão constar da solução final é um dos conceitos base para esta solução.

A utilização de processos que conduzam à segurança dos dados transmitidos é também de especial importância pois, além dos dados dos animais, vão ser guardados dados dos seus respetivos detentores. Dados esses que não deverão ser comprometidos, devendo ser assegurada a sua autenticidade, integridade e confidencialidade, tanto durante as comunicações como no seu armazenamento.

Existência de concorrência

A existência de concorrência é um fator que não foi descurado na abordagem ao problema, sendo esta referida na secção 2.3.4 do estado da arte.

Tecnologias e teorias apropriáveis

A utilização de *frameworks* recentes permite a adaptação à realidade dos dias de hoje com a proliferação de equipamentos tão distintos como *smartphones, tablets, smart tvs* ou os convencionais computadores e portáteis. Hoje em dia não é apenas no computador pessoal que se acede à Internet, mas sim num infindável número de dispositivos. É então importante que sejam utilizadas tecnologias capazes de chegar a esses equipamentos.

Na construção de aplicações *web*, por exemplo, é hoje em dia possível através da utilização de *frameworks* de *front-end* como o Bootstrap ou Foundation chegar a esses equipamentos. Uma das teorias que assenta na utilização destas *frameworks* é que se deve em primeiro lugar construir as aplicações *web* para dispositivos com ecrã menor e só depois para ecrãs maiores; a essa forma de proceder dá-se o nome de *Mobile-First*.

Recorrer-se à interligação de componentes de forma menos agarrada (acoplada), mais sobre a forma da utilização de serviços ao invés de utilização de sistemas monolíticos é também uma teoria que deve ser tomada em consideração para facilitar, entre outras razões, a manutenção, a reutilização e a dependência entre componentes.

Tecnologias emergentes

Há alguma recetividade, embora cautelosa, à adoção de algumas tecnologias recentes no mercado. A título de exemplo a *framework* .NET Core da Microsoft será tida em consideração para análise. Esta *framework* é uma evolução da .NET Framework, tendo evoluído para uma estrutura *open-source* sendo possível a sua utilização em múltiplas plataformas.

2.1.6 Implicações

É normal, durante o processo de aplicação de algum tipo de solução de engenharia, surgir alguma consequência ou implicação, nesse sentido são, nas subsecções seguintes, enumeradas as que foram identificadas.

Implicações da informação recolhida

Os dados recolhidos pela plataforma deverão ser analisados automaticamente pela ferramenta de modo a que o processo de identificação dos animais possa decorrer normalmente. Deverá, contudo, existir um tratamento cuidado da informação, de forma a que dados sensíveis em nenhum momento possam ser divulgados. A informação deverá ser guardada na base de dados da solução e deverão ser implementados mecanismos restritivos para acesso à mesma.

Implicações mais importantes da tecnologia no mercado

O aparecimento de novas ferramentas, tecnologicamente mais avançadas, trará ao mercado algum dinamismo pelo que é de esperar que produtos concorrentes acompanhem a tendência ou que acabem por perder utilizadores.

Implicações das tecnologias não atingirem um nível de maturidade aceitável

A constante evolução tecnológica empurra muitas vezes o mercado para a adoção de tecnologias nem sempre no estado de maturidade que seria de esperar. No âmbito do desenvolvimento da plataforma referida, recorrer-se-á preferencialmente a ferramentas e soluções com algum histórico de fiabilidade e apenas a versões estáveis de *software*.

Importância da sustentabilidade no mercado

A sustentabilidade é uma questão importante no âmbito em que o projeto se insere, pelo que foram pensadas algumas formas de a atingir. Os custos relacionados com o alojamento da plataforma, eventuais ferramentas de *Machine Learning,* certificados digitais e comunicações terão de ser cobertos, numa primeira fase, através de venda de *merchandising* e solicitação de doações.

Possíveis melhorias

A tecnologia está em constante atualização, pelo que é previsível que possam vir a ser implementadas melhorias à medida que forem surgindo técnicas ou métodos mais recentes. Uma das melhorias previstas é a criação de uma aplicação *Mobile* que permita mais agilidade no processo de inserção de anúncios, uma vez que as fotografias são, hoje em dia, em grande parte, tiradas através de *smartphones*.

Considerações do ciclo de vida

A evolução constante no mundo das tecnologias acarreta também a constante obsolescência dos produtos existentes. Nesse sentido tanto os componentes de *hardware* como os de *software* vão sendo ultrapassados por tecnologias mais recentes, mais eficientes e mais adaptadas à presente realidade. Prevê-se, no entanto, que não seja necessário, durante um período de cerca de cinco anos, efetuar alterações ao funcionamento da plataforma.

Implicações em caso de falha do produto

As possíveis causas para falha do produto estão associadas essencialmente com falhas de *hardware* ou comunicações, no entanto poderão também surgir falhas associadas à sua construção. A indisponibilidade do serviço por razões de falha, dependendo da seriedade e tempo de recuperação e/ou reparação, poderá levar à insatisfação dos utilizadores, podendo em último caso levar à desistência de utilização do sistema.

Consequências da alteração de características essenciais

A dependência de API externas é um fator de risco, pelo que alterações ao seu funcionamento poderão afetar a execução da correspondência automática de anúncios, fazendo entrar o sistema num modo de atuação manual, ou seja, sem a utilização das características descobertas pelas API.

Funcionalidades insensíveis à mudança

De acordo com a dimensão do sistema a construir calcula-se que os protocolos de comunicação utilizados não terão de ser alterados caso sejam feitas modificações à arquitetura do sistema ou outras características. A estrutura e base de dados também não terão de ser alteradas caso existam funcionalidades que sejam modificadas.

Potenciais benefícios de subprodutos

O alargamento das funcionalidades do sistema a outro tipo de animais menos comuns, como aves ou répteis poderá ser uma mais-valia para a plataforma.

2.1.7 Conceito básico do problema

As plataformas de recuperação de animais desaparecidos, atualmente existentes no mercado, carecem de uma grande interação por parte das entidades responsáveis, estando também muito dependentes em métodos de identificação que nem sempre são aplicados, como é o caso do *microchip*.

2.1.8 Pergunta chave do problema

Do problema proposto, pretende-se ver respondida a seguinte questão: É possível tornar o processo de recuperação de animais desaparecidos mais eficaz e mais eficiente, de forma a que mais animais voltem às suas casas num menor período de tempo?

2.2 Análise de valor

Uma das metodologias utilizadas na análise de um novo produto ou serviço é a análise de valor. A aplicação desta metodologia tem como principal objetivo avaliar como aumentar o valor de um determinado produto ou serviço com o menor custo possível sem sacrificar a qualidade. No âmbito desta dissertação utilizar-se-á o modelo *New Concept Development* (NCD) de (Koen et al. 1996) para analisar o valor da solução a desenvolver.

2.2.1 Modelo NCD

O modelo NCD (Koen et al. 1996) é utilizado para representar a primeira, de três partes do processo de inovação também chamado de *Fuzzy Front End* ou *Front-End of Inovation*. Sendo o desenvolvimento de novo produto (NPD) e comercialização a segunda e terceira fases respetivamente. O NCD é composto por três componentes chave: o motor, os fatores influenciadores e os cinco elementos internos; sendo representado através de uma forma circular que pretende ilustrar a forma como os componentes estão ligados e como é possível a passagem entre cada elemento que o constitui. O motor representa os níveis superiores de gestão de uma organização que gere os cinco elementos do modelo. Os fatores influenciadores representam "o quê" ou "quem" influencia o processo de inovação, que podem ser elementos internos à organização ou elementos externos, tais como: fatores políticos, legislativos, ambientais ou outros. Por fim, os cinco elementos do modelo são a identificação da oportunidade, a análise de oportunidade, a geração e enriquecimento de ideias, a seleção de ideias e a definição do conceito.

Identificação de oportunidade

A oportunidade para a realização deste projeto surge com a perceção de uma maior tendência para a adoção de animais em Portugal, bem como com a perceção de que existe uma maior sensibilidade da sociedade para as problemáticas associadas ao bem-estar destes. As redes sociais são hoje em dia utilizadas por milhões de pessoas, sendo um ponto de partilha de informação de todo o tipo de assuntos. É notória a constante partilha de temas relacionados com maus-tratos, proteção ou perda e aparecimento de animais. Nos últimos anos, têm também aparecido diversas entidades ligadas a estas problemáticas e o surgimento de legislação focada num maior cuidado com os direitos dos animais vem então potenciar a realização de projetos nestas áreas.

Análise de oportunidade

Através da análise do mercado português, verificou-se que existe um crescimento do número de animais de companhia (GFK 2015), nomeadamente cães e gatos, e uma maior sensibilidade da sociedade para as problemáticas associadas ao bem-estar animal.

Verificou-se também que nos últimos anos a legislação portuguesa foi sofrendo diversas alterações no sentido de uma maior proteção dos direitos dos animais, como a criminalização de maus-tratos (Assembleia da República 2014) a proibição do seu abate para fins de controlo de população (Assembleia da República 2016) ou ainda a alteração do código civil para a definição de um estatuto diferenciado dos animais, até então classificados como "coisas".

A aplicação da Lei n.º 27/2016 fará alterar a filosofia de atuação do estado português para com os animais fazendo com que o foco se direcione agora para o apoio a programas de esterilização e de adoção. É esperado por isso que, nos próximos anos, o número de animais adotados aumente e como consequência aumente também a ocorrência do número de casos de animais desaparecidos das casas dos seus detentores.

Embora exista uma plataforma que se dedica ao apoio nos casos de desaparecimento e aparecimento de animais, percebe-se que o processo é pouco ágil necessitando de uma interação constante por parte do cidadão.

Geração e enriquecimento de ideias

Algumas das ideias para a execução deste projeto surgiram da realização de *brainstormings* com os intervenientes do projeto, onde foi possível identificar que o processo de correspondência entre os anúncios de animais desaparecidos e os anúncios de animais encontrados poderia ser coadjuvado através de processos mais automáticos.

Para isso, a associação teria de ser feita através do cruzamento da informação de ambos os anúncios, pelo que a sua fiabilidade teria de ser grande. Acontece, porém, que nem sempre o cidadão sabe reconhecer as características dos animais (raça, idade ou outros elementos de relevo) para a correspondência, não preenchendo os campos necessários para este processo.

Nesta reflexão surgiu então a hipótese de utilização de ferramentas de aprendizagem automática para reconhecimento de padrões em imagens que de algum modo enriquecessem a informação fornecida pelos cidadãos no preenchimento dos anúncios.

Estas ferramentas apresentam-se sob a forma de *software* construído praticamente do zero, e que se podem reutilizar através de bibliotecas, ou sob a forma de serviços *online* que podem simplesmente ser invocados. A estas ferramentas de aprendizagem automática dá-se o nome de *Machine Learning*, sendo um subcampo das ciências da computação.

Seleção de ideia

O processo de aprendizagem de um sistema de *Machine Learning* é enriquecido através da quantidade e qualidade de *inputs* que são fornecidos. Pode-se assim afirmar que um serviço *online*, por exemplo da Google, através do *input* de milhões de imagens terá uma melhor aprendizagem do que aquele que poderia ser feito através da utilização de um sistema instalado localmente.

Definição de conceito

A ligação entre os cidadãos portugueses e os animais tornou-se de tal forma emocional que cerca de 49% das pessoas detentoras de gatos, os consideram como membros da família, sendo que os cães ocupam a segunda posição com 47% dos casos (GFK 2015).

Esta ligação afetiva vem então demonstrar a importância dos animais de companhia no seio das famílias portuguesas, sendo fácil de entender o desespero sentido por uma pessoa que se vê numa situação de desaparecimento do seu animal.

Pretende-se que com a construção de uma ferramenta informática, se possa auxiliar os cidadãos nestas situações, bem como aqueles que possam encontrar animais perdidos na rua e que de alguma forma queiram ver o seu bem-estar salvaguardado, tentando fazer com que retornem às suas famílias.

O objetivo deste projeto é então a construção de uma plataforma *online* que possa ser utilizada num *browser* de um computador assim como num *smartphone*, e que beneficiando das suas características, possa ser amplamente divulgada. Para além disso pretende-se também que possa recorrer às redes sociais para a partilha de anúncios, e que por fim venha a ser a plataforma elegida pelos cidadãos para a recuperação de animais. É também intuito do projeto a integração desta ferramenta com outras de cariz semelhante, onde, através da partilha de dados, se forneça informação importante para o apoio à decisão do cidadão, como por exemplo os locais de procura mais aconselhados.

O intuito geral da ideia selecionada é recorrer a *software* de reconhecimento de padrões em imagens de forma a enriquecer a informação contida nos anúncios publicados. Desta forma conseguir-se-á reunir informação que possa ser utilizada para efetuar correspondência entre animais encontrados e desaparecidos de um modo totalmente automático.

Relativamente à sustentabilidade financeira do projeto o objetivo é que, nos vários momentos de utilização da plataforma se façam apelos a doações voluntárias de forma a fazer face aos custos da infraestrutura e serviços.

Com a utilização desta plataforma os nossos clientes beneficiarão de um recurso de elevada eficácia e eficiência, oferecendo-se desta forma um valor acrescentado ao mercado.

2.2.2 Valor, valor percecionado e valor para o cliente

Alguns conceitos são de seguida explicados para facilitar a compreensão dos conceitos ao leitor.

Valor

"*Value creation is a concept that is difficult to achieve, understand, model and/or conceptualize. Some authors consider value creation a trade-off between benefits and sacrifices perceived by customers during a supplier's offering*" (Geoff Lancaster 2000).

O valor é visto como um conceito subjetivo que está muito ligado ao indivíduo. No contexto empresarial, a sua criação é uma atividade que requer bastante esforço na medida em que se deve pesar muito bem os custos e benefícios para o cliente.

Este projeto propõe a criação de um serviço ágil, de fácil utilização e de maior eficiência do que o existente, sendo o grau de sacrifício para o cliente bastante diminuto.

Valor percecionado

"*Different customers perceive different value for the same products/services. In addition, organizations involved in the purchasing process can have different perceptions of customers' value delivery*" (Ulaga and Eggert 2006).

Tal como referido, o valor percecionado varia de acordo com cada indivíduo, sendo normal duas pessoas atribuírem diferente valor a um mesmo bem ou serviço, o que significa que também é visto de diferentes formas por quem vende e por quem compra. Pretende-se que os utilizadores do nosso sistema o vejam como uma inovação pelos seus métodos de sugestão automática de informações baseadas na localização e pelos métodos de correspondência entre os anúncios de animais encontrados com os desaparecidos.

Valor para o cliente

"*Value for the customer (VC) is any demand-side, personal perception of advantage arising out of a customer's association with an organization's offering, and can occur as reduction in sacrifice; presence of benefit (perceived as either attributes or outcomes); the resultant of any weighed combination of sacrifice and benefit; or an aggregation, over time, of any or all these*", (Woodall 2003).

O valor para o cliente poderá ser visto como a perceção de vantagem que se obtém entre os benefícios e os sacrifícios de uma determinada oferta de valor. Espera-se então, que no âmbito deste projeto, se crie um conjunto de benefícios para o cliente, e que este esteja disposto em troca a sacrificar-se, contribuindo com possíveis doações monetárias. Na Tabela 1 são apresentados os benefícios e sacrifícios para o cliente.

Tabela 1 – Benefícios e sacrifícios para o cliente

Domínio / âmbito	Serviço	Relacionamento
Benefícios	Rede de informação Agilidade Usabilidade Fiabilidade	Solidariedade Recomendação
Sacrifícios	Possíveis doações Possível aquisição de medalhas de identificação	

2.2.3 Proposta de valor

Através da disponibilização de uma plataforma online, de fácil utilização pretende-se agilizar o processo de recuperação de animais desaparecidos, conseguindo-o no menor tempo possível.

Pretende-se, com o recurso a informação orientada para o cidadão, tornar o processo simples, ágil, eficaz e mais eficiente do que o existente. Para isso ter-se-á em consideração a localização do cidadão de forma a poder facultar-se informação sobre entidades geograficamente mais próximas, tais como: médicos veterinários ou clínicas veterinárias, CRO, forças de segurança, ou outras de relevo.

A utilização de algoritmos de reconhecimento de padrões em imagens enriquecerá a informação já constante do sistema, facilitando a correspondência entre os animais desaparecidos e encontrados, tornando este processo mais automatizado.

A proteção ambiental e humana são fatores tidos em consideração, no sentido em que quanto mais cedo forem retirados os animais da rua, menos riscos existem de atropelamentos, reprodução, proliferação de doenças ou outros malefícios.

2.2.4 Modelo de negócio CANVAS

Por forma a se ter uma visão global do modo de atuação de um determinado modelo de negócio é comum recorrer-se a representações estruturadas com visões de diferentes perspetivas. Um modelo amplamente utilizado é o Modelo de Negócio CANVAS.

Este modelo está estruturado, tal como ilustrado na Figura 2, em 9 blocos: 1) Segmentos de clientes, 2) Proposta de valor, 3) Canais, 4) Relacionamento com clientes, 5) Receitas, 6) Recursos-chave, 7) Atividades-chave, 8) Parcerias-chave, 9) Estrutura de custos.

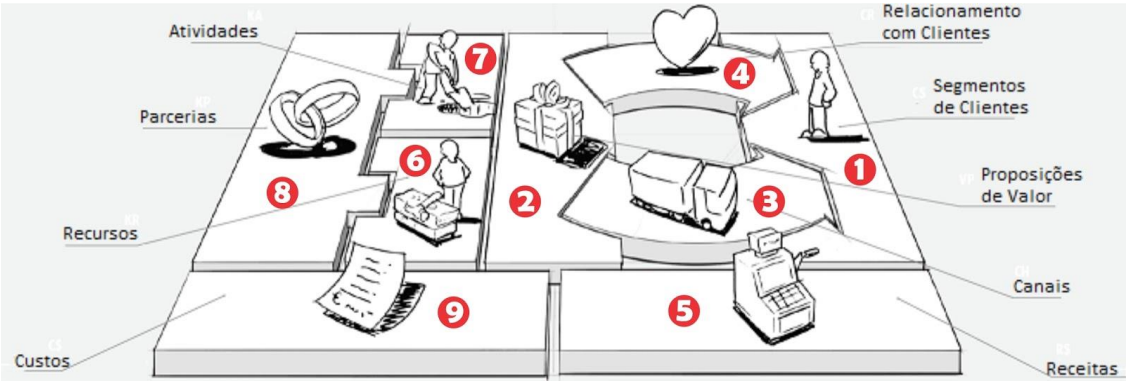

Figura 2 – Preenchimento modelo CANVAS [5]

1) Segmentos de clientes
   Neste primeiro bloco pretende-se definir quais são os grupos de pessoas ou organizações que se pretende atingir e servir, existindo diversos tipos de segmentos, como por exemplo: mercado de massas, nichos de mercado, segmentado, diversificado entre outros.
2) Proposta de valor
   Aqui é definido o valor proposto aos segmentos de clientes definidos no bloco anterior, definindo-se que problemas ou necessidades se pretendem satisfazer. Aqui são descritos que produtos fornece ou que serviços presta ao cliente, o que faz com que este prefira esta organização a outra.
3) Canais
   Neste bloco é sugerido que se definam quais os canais utilizados para a organização entregar a proposta de valor. Habitualmente canais de comunicação, distribuição e vendas.
4) Relações com os clientes
   A forma como a organização estabelece e mantém relação com cada um dos segmentos de clientes deve ser descrito neste bloco.
5) Receitas
   Que receitas são geradas de cada segmento de clientes. Deve, contudo, avaliar-se quanto é que cada segmento de clientes está disposto a pagar pelo produto ou serviço proposto.
6) Recursos-chave
   Os recursos-chave são todos os recursos necessários para dar suporte aos blocos anteriores. Isto é, todos os recursos que são utilizados para criação e oferta da proposta de valor, para levar o produto ou serviço aos mercados, para manter os relacionamentos com os clientes e por fim todos os recursos utilizados para a área de receitas. Recursos físicos, intelectuais ou humanos são alguns dos exemplos que podem ser definidos neste bloco.

[5] Fonte: (Parloff 2016)

7) Atividades-chave
   Neste bloco descrevem-se as atividades mais importantes que a organização tem de executar por forma a fazer o modelo de negócio funcionar.
8) Parcerias-chave
   No bloco de parcerias-chave devem estar definidos quais os fornecedores e parceiros principais para o modelo de negócio.
9) Estrutura de custos
   Por fim, no bloco de estrutura de custos devem ser referidos os principais custos associados ao funcionamento do modelo de negócio.

O modelo CANVAS elaborado para este projeto poderá ser consultado na Tabela 2 de seguida.

Tabela 2 – Modelo CANVAS do negócio

<table>
<tr>
<td rowspan="2">Parceiros-chave

Associações de proteção animal
Abrigos de animais
Famílias de acolhimento temporário
Médicos veterinários
Cidadão</td>
<td>Atividades-chave

Correspondência automática de animais encontrados com animais perdidos.
Correspondência automática de animais perdidos com animais encontrados.
Elaboração de relatórios estatísticos</td>
<td rowspan="2">Proposta de valor

O cidadão vai ter acesso a uma plataforma centralizada, que possui informação de diversas entidades, e que vai possibilitar encontrar rapidamente um animal de estimação perdido.
Permite poupança de tempo utilizando algoritmos de reconhecimento de padrões em imagens/fotografias.
Proteção ambiental e humana no sentido em que se pretende que rapidamente se tirem os animais da rua.
Informação de locais onde procurar os animais.</td>
<td>Relacionamento com clientes

Suporte técnico</td>
<td rowspan="2">Segmentos de clientes

Cidadão que encontra animais na rua.
Cidadão que perde um animal doméstico.
Associação Santuário Animal.
Clinicas e médicos veterinários</td>
</tr>
<tr>
<td>Recursos-chave

Servidor Web
Base de dados
Internet
Alojamento/Domínio
Know-how tecnológico</td>
<td>Canais

Plataforma
E-mail
Redes sociais</td>
</tr>
<tr>
<td colspan="3">Custos
Alojamento, Certificado digital, Domínio, API de análise de imagens.</td>
<td colspan="3">Receitas
Doações, patrocínios e fundos provenientes de financiamento para projetos.</td>
</tr>
</table>

2.3 Estado da arte – Soluções e abordagens existentes

Esta secção tem como objetivo a apresentação dos métodos de identificação de animais de companhia existentes, por forma a reunir-se quais os que poderão ser úteis para o funcionamento da plataforma a construir. São também abordadas as plataformas concorrentes e as bases de dados de registo de animais a nível nacional que poderão ser utilizadas para a pesquisa de animais por *Microchip*.

2.3.1 Métodos de identificação externa

A identificação de animais é um método utilizado desde a antiguidade até aos dias de hoje como meio de rastreamento de animais vivos. Há evidências que a identificação de animais através de marcas é utilizada desde o século XVIII a.C. estando referido no Código de Hammurabi (Blancou 2001).

Na atualidade, o processo de identificação continua a constituir um desafio à escala mundial, existindo, contudo, múltiplas formas de o minimizar. Nesse sentido, e porque é importante perceber-se de que forma a solução a apresentar irá apoiar no processo de recuperação de animais, é necessário entender que formas existem de distinção entre animais.

Também a perceção dos meios de identificação dos animais de companhia permitirá mais tarde obter automaticamente algumas características que o cidadão poderá não saber reconhecer. A título de exemplo, é útil, para o sistema a construir, saber-se que apenas 1 em cada 3000 dos gatos tricolor (*calico* e *tortoiseshell*) são machos (Hageltorn & Gustavsson 1981).

Existem três tipos de categorias de identificação de animais: as permanentes, cuja aplicação se torna efetiva e muito dificilmente é apagada ou retirada; as semipermanentes que envolvem geralmente coleiras com identificadores ou etiquetas colocadas em orelhas; e ainda os métodos temporários, como pinturas ou transmissores rádio (WSPA 2008). Os principais métodos identificados são os seguintes:

Métodos permanentes

- Tatuagem;
- *Microchip*;
- Biometria;
- Entalhe de orelha;
- *Freeze Brand.*

Métodos semipermanentes

- Coleiras;
- Etiquetas de orelha.

Métodos temporários

- Pintura/tintura;
- Rádio transmissor.

Métodos permanentes

Aos métodos de identificação que se caracterizam pela aplicação de processos, cujas características dificultam ou impossibilitam a sua remoção atribui-se a designação de permanentes. Destes, destacam-se as tatuagens e *microchips*, mas também as características biométricas dos animais.

1) Tatuagem

A utilização de tatuagens, como meio de identificação de animais, é provavelmente um dos métodos mais antigos de identificação permanente. Tendo sido aceite como meio de identificação na União Europeia para efeitos de circulação de cães, gatos e furões até 3 de julho de 2011, passando depois a ser excluída dos meios permitidos (Parlamento Europeu & Conselho da União Europeia 2013).

2) *Microchip*

Um dos métodos mais utilizados no mundo inteiro para identificação única de animais é a aplicação de *microchip*, que é inserido por injeção subcutânea.

Com a entrada em vigor do Decreto-Lei n.º 312/2003 de 17 de dezembro de 2003, passou a ser obrigatória a identificação por *microchip* a cães das raças consideradas perigosas, cães perigosos, cães que participem em atos venatórios e animais de reprodução com registo em Livro de Origem[6]. O carácter obrigatório passou a abranger todos os cães nascidos a partir de 1 de julho de 2008.

Este método poderá, no entanto, ser auxiliado por métodos externos, visíveis, que auxiliem na identificação dos detentores dos animais. O tamanho do microchip é pouco maior do que um grão de arroz, tal como se pode ver na Figura 3.

Figura 3 – *Microchip*[7] com o tamanho de um grão de arroz

[6] Registo e identificação, em Portugal, dos animais de raça canina pura.
[7] http://www.lifewithbeagle.com/2014/07/the-dos-and-donts-of-microchipping-your.html

3) **Biometria**

A leitura da retina dos animais é um método não invasivo já utilizado em alguns países para identificação de gado e animais de companhia, embora não seja amplamente adotado.

4) **Corte de ponta da orelha ou entalhe**

Estes dois métodos são utilizados habitualmente para marcar animais esterilizados que são assim marcados por programas de controlo de população. Enquanto que o método de corte da ponta da orelha é reconhecido pela maioria das organizações, o entalhe de orelha não é recomendado por poder ser facilmente confundido com ferimentos naturais (WSPA 2008).

5) ***Freeze Brand***

O método de marcação a frio é utilizado essencialmente em gado ou cavalos, sendo aplicado através de uma marca com um ferro, que anteriormente foi submetido a nitrogénio líquido. Este método caracteriza-se por, após o processo de caracterização da pele, deixar uma marca branca na pele do animal.

Métodos semipermanentes

Estes métodos são habitualmente utilizados em animais de companhia, mas também em gado. Caracteriza-se essencialmente pela utilização de coleiras ou etiquetas que identifiquem os animais para com os seus detentores. Estes são métodos que através das suas características externas poderão ser facilmente removidos, tornando o processo de identificação difícil ou praticamente impossível.

1) **Coleiras**

A utilização de coleiras é um dos métodos mais simples, baratos e ao mesmo tempo mais eficazes, sendo usado tanto por questões de segurança (aplicação de trela) como por razões relacionadas com a identificação dos animais. A aplicação de uma simples medalha com a informação básica do animal e seu detentor são cruciais para os processos de recuperação de animais.

2) **Brincos de orelha**

Os brincos de orelha são utilizados essencialmente em gado para identificação do seu proprietário ou para fazer prova de vacinação podendo também ser utilizado para identificação individual de cada animal (WSPA 2008).

Métodos temporários

Caracterizam-se por temporários os métodos cuja aplicação provoque um efeito de curta duração ou que esteja associado a aparelhos cuja autonomia é limitada.

1) **Rádio transmissores**

A aplicação de rádio transmissores é vulgarmente utilizado em animais selvagens para rastrear as deslocações dos animais, no entanto não é um método muito utilizado para monitorização de cães e gatos (WSPA 2008).

2) **Sistemas de localização**

Há já um conjunto de produtos comerciais que permitem a localização de animais através da utilização de tecnologias como *Global Positioning System* (GPS) e *Global System for Mobile Communications* (GSM). Este tipo de produtos permite em qualquer momento saber onde se encontra o animal, estando, no entanto, sujeito à utilização de baterias com tempo de vida limitado.

3) **Tinta ou corante**

A aplicação de tinta ou corante poderá ser utilizada como método de identificação temporária de cães e gatos, sendo um método de fácil aplicação e de elevada visibilidade (WSPA 2008).

2.3.2 Identificação de características físicas e comportamentais

Por observação direta é possível determinar grande parte das características físicas e comportamentais de um animal, pelo que estas deverão também ser utilizadas como meio complementar de identificação.

Existem características que poderão à partida ser evidentes para uma grande parte das pessoas, no entanto para outras poderão não ser tão óbvias. De seguida são enumeradas algumas das características físicas que poderão ser utilizadas para identificação dos animais, sem que, no entanto, tenha sido utilizado um grande rigor quanto à aplicação de terminologia do ramo da Biologia. Procurou-se a utilização de termos mais próximos do cidadão comum, de forma a que este possa identificar as características de forma mais aproximada possível.

Características físicas

- Espécie (cão, gato);
- Raça (ex. Cão – Castro Laboreiro, Gato – Europeu Comum);
- Tamanho (mini, pequeno, médio, grande, gigante);
- Idade (cria, jovem, adulto, sénior);
- Pelo (curto, médio, longo);
- Padrão Pelo (pintas, listas, atartarugado, bicolor, liso, malhado, tigrado, tricolor)
- Sexo (macho, fêmea);
- Cores (amarelo, azul, bege, branco, castanho, cinzento, dourado, laranja, preto, rosa, verde, vermelho);
- Cauda (curta, média, longa, cortada);
- Orelhas (curta, média, longa, cortada).

Características comportamentais

- Agressividade;
- Docilidade;
- Adestramento;
- Hiperatividade;
- Apatia;
- Medo.

2.3.3 Bases de dados nacionais

Devido à necessidade da existência de um registo centralizado de animais e respetivos detentores foram criadas as bases de dados SIRA e SICAFE. Embora sob a responsabilidade de entidades diferentes, estas bases de dados permitem relacionar o par animal-detentor através de mecanismos de identificação como o *microchip* ou tatuagem.

1) Sistema de Identificação e Recuperação Animal (SIRA)

O Sindicato dos Médicos Veterinários (SNMV) possui desde 1992 um registo nacional de animais de companhia, contendo na sua base de dados cerca de um milhão de registos. Em 2011, foi lançada a plataforma *online* SIRA que tem o pressuposto de "dotar os Médicos Veterinários de um novo serviço para os seus clientes" (SNMV 2016b). Esta plataforma é utilizada essencialmente por médicos veterinários, que efetuam o registo *online* na plataforma, mas também através de registos recebidos em formato próprio e que são posteriormente inseridos no sistema.

Com base neste registo, a plataforma presta também o serviço de recuperação de animais, sendo possível, para os animais registados, facilmente localizar o seu detentor.

O registo de animais desaparecidos e encontrados funciona em moldes ligeiramente diferentes dos referidos na plataforma disponibilizada em "www.encontra-me.org". No âmbito do SIRA não são publicados anúncios, mas sim o registo de um pedido de apoio. Este registo é feito através de um formulário disponibilizado pela ferramenta Google Forms. O SNMV emite então um comunicado a todas as clínicas veterinárias com os dados dos animais desaparecidos, para que em qualquer futuro contacto seja possível a estas detetarem a origem dos animais.

No entanto, o processo de animal perdido ou encontrado só pode ser aberto para os animais com Identificação Eletrónica.

2) Sistema de Identificação de Canídeos e Felídeos (SICAFE)

A criação desta plataforma tem origem no Decreto-Lei n.º 313/2003, onde se estabelece a obrigatoriedade da identificação eletrónica, ou seja, por meio de microchip, de cães e gatos. Nesse sentido surgiu então o SICAFE que se encontra à responsabilidade da Direção-Geral de

Alimentação e Veterinária (DGAV) que se encontra disponível para os médicos veterinários municipais e funcionários de Juntas de Freguesia.

O processo de inserção de registos nessa base de dados é então responsabilidade das entidades referidas anteriormente, mas o processo está dependente do cumprimento da lei por parte do cidadão.

2.3.4 Plataformas concorrentes

Existem em Portugal pelo menos duas plataformas amplamente utilizadas e divulgadas no processo de recuperação de animais desaparecidos, pelo que são analisadas quais as características que ambas possuem.

1) **Plataforma www.encontra-me.org**

Esta plataforma existe desde 2005 em Portugal, sendo amplamente divulgada e tendo como característica principal a publicação e a pesquisa de anúncios de animais desaparecidos e encontrados.

Para publicar anúncios o cidadão precisa de se registar na plataforma, através de *e-mail* e *password* que serão utilizados posteriormente no processo de autenticação. A plataforma não permite, no entanto, a utilização de autenticação baseada em *providers* externos, tais como Facebook, Google+ ou outros.

Animais desaparecidos e encontrados

Os anúncios são o meio utilizado por esta plataforma para a participação do desaparecimento e aparecimento de animais. Para tal, é necessário preencher um formulário correspondente ao caso em que o cidadão se encontra.

O formulário utilizado para preenchimento de anúncios de animais desaparecidos é composto por cinco grandes tópicos: data e local do desaparecimento, até três fotografias do animal, características do animal, comentários e contactos.

Depois da publicação do anúncio é possível ainda a partilha do anúncio nas diversas redes sociais e impressão de panfleto com a informação necessária para quem possa encontrar os animais.

Da mesma forma, os cidadãos que encontram animais podem publicar um anúncio com características semelhantes às encontradas no formulário para animais desaparecidos, no entanto, com as variações expectáveis. O campo de "local de desaparecimento" surge como "local do aparecimento" e existe ainda um grupo para especificar em que situação se encontra atualmente o animal, isto é, se ainda se encontra na rua, num canil ou se foi recolhido.

Disponível em https://www.encontra-me.org

2) **Plataforma *FindMyPet* - Ponto de Encontro de Animais Perdidos**

A plataforma *FindMyPet* foi desenvolvida pela Ordem dos Médicos Veterinários (OMV). Oferece um serviço de criação de casos de animais desaparecidos e encontrados e fornece o serviço de acesso à plataforma de forma gratuita sem necessidade de registo. O processo de recuperação de animais desaparecidos cinge-se, no entanto, apenas a cães e gatos. Todos os casos abertos são acompanhados através de um número de caso.

O registo de casos, bem como a pesquisa, é apoiado pela utilização de ferramentas de mapas da Google (*Google Maps*).

Animais desaparecidos e encontrados

Antes de iniciar o processo de inserção de novos casos, é sugerido ao cidadão que utilize a ferramenta de pesquisa para verificação da existência de casos já abertos. Isto é, para alguém que tenha perdido um animal, deverá em primeiro lugar procurar se não existe nenhum caso onde o seu animal tenha sido encontrado.

Caso contrário é sugerido então que seja inserido um novo caso, através do preenchimento de um formulário com informação da data da ocorrência, local e características do cão ou gato em questão.

Depois de inserido o caso, pede-se ao utilizador que se mantenha atento à plataforma e no caso de encontrar o animal correspondente ao seu caso, contactar via telefone ou *e-mail* a OMV.

Disponível em http://findmypet.omv.pt/

2.4 Estado da arte – Tecnologia relevante

As áreas de inteligência artificial relacionadas com os processos de visão computacional podem ser divididas em áreas de estudo como *pattern recognition ou machine learning sendo o deep learning* ou *deep neural networks* um subcampo deste último.

Embora não seja intuito desta dissertação o desenvolvimento de um sistema de *machine learning,* o assunto é abordado para contextualização e elucidação do processo envolvido. O propósito da sua abordagem tem como objetivo a justificação da utilização destes sistemas para auxílio no processo de recuperação dos animais desaparecidos.

2.4.1 Pattern recognition

O campo do reconhecimento de padrões teve o seu início na engenharia (Bishop 2006), sendo alguns dos exemplos da sua utilização, o reconhecimento de caracteres, reconhecimento de texto manuscrito, reconhecimento de voz ou reconhecimento de texto.

"O campo de reconhecimento de padrões preocupa-se com a descoberta automática de regularidades em dados, através da utilização de algoritmos de computador e com o uso dessas regularidades efetuar ações, tais como a classificação de dados em diferentes categorias." (Bishop 2006)

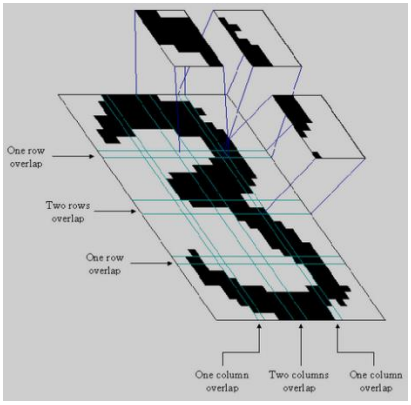

Figura 4 – O caracter "3" dividido em 16 sub-matrizes [8]

A Figura 4 ilustra o modo como um algoritmo divide a imagem em pequenos segmentos que são analisados separadamente de forma a tentar descobrir-se se determinado padrão se encontra presente ou não.

2.4.2 Machine Learning (ML)

"*Sometime in the early 90s people started realizing that a more powerful way to build pattern recognition algorithms is to replace an expert (who probably knows way too much about pixels) with data (which can be mined from cheap laborers).*" (Tomasz Malisiewicz 2015).

O ML surge então como campo da inteligência artificial onde os sistemas são capazes de aprender através dos *inputs* de dados fornecidos. Estes dados são então processados e tratados de forma a descobrirem padrões que possam vir a ser utilizados na análise de novos dados (Konstantinova 2014).

Os tipos de métodos de ML existentes podem então ser classificados em diferentes categorias, sendo as principais: ML supervisionado, ML não supervisionado e reforço de aprendizagem (Konstantinova 2014). De seguida serão dados exemplos de como cada categoria funciona.

ML supervisionado

O técnica de ML supervisionado (Konstantinova 2014) passa inicialmente por treinar o sistema com exemplos, atribuindo-lhes características. Isto é, quando se quer treinar um sistema para detetar as diferenças entre um gato e um cão, é pedido a uma pessoa ou conjunto de pessoas que categorizem imagens, atribuindo a etiqueta "cão" ou "gato" consoante os casos. A partir

[8] http://www.computervisionblog.com/2015/03/deep-learning-vs-machine-learning-vs.html

do momento em que o sistema aprende a distinguir as características de um e de outro, é possível, com a inserção de imagens nunca inseridas que os algoritmos aplicados as distingam e categorizem como sendo "cão" ou "gato".

ML não supervisionado

Para o caso de ML não supervisionado (Konstantinova 2014), ao contrário do que é feito com o ML supervisionado, o conjunto de dados fornecido ao sistema não possui etiquetas. Neste caso, é fornecido um grande volume de dados de forma a que o sistema os categorize em grupos distintos, baseado em *features* (características) dos dados.

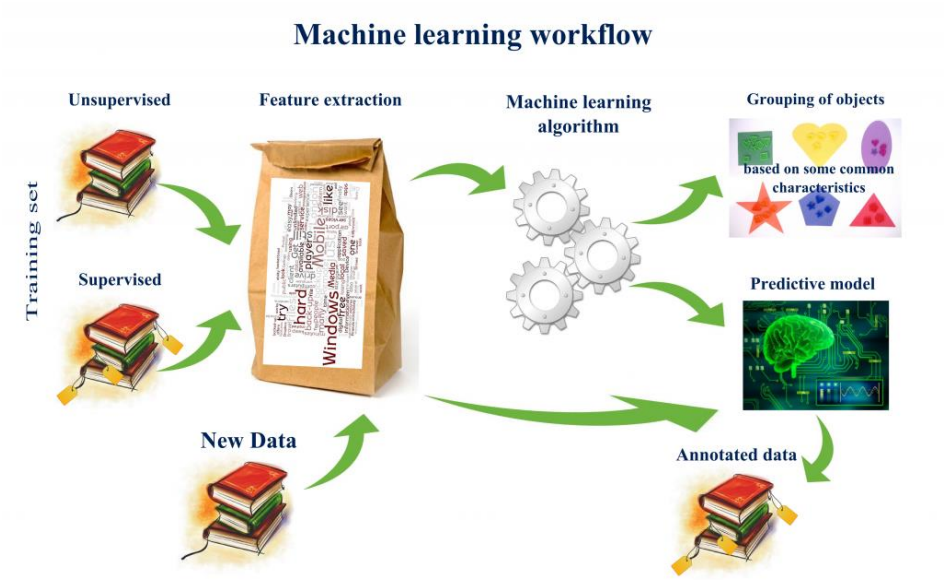

Figura 5 – Exemplo do processo de ML típico [9]

O processo de aplicação de ML típico é, tal como ilustrado na Figura 5, a aplicação de um conjunto de passos que vão desde o processo de treino do sistema até à aplicação de modelo preditivo que atribuirá aos dados um conjunto de etiquetas com o resultado.

2.4.3 Deep Learning ou D*eep Neural Networks*

Nos últimos anos tem-se assistido a um grande número de ferramentas inteligentes capazes de executar tarefas como reconhecimento de voz, reconhecimento de pessoas em fotografias, a tradução de texto em contextos diversos, a verificação de correio não solicitado (SPAM) ou até a proliferação de projetos de condução autónoma. Isto deve-se em grande parte à aplicação de *Deep Learning* – também conhecida no contexto académico por *Deep Neural Networks* ou simplesmente redes neuronais – a grandes volumes de dados, habitualmente na ordem dos *terabytes.* Dados como, voz, imagem e texto são então analisados por redes neuronais

[9] http://nkonst.com/machine-learning-explained-simple-words/

multicamada (Parloff 2016) para treino do sistema. Baseado nesta aprendizagem é então possível a estes sistemas de grande escala, classificar de forma bastante fiável um determinado conjunto novo de dados.

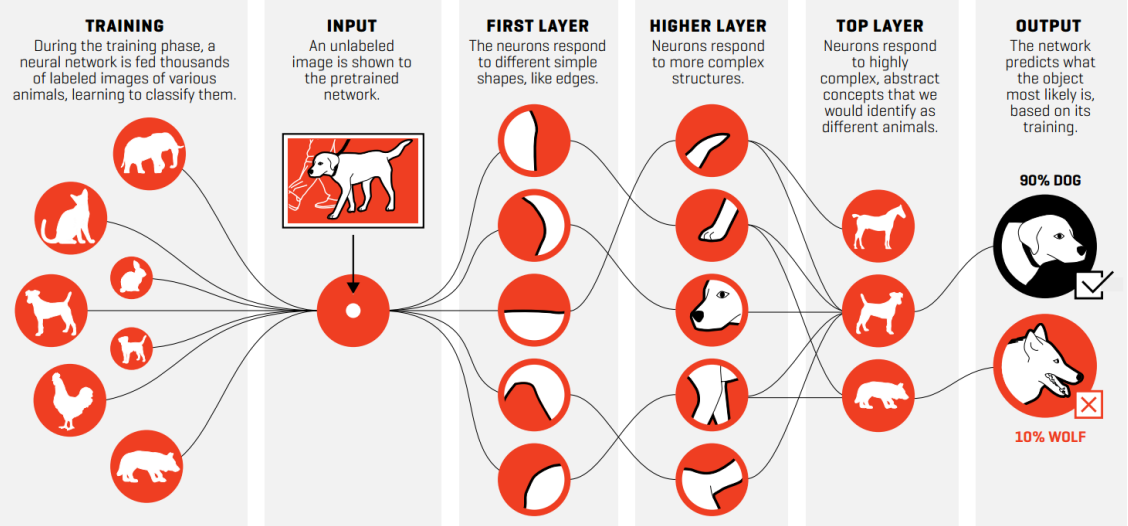

Figura 6 – Aplicação de redes neuronais à classificação de imagens (Parloff 2016)

Na Figura 6 pode verificar-se por que fases transita o processo de análise de imagens de modo a que estas sejam classificadas de forma fiável. O processo começa então pelo treino, onde o sistema é alimentado com milhares de imagens etiquetadas. Findo o processo de treino, é possível submeter ao sistema (*input*) uma imagem nova sem qualquer tipo de etiqueta para que este a avalie. Através da passagem da imagem por diversas camadas de avaliação são detetadas então formas e estruturas mais e menos complexas. O resultado deste conjunto de iterações é então um conjunto de possíveis resultados, classificados com um nível de confiança atribuído pelo sistema.

2.4.4 Ferramentas visão computacional

Os grandes colossos tecnológicos têm vindo a disponibilizar as suas redes neuronais através de serviços comerciais. Empresas como a Google, Microsoft, IBM ou Facebook têm contratado especialistas em inteligência artificial para os seus projetos, havendo inclusive alguns casos de compra de empresas inteiras dedicadas a estas áreas. Estas ferramentas são disponibilizadas, genericamente através de APIs Representational State Transfer (REST).

No âmbito desta dissertação pretende-se fazer uso de APIs de visão computacional que possam detetar características fundamentais dos animais. Não é intuito deste trabalho o desenvolvimento de algoritmos próprios de deteção. A referência à ferramenta OpenCV tem apenas o objetivo de elucidar o leitor da existência deste tipo de soluções.

2.4.4.1 Google Cloud Vision

O projeto Google Brain nasceu em 2011 com o intuito de fazer uso das redes neuronais, tendo em meados de 2012, sido introduzidos os resultados nos produtos de reconhecimento de voz da empresa. Em outubro de 2016 eram já mais de 1000 projetos de *deep learning* que iam desde a sua atividade principal, o seu motor de pesquisa a outros serviços, como o Gmail e Inbox, fotografias, mapas, os carros autónomos, entre outros (Google 2016).

Entretanto, em fevereiro de 2016 a Google lançou a Cloud Vision API na sua versão Beta, com o intuito de disponibilizar a programadores uma ferramenta capaz de descrever os conteúdos de imagens através de modelos treinados de *machine learning*. O acesso a estes conteúdos é essencialmente efetuado através de invocações a uma REST API (Parloff 2016).

O serviço fornecido, na sua primeira versão tem as seguintes características:

- Deteção de caras;
- Descrição do conteúdo das imagens através de etiquetas;
- Deteção de logótipos;
- Deteção de locais de interesse;
- Atributos da imagem, tais como as cores predominantes;
- Conteúdo impróprio;
- Deteção de sentimentos em pessoas;
- Reconhecimento de texto através de *Optical Character Recognition* – OCR.

Este serviço pode ser utilizado gratuitamente até ao limite de 1000 imagens por mês.

2.4.4.2 Microsoft Computer Vision

A utilização de redes neuronais pela Microsoft surgiu em 2011 através de produtos, como o reconhecimento de voz do serviço Bing ou comandos de voz da Xbox, aparecendo posteriormente em serviços como a pesquisa de fotografias, serviços de tradução ou o *ranking* de pesquisa (Parloff 2016).

Computer Vision API Version 1.0

As suas principais características são:

- Deteção de caras e respetivas coordenadas;
- Idade e género;
- Descrição do conteúdo das imagens através de etiquetas;
- Deteção de logótipos;
- Deteção de locais de interesse;
- Atributos da imagem, tais como as cores predominantes;
- Conteúdo impróprio;
- Sentimentos;
- Texto através de *Optical Character Recognition* – OCR;

- Gera descrição do conteúdo da imagem;
- Identificação do tipo e qualidade das imagens.

Esta API poderá ser utilizada de forma gratuita até ao limite de 5000 imagens por mês.

2.4.4.3 IBM Watson

A IBM é uma empresa mundialmente conhecida pela sua história na área das tecnologias de informação, tendo contribuído para a sua evolução, tanto no mercado do *hardware* como do *software*. Em 2011, através do seu projeto Watson – um projeto de inteligência artificial – foi capaz de defrontar e derrotar dois campeões de Jeopardy[10] (Pulli et al. 2012). Mais tarde, em 2015, através da aquisição da AlchemyAPI (*startup* da área de *machine learning*), a IBM quis dar um passo em frente, integrando esta empresa no seu projeto Watson.

Esta ferramenta utiliza também algoritmos de *Deep Learning* para analisar as imagens enviadas, estando disponível através de uma API REST.

Esta ferramenta possui as seguintes características:

- Detetar faces humanas;
- Idade aproximada e género;
- Encontra imagens semelhantes.

Embora não seja publicitada, a ferramenta é também capaz de detetar animais presentes em fotografias, bem como as respetivas raças.

Possui um plano de treino da ferramenta para que seja possível ensinar o sistema a reconhecer os nossos próprios modelos.

2.4.4.4 OpenCV

Surgiu em 1998 como um projeto de investigação da Intel, transformando-se mais tarde, em 2000, numa biblioteca *open source* sob licença *Berkeley Software Distribution* (BSD). Esta biblioteca surgiu com o objetivo de fornecer ferramentas para resolver problemas de visão computacional (OpenCV 2016), possuindo várias centenas de algoritmos para o efeito (Pulli et al. 2012).

Em 2010 foi adicionado ao OpenCV um novo módulo com o intuito de fazer uso dos benefícios de aceleração da unidade de processamento gráfico (GPU). Esta funcionalidade veio então trazer as capacidades dos processadores gráficos aos algoritmos de resolução de problemas de visão computacional, podendo ser implementado por grande parte das funcionalidades desta biblioteca (Manes 2007).

[10] Programa televisivo de cultura geral que através de pistas dadas aos participantes, estes têm de formular as respetivas questões.

Este conjunto de bibliotecas possui interfaces para linguagens de programação como o C, C++ Python e Java sendo suportada nos principais sistemas operativos, nomeadamente Windows, Linux, Mac OS, iOS e Android.

3 Avaliação das ferramentas existentes

Por forma a que sejam tomadas boas decisões, é necessário conhecer bem as hipóteses de solução existentes e de alguma forma selecionar uma solução em detrimento de outra. Esta decisão pode passar, por exemplo pela avaliação das soluções através de ferramentas estatísticas, que permitirão produzir as evidências necessárias para comprovar a mais-valia da sua utilização.

No âmbito desta dissertação pretende-se proceder à avaliação de fotografias através da utilização de ferramentas de análise de imagem. No caso, pretende-se fazer uso de ferramentas de *Machine Learning* que estejam treinadas para reconhecer padrões em imagens e que com alguma certeza possam indicar o que "veem" nelas. Pretende-se, portanto, que estas ferramentas possam indicar com algum grau de certeza que tipo de animal e que raça consta das fotografias submetidas. Esta funcionalidade tem como propósito o enriquecimento dos anúncios de animais desaparecidos / encontrados que serão colocados na plataforma a desenvolver.

Nesse sentido foi efetuado um levantamento de quatro ferramentas que vão de encontro ao propósito estabelecido, sendo então necessária a sua avaliação, por forma a selecionar a que se adapta melhor à realidade deste projeto.

As ferramentas referidas estão disponíveis sob a forma de APIs *online* ou de *software standalone*. Contudo, apenas foram selecionadas ferramentas que disponibilizassem os seus serviços através de APIs *online*.

Este tipo de ferramenta retorna o resultado da avaliação das imagens sob a forma de etiquetas, como por exemplo "dog", "grass" ou "mammal". Isto é, para além do que será o foco das fotografias, ou seja o animal, estas ferramentas retornam também informação sobre a envolvente da fotografia que não interessará para este estudo.

A estas etiquetas, está associada uma percentagem, indicando qual o grau de confiança que a ferramenta possui na existência daquela entidade – representada pela etiqueta – na imagem.

3.1 Grandezas a avaliar

Pretende-se apurar qual das soluções avaliadas é a mais exata na deteção das caraterísticas que constam das fotografias. Para o efeito, foi selecionada uma amostra de 15 imagens de cães e 15 imagens de gatos de raças conhecidas. Como só se pretende fazer uso de uma ferramenta de ML, juntaram-se as duas amostras e efetuaram-se alguns testes com imagens, onde foi possível perceber que as ferramentas de ML selecionadas estão mais treinadas para reconhecer cães do que gatos.

Cada uma das imagens foi então submetida a cada uma das ferramentas, esperando-se que pelo menos fosse identificado que tipo de animal se encontrava na imagem, bem como qual a sua raça. Um dos pressupostos assumidos à partida é que todas as ferramentas detetariam o tipo de animal que as fotografias continham, pelo que, o que se quis verificar foi se também detetariam corretamente a raça do animal.

Todas as ferramentas de ML avaliadas retornam um conjunto de etiquetas que representam as entidades que a ferramenta reconhece na imagem e uma percentagem com o nível de confiança associado a cada etiqueta.

Formalmente, existem então 4 variáveis aleatórias (correspondentes às ferramentas de ML) que representam o grau de identificação, em percentagem, associado à raça de um cão/gato indicado pelas ferramentas, $0 \leq x \leq 1$ com média e desvio padrão σ desconhecidos.

Através do nível de confiança dado pela ferramenta à raça efetivamente constante na imagem submetida, pretende-se então apurar a ferramenta que melhores resultados produz.

3.2 Hipóteses a testar

Após terem sido obtidos todos os resultados da submissão das 30 fotografias a cada uma das ferramentas de ML, pretende-se perceber se existem diferenças significativas entre elas.

Em primeiro lugar estabelece-se uma hipótese nula, isto é, define-se que as quatro ferramentas têm grau de confiança idêntico para um determinado erro α, neste caso de 5%, pelo que se estabelece que há uma igualdade entre as ferramentas.

$$H_0 = \mu_1 = \mu_2 = \mu_3 = \mu_4$$

Através da comparação entre o erro dado e o valor da significância (*p-value*) é possível perceber se existe uma diferença significativa entre as ferramentas.

Caso existam diferenças refuta-se a hipótese nula, e é necessário verificar qual das ferramentas apresenta melhores resultados, como tal, estabelecem-se as hipóteses alternativas onde se compararão as ferramentas duas a duas para determinar qual das ferramentas tem maior grau de confiança.

No pior dos cenários, será necessário testar todas as hipóteses alternativas, ou seja, as que se enunciam abaixo entre H_1 e H_6 para quatro ferramentas.

$$H_1: \mu_1 > \mu_2$$

$$H_2: \mu_1 > \mu_3$$

$$H_3: \mu_1 > \mu_4$$

$$H_4: \mu_2 > \mu_3$$

$$H_5: \mu_2 > \mu_4$$

$$H_6: \mu_3 > \mu_4$$

3.3 Metodologia de avaliação

Para avaliar as diferentes ferramentas, foram submetidas as mesmas 30 fotografias a cada uma delas, de forma a obter-se com que grau de confiança são identificadas as raças dos animais presentes nas imagens. As fotografias selecionadas estão distribuídas em dois segmentos: 15 fotografias de gatos e 15 fotografias de cães.

Todas as raças de cães e gatos foram selecionadas de forma completamente aleatória, bem como as imagens que as representam.

As 15 raças de cães selecionadas são enumeradas de seguida na Tabela 3.

Tabela 3 – Raças de cães selecionadas

Id	Raça cão
1	Chihuahua
2	Galgo
3	Golden Retriever
4	Yorkshire Terrier
5	German Shepherd
6	Pitbull Terrier
7	Beagle
8	Boxer
9	Husky Siberiano
10	Poodle
11	St Bernard
12	Dálmata
13	Dobermann
14	Rottweiler
15	Perdigueiro Português

Para a obtenção de imagens que pudessem representar as raças escolhidas, recorreu-se ao motor de busca da Yahoo. As imagens recolhidas e utilizadas para a avaliação das ferramentas são ilustradas no mosaico da Figura 7. Não foi incluído o motor de pesquisa da Google nem da Microsoft na pesquisa de imagens porque no âmbito deste projeto são incluídas ferramentas de ML destas empresas.

Figura 7 - Fotografias utilizadas na avaliação das raças de cães

Foram ainda selecionadas 15 raças de gatos de forma aleatória, que são de seguida enumeradas na Tabela 4.

Tabela 4 – Raças de gatos selecionadas

Id	Raça gato
1	Persa
2	Siamese
3	Sphynx
4	Norwegian Forest
5	Bengal
6	Sagrado da birmânia
7	Curl Americano
8	Siberiano
9	Maine Coon
10	Bobtail japonês
11	British Shorthair
12	Munchkin
13	Balinês
14	Singapura
15	Devon rex

À semelhança do que foi feito com as imagens dos cães, recorreu-se ao mesmo motor de busca no sentido de pesquisar e recolher imagens que pudessem representar as raças. As imagens recolhidas e utilizadas para a avaliação das ferramentas são ilustradas no mosaico da Figura 8.

Figura 8 – Fotografias utilizadas na avaliação das raças de gatos

Tal como dito anteriormente, cada uma das imagens foi submetida a cada uma das ferramentas, sendo guardado o grau de confiança atribuído à etiqueta da raça constante da imagem. Os valores são expressos em percentagem, onde o valor zero (0) significa que a ferramenta não identificou a raça na imagem e o valor um (1) significa que a ferramenta tem um grau de confiança de 100% na existência da raça naquela imagem.

De seguida, na Tabela 5 são descritos os resultados obtidos na avaliação das 30 imagens. Apesar de todas as ferramentas, em todas as situações, terem detetado com maior ou menor grau de confiança qual a espécie do animal presente nas imagens, a ferramenta Oxford da Microsoft não obteve quaisquer resultados no que diz respeito às raças.

Tabela 5 – Submissão de imagens

Id	Espécie	Raça	Google Cloud Vision	IBM Watson	Clarifai
1	Cão	Chihuahua	0,93	1	0
2	Cão	Galgo	0,91	0,95	0
3	Cão	Golden Retriever	0,8	0,93	0,936
4	Cão	Yorkshire Terrier	0,52	0	0
5	Cão	Pastor Alemão	0,69	0,67	0
6	Cão	Pitbull Terrier	0,85	0	0
7	Cão	Beagle	0,99	0,98	0,982
8	Cão	Boxer	0,91	0,97	0
9	Cão	Husky Siberiano	0,95	0,88	0
10	Cão	Poodle	0,65	0,78	0,873
11	Cão	St Bernard	0,95	1	0
12	Cão	Dálmata	0,94	0,98	0,999
13	Cão	Dobermann	0,6	0	0
14	Cão	Rottweiler	0,71	0,73	0
15	Cão	Perdigueiro Português	0	0	0
16	Gato	Persa	0,87	0,98	0
17	Gato	Siamese	0,9	0,91	0
18	Gato	Sphynx	0,81	0	0
19	Gato	Norwegian Forest	0,71	0	0
20	Gato	Gato-de-Bengala	0,82	0	0
21	Gato	Sagrado da birmânia	0	0	0
22	Gato	Curl Americano	0	0	0
23	Gato	Siberiano	0,63	0	0
24	Gato	Maine Coon	0,72	0	0
25	Gato	Bobtail japonês	0	0	0
26	Gato	British Shorthair	0,86	0	0
27	Gato	Munchkin	0	0	0
28	Gato	Balinês	0,67	0	0
29	Gato	Singapura	0,66	0	0
30	Gato	Devon rex	0,65	0	0

3.4 Teste de hipóteses

Como se pretende testar o mesmo conjunto de imagens para cada uma das ferramentas, utilizar-se-á o teste estatístico ANOVA para variáveis emparelhadas.

Caso se verifique a condição:

$$p\ value > \alpha$$

Não se poderá rejeitar a hipótese nula, significando que as ferramentas têm um grau médio de confiança idêntico. No caso contrário, em que o *p-value* é menor ou igual ao erro α estabelecido, poderá concluir-se que há diferenças entre as ferramentas e terá de se verificar qual delas apresenta melhor grau médio de confiança.

Devido ao facto de a ferramenta de ML da Microsoft não ter apresentado qualquer resultado positivo, tanto na avaliação de cães como de gatos, esta foi excluída dos testes.

No caso da rejeição da hipótese nula recorrer-se-á a um teste *post-hoc* como o *Tukey test* para verificar entre que ferramentas existem diferenças.

Por fim, baseado nos resultados do *Tukey test* serão comparadas as ferramentas onde há diferenças, duas a duas, através do método *Student's t-test* até que se alcance qual das ferramentas possui melhores resultados.

3.5 Resultados

Na Tabela 6 são apresentados os resultados do teste estatístico ANOVA para os grupos de ferramentas selecionadas onde é possível verificar que existe um intervalo de confiança com um grau de confiança, 95%, em que o grau médio de identificação associado à raça de um cão/gato indicado pela ferramenta Google Cloud Vision está contido no intervalo (0.5202,0.7931).

Pode-se verificar ainda pelos valores produzidos pelo teste estatístico ANOVA que o valor de *p-value* < α, o que leva claramente a rejeitar H0, ou seja, a hipótese em que as ferramentas teriam graus médios de identificação iguais. Visto que do teste ANOVA apenas se pode concluir que há diferenças entre as médias será então necessária a utilização do Tukey Test para verificar entre que ferramentas é que existem diferenças.

Da aplicação do Tukey Test – Tabela 7 – a cada par de grupos podemos concluir que existem diferenças entre todos os grupos, através da avaliação dos valores *q-stat* e *q-crit*. Ou seja, em todos os grupos comparados o valor de *q-stat* é superior ao valor de *q-crit*.

Por fim, como há diferenças entre todos os grupos terá de ser aplicado o Student's t-test a cada par de aplicações para verificar qual delas é a melhor.

Visto que já se tinha rejeitado a hipótese da igualdade das médias, pretende-se avaliar o *effect size* que quantifica a relação entre dois grupos, de modo a perceber-se quão grande é a diferença entre os grupos.

Tendo-se concluído que os graus médios de identificação associados à raça de um cão/gato indicados pelas ferramentas Google Cloud Vision e IBM Watson são diferentes. O valor do coeficiente Cohen d, indica que a diferença entre os dois graus médios é razoavelmente elevada (acima do valor que qualifica um *effect size* médio).

Tabela 6 – Teste estatístico ANOVA (Single Factor)

DESCRIPTION					**Alpha**	**0,05**		
Groups	**Count**	**Sum**	**Mean**	**Variance**	**SS**	**Std Err**	**Lower**	**Upper**
Google Cloud Vision	30	19,7	0,656667	0,103809195	3,010466667	0,068661	0,520195026	0,793138
IBM Watson	30	11,76	0,392	0,212837241	6,17228	0,068661	0,25552836	0,528472
Clarifai	30	3,79	0,126333	0,107646437	3,121746667	0,068661	-0,010138307	0,262805
ANOVA								
Sources	**SS**	**df**	**MS**	**F**	**P value**	**F crit**	**RMSSE**	**Omega Sq**
Between Groups	4,218806667	2	2,109403	14,9147214	2,69502E-06	3,101296	0,70509388	0,236184
Within Groups	12,30449333	87	0,141431					
Total	16,5233	89	0,185655					

Tabela 7 – *Tukey test* aplicado aos pares de ferramentas

TUKEY HSD/KRAMER			**alpha**	**0,05**						
Groups	***mean***	***n***	***ss***	***df***	***q-crit***					
Google Cloud Vision	0,656666667	30	3,010467							
IBM Watson	0,392	30	6,17228							
Clarifai	0,126333333	30	3,121747							
		90	12,30449	87	3,371931034					
Q TEST										
Group 1	***Group 2***	***mean***	***std err***	***q-stat***	***lower***	***upper***	***p-value***	***x-crit***	***Cohen d***	***sig***
Google Cloud Vision	IBM Watson	0,264667	0,068661	3,854673816	0,033145716	0,496188	0,020975431	0,231521	0,703764	yes
Google Cloud Vision	Clarifai	0,530333	0,068661	7,72391189	0,298812382	0,761854	1,31584E-06	0,231521	1,410187	yes
IBM Watson	Clarifai	0,265667	0,068661	3,869238074	0,034145716	0,497188	0,020402944	0,23ç1521	0,706423	yes

Tabela 8 – T Test Google Cloud Vision e IBM Watson

T Test: Two Independent Samples									
SUMMARY			Hyp Mean Diff	0					
Groups	*Count*	*Mean*	*Variance*	*Cohen d*					
Google Cloud Vision	30	0,656667	0,103809						
IBM Watson	30	0,392	0,212837						
Pooled			0,158323	0,665161253					
T TEST: Equal Variances				Alpha	0,05				
	std err	*t-stat*	*df*	*p-value*	*t-crit*	*lower*	*upper*	*sig*	*effect r*
One Tail	0,102736952	2,576158	58	0,006279686	1,671553			yes	0,32043
Two Tail	0,102736952	2,576158	58	0,012559372	2,001717	0,059016	0,470317019	yes	0,32043
T TEST: Unequal Variances				Alpha	0,05				
	std err	*t-stat*	*df*	*p-value*	*t-crit*	*lower*	*upper*	*sig*	*effect r*
One Tail	0,102736952	2,576158	51,85252	0,006439537	1,674776			yes	0,336849
Two Tail	0,102736952	2,576158	51,85252	0,012879073	2,006783	0,058496	0,4708374	yes	0,336849

Os T Test efetuados para os restantes dois pares (Google Cloud Vision e Clarifai) e (IBM Watson e Clarifai) têm uma interpretação idêntica à feita a este par e que são descritos de seguida:

Google Cloud Vision e IBM Watson – d = 0.66516

Google Cloud Vision e Clarifai – d = 1.631

IBM Watson e Clarifai – d = 0.663

4 Design

"*Architecture represents the significant decisions, where significance is measured by cost of change.*" – Grady Booch

Neste capítulo pretende-se dar ênfase ao *design* da solução, tendo em consideração as decisões tomadas desde a fase de levantamento de requisitos até ao desenho da arquitetura da solução. Tem início com o levantamento dos requisitos funcionais e não funcionais capturados nas reuniões de trabalho com a equipa do projeto, bem como com a definição dos principais casos de uso. É proposto, com base no problema, um *design* e alternativas para a arquitetura da solução, baseadas nos componentes que se pretende que façam parte da mesma.

Sempre que necessário foram utilizados diagramas *Unified Modeling Language* (UML) e diagramas *Business Process Model and Notation* (BPMN) para a descrição das propostas de solução. Foram ainda incluídos alguns padrões empresariais que darão resposta aos requisitos de fiabilidade na entrega de mensagens do sistema, tendo sido representados através de diagramas *Enterprise Integration Patterns* (EIP).

4.1 Identificação de requisitos

Durante a fase de levantamento de requisitos de *software*, são "capturadas" as funcionalidades/comportamentos que o sistema a desenvolver deverá possuir, denominando-se requisitos funcionais. As funcionalidades associadas à qualidade do *software* são por outro lado denominadas por requisitos não funcionais.

A seguir são listados todos os requisitos que foram levantados nesta fase.

4.1.1 Requisitos funcionais

"Os requisitos funcionais de um sistema descrevem o que o sistema deve fazer. Estes requisitos dependem do tipo de *software* a ser desenvolvido, dos utilizadores esperados do *software* e da abordagem geral utilizada pela organização no momento da sua elaboração. Quando expressos como requisitos de utilizador, os requisitos funcionais são habitualmente descritos de uma forma abstrata para que seja percebido pelos utilizadores do sistema. Contudo, os requisitos funcionais mais específicos descrevem as funções do sistema, os seus *inputs, outputs,* exceções, entre outros." (Sommerville 2009).

Devido ao elevado número de requisitos levantados, optou-se pela sua divisão em três partes com o objetivo de serem produzidas três entregas do produto. Esta divisão teve em linha de consideração a prioridade atribuída pelos *stakeholders* a cada uma das funcionalidades especificadas. Isto é, foi-lhes pedido que atribuíssem uma prioridade entre um e três que se traduziu nas três iterações de desenvolvimento que se seguem. Por forma a conseguir perceber-se quem são os atores envolvidos, bem como as ações e funcionalidades que se pretende que o sistema possua, os requisitos foram escritos sob a forma de *user stories*.

1ª iteração de desenvolvimento

Na primeira iteração de desenvolvimento, pretende-se dar resposta aos requisitos com prioridade um (1) que permitirão a colocação do *website online.*

- Como futuro anunciante, eu quero registar-me no sistema através da utilização de informação básica, tal como: nome, telefone, *e-mail* e *password* para que apenas eu tenha acesso aos meus anúncios, bem como a minha informação;
- Como futuro anunciante, quero que o sistema me envie um *link* de confirmação para o *e-mail* por mim fornecido no registo, evitando o *login* até que seja confirmada a minha conta, por forma a diminuir a possibilidade de usurpação de identidade;
- Como utilizador do *website*, quero poder autenticar-me através de *e-mail* e *password* para que possa ter acesso a funcionalidades de acesso restrito;
- Como administrador do sistema, quero poder adicionar ou remover utilizadores dos grupos existentes para que estes passem a ter mais ou menos permissão sobre o sistema;
- Como utilizador registado, ao encontrar um animal na rua, pretendo publicar um anúncio com as características do animal, bem como submeter algumas fotografias do mesmo, de forma a que seja possível ao sistema verificar se existem anúncios de animais desaparecidos que possam fazer correspondência com o meu;
- Como utilizador registado, quando um animal meu desaparecer, pretendo publicar um anúncio com as características do animal, bem como submeter algumas fotografias do mesmo, de forma a que seja possível ao sistema verificar se existem anúncios de animais encontrados que possam fazer correspondência com o meu;

- Como anunciante, quero que o sistema me notifique por *e-mail* caso existam anúncios semelhantes àquele que publiquei de forma a conseguir encontrar um anúncio correspondente ao meu;
- Como anunciante, pretendo filtrar os anúncios ativos do *site* por local e características dos animais de modo a poder encontrar de forma "manual" um anúncio correspondente ao meu;
- Como utilizador anónimo do *website*, pretendo clicar sobre uma determinada fotografia de um anúncio e assim abrir os detalhes do mesmo;
- Como anunciante, quero entrar nos detalhes do meu anúncio e poder gerar um documento em formato A4 (tipo panfleto) com os detalhes sobre o mesmo e com os meus contactos para que possa distribuir pela zona do acontecimento;
- Como utilizador não autenticado, quero, ao ver um anúncio, poder também ver os dados básicos do anunciante, como o seu nome;
- Como anunciante, quero que o sistema esconda o meu contacto telefónico nos meus anúncios e registe quem quiser obter essa informação para poder ter alguma segurança na divulgação destes dados;
- Como anunciante, quero que o sistema esconda o meu *e-mail* e possua um formulário para contacto nos meus anúncios, de forma a ser contactado pelo sistema e nunca diretamente por outros anunciantes;
- Como anunciante, quero poder partilhar uma fotografia do meu anúncio, bem como um título e uma descrição nas redes sociais Facebook, Twitter, Google+ e Instagram de forma a usufruir da rápida propagação de informação nestes meios;
- Como anunciante, quero ter a possibilidade de depois de o meu anúncio expirar automaticamente, poder alterar o seu estado, publicando-o ou cancelando-o;
- Como utilizador não autenticado, quero poder escolher a língua, português ou inglês, para a apresentação de conteúdos do *website*.

2ª iteração de desenvolvimento

Após a colocação *online* do *website* seguir-se-á a iteração dois que contempla ainda funcionalidades importantes do sistema, mas que não implicam, para os *stakeholders*, a sua entrada em produção.

- Como administrador do *website*, quero ter um formulário onde possa preencher *username*, *password*, *e-mail* e grupos de permissão, para poder criar novos utilizadores do sistema;
- Como administrador do *website*, quero que o sistema coloque uma marca de água em cada fotografia submetida nos anúncios para que seja publicitado o *website* no caso de partilha e também para que não sejam copiadas as imagens para outros *websites*;
- Como utilizador registado, quero poder marcar um anúncio como favorito de forma a que eu seja notificado por *e-mail* em caso de resolução do mesmo ou outro tipo de mudança de estado;

- Como futuro anunciante, quero não ter que decorar mais um par de *username* e *password* e poder registar-me no sistema através das credenciais que utilizo nas redes sociais Facebook e Google+ para poder colocar anúncios no *website*;
- Como administrador do *website*, quero ter a possibilidade de bloquear utilizadores tanto por algum tipo de má conduta como por extinção de vinculo com algum utilizador para que os utilizadores não possam efetuar novamente *login*;
- Como administrador do sistema quero que os anúncios colocados no *website* expirem automaticamente após 30 dias, para conseguir ter atualização do estado do caso por parte do anunciante;
- Como anunciante, quero ter a possibilidade de marcar num mapa, a localização do desaparecimento/aparecimento de um animal para que o sistema utilize esta informação para me ajudar a encontrar um anúncio correspondente;
- Como utilizador registado, quero poder utilizar o serviço PayPal para efetuar doações espontâneas ajudando assim a associação com os custos do projeto;
- Como utilizador (pré)registado, quero poder reenviar o *link* de confirmação da conta de utilizador no caso de não ter recebido ou ter apagado o *link* para poder confirmar a conta e assim conseguir aceder ao sistema.

3ª iteração de desenvolvimento

Por fim, na terceira e última iteração acrescentar-se-ão as funcionalidades com menos prioridade, fechando o ciclo de desenvolvimento da solução.

- Como administrador do *website*, quero ter uma lista de utilizadores que pertencem a cada grupo de permissão para controlar quem pertence a que grupo e que permissões possui;
- Como utilizador registado do *website*, no caso de não me recordar da *password* quero no ecrã de *login* ter uma funcionalidade que me envie um *e-mail* com um *link* de recuperação para poder fazer um *reset* à *password* e voltar a entrar no sistema. O mesmo só deverá ser possível após o preenchimento do *username* utilizado no sistema;
- Como administrador do *website*, quero ter um serviço para criar grupos de permissão para que seja possível adicionar utilizadores a estes grupos;
- Como anunciante, quero poder adicionar e alterar a qualquer altura os dados pessoais, bem como os meus contactos para que o sistema reflita os novos dados;
- Como utilizador do *website,* quero, no âmbito do Regulamento Geral sobre a Proteção de Dados (RGPD 2016/679), poder remover a minha conta e ser esquecido para que não possam rastrear nada que tenha feito no *website*;
- Como anunciante, quero poder ver num mapa pontos de interesse como clínicas veterinárias, hospitais e CRO para poder ter acesso a contactos telefónicos e moradas desses pontos;
- Como administrador do *website* quero que as imagens submetidas nos anúncios sejam validadas quanto ao seu conteúdo e no caso de existência de dúvidas passem por um

processo de aprovação para que não sejam publicados anúncios com imagens impróprias;
- Como moderador do *website,* quero ter uma funcionalidade para visualização de todos os anúncios para aprovação e aprová-los para poderem ficar visíveis para todos os utilizadores;
- Como administrador da associação, quero poder adicionar, alterar e remover notícias do *website* para promover o projeto na página de entrada;
- Como administrador da associação, quero poder adicionar e alterar parcerias e colocá--las em destaque no *website* para enaltecer os apoios dados por estes;
- Como administrador do *website* quero que os *inputs* em formulários sejam validados através de uma ferramenta como o reCAPTCHA para evitar que o sistema aceite entradas forjadas através de *bots*.

4.1.2 Requisitos não funcionais

"Tal como o nome sugere, os requisitos não funcionais são requisitos que não estão diretamente ligados com serviços específicos que o sistema disponibiliza aos seus utilizadores." (Sommerville 2009)

Este tipo de requisito é utilizado para definir características, propriedades e restrições a que o sistema está sujeito. Propriedades como fiabilidade ou tempos de resposta, mas também restrições para o sistema a construir, como por exemplo a definição de capacidades dos dispositivos de i*nput*/o*utput* do sistema (Sommerville 2009). É comum ainda, ser utilizado para a definição de requisitos de usabilidade, desempenho, disponibilidade ou segurança.

Tal como acontece com os requisitos funcionais, também com os requisitos não funcionais foi tida em consideração a prioridade atribuída pelos *stakeholders.* Os requisitos são os que se apresentam de seguida, também escritos sobre a forma de *user stories*.

1ª iteração de desenvolvimento

Na primeira iteração é esperado que sejam à partida tidas em consideração uma série de propriedades chave que o sistema deve possuir. Durante esta fase devem ser asseguradas algumas características de usabilidade, fiabilidade e segurança que são descritas a seguir.

- Como utilizador do *website,* quero poder utilizá-lo nos meus equipamentos com acesso à internet, tais como *smartphone*, *tablet* ou computador pessoal;
- Como utilizador do *website*, quero ter a garantia, pela sua importância, que as mensagens trocadas no âmbito dos meus anúncios ou dos meus favoritos me são sempre entregues;
- Como administrador do *website,* quero ter a garantia que as mensagens trocadas com componentes externos sejam sempre entregues e que, no caso de isso não ser possível, fiquem num estado de erro para que seja tratado de forma manual;

- Como administrador do sistema, quero que sejam utilizados mecanismos de segurança nas comunicações e na base de dados de forma a que fiquem salvaguardados os dados dos utilizadores do *website;*

2ª Iteração de desenvolvimento

Os requisitos definidos nesta fase são requisitos relativos à disponibilidade e desempenho do sistema que se pretendem ver satisfeitos na 2ª iteração de desenvolvimento.

- Como utilizador do sistema, quero que o mesmo responda de forma célere em qualquer pedido, não devendo ultrapassar os 10 segundos em nenhum caso;
- Como administrador do sistema quero fornecer aos clientes desta solução, um sistema com elevada taxa de disponibilidade, idealmente que esteja operacional 24/7;

4.1.3 Casos de uso

De seguida, na Figura 9, é ilustrado o diagrama de casos de uso para as funcionalidades core do sistema que são referidas pelos *user stories* da 1ª iteração de desenvolvimento e que estão relacionadas com o processo de visualização e publicação de anúncios no sistema.

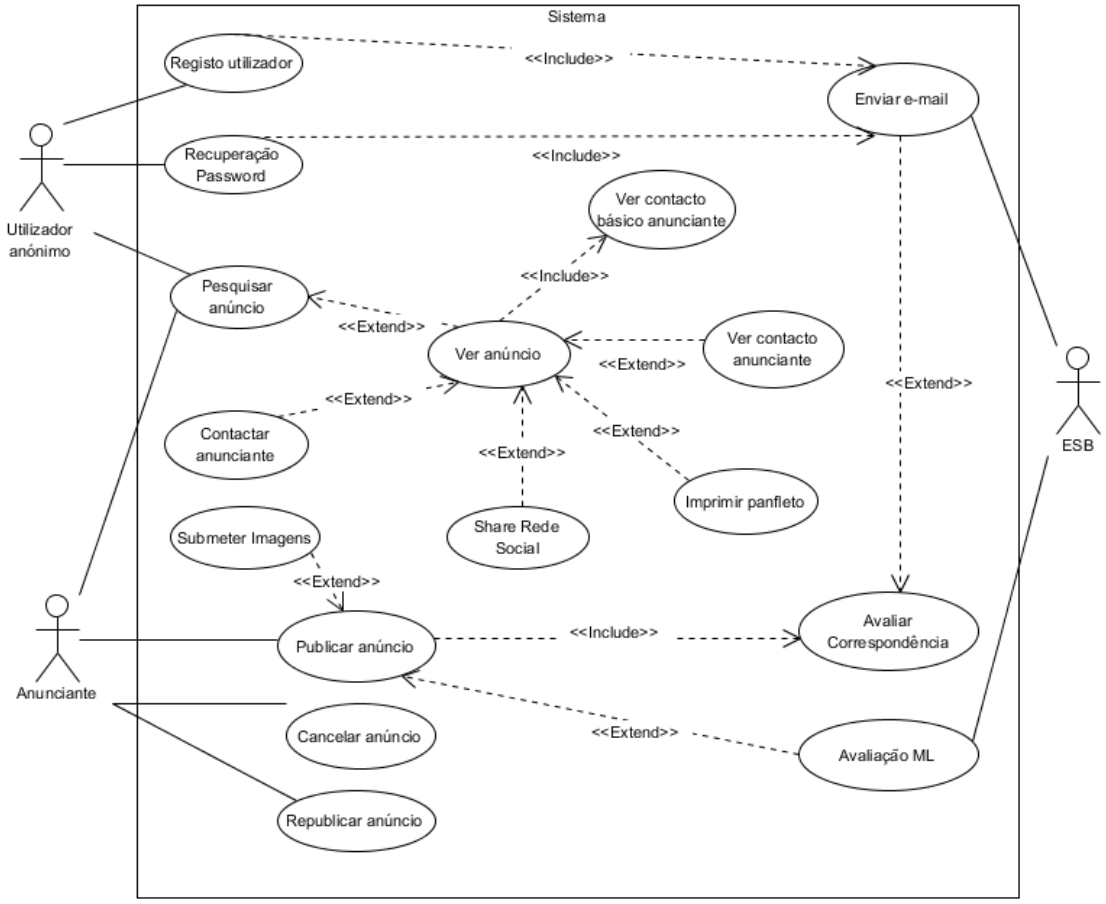

Figura 9 – Diagrama de casos de uso

4.2 *Design* da solução para o problema

Pretende-se que a plataforma a construir possa estar disponível tanto em dispositivos móveis, tais como *tablets* ou *smartphones*, mas também em dispositivos *desktop*. Neste ponto surgem então duas hipóteses para a construção da solução. A construção de aplicações *standalone*, uma por cada tipo de dispositivo, ou a construção de uma só aplicação *web*, que poderá ser acedida através de um simples *browser*.

Embora aplicações locais possam ser mais simples de construir, acarretam diversas dificuldades a outros níveis, nomeadamente a complexidade com a manutenção e instalação do *software* ou a necessidade de requisitos, como *frameworks,* bibliotecas ou outras dependências que terão de estar instaladas no sistema operativo cliente. As aplicações *web* por outro lado, apesar de não serem isentas de problemas, facilitam a gestão, centralizam a manutenção e instalação, e não requerem qualquer interação com o cliente.

Através da utilização de uma abordagem denominada *Responsive Web Design* é possível a construção de aplicações *web* que se adaptam automaticamente aos dispositivos dos clientes, quer sejam *smartphones*, *tablets* ou *desktops*. Isto é, o *website,* através da utilização de HTML e CSS, mostra o seu conteúdo ao cliente em função do tamanho do ecrã onde é visualizado. Por forma a responder aos requisitos levantados, pretende-se construir de raiz, e fazendo uso desta abordagem, uma plataforma *web* que se adapte aos diferentes dispositivos dos utilizadores.

A plataforma terá ainda que recorrer a um conjunto de APIs externas, pelo que a variedade de protocolos e implementações destes sistemas leva a uma dependência muito grande de ligações ponto a ponto. Com o intuito de evitar a implementação de um sistema com elevado acoplamento pretende-se utilizar uma arquitetura do tipo *Enterprise Service Bus* (ESB) para a comunicação entre a aplicação *web* e as APIs externas.

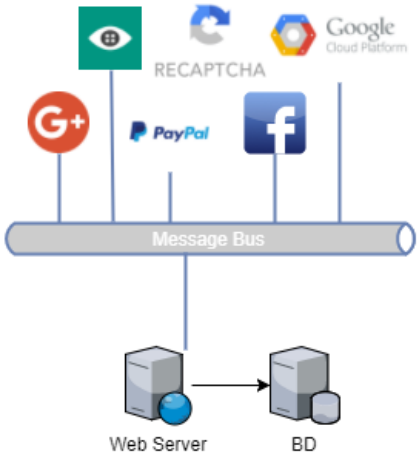

Figura 10 – Visão genérica do sistema

O ambiente a construir terá de ter em conta a modularidade do sistema, bem como a elevada dependência de serviços externos. Como tal, é apresentada na Figura 10 uma visão genérica do

sistema a construir e que poderá ser também entendida por pessoal não técnico. A divisão de responsabilidades entre os diversos serviços é expressa através de diferentes servidores, nomeadamente o servidor de base de dados (BD) e servidor de alojamento da plataforma (*web server*). Ao centro, temos um elemento de comunicação central que permitirá efetuar a troca de mensagens entre os intervenientes nas comunicações e garantir a sua entrega. Por fim são apresentados os serviços a que se pretende ter acesso.

4.3 Arquitetura

Uma das áreas mais importantes do processo de desenvolvimento de *software* é a criação da sua arquitetura, pois representa a forma como os diferentes componentes ou aplicações são organizados e estruturados para formarem um sistema.

A arquitetura do sistema é apresentada através de diversas visões de arquitetura, que representam diversos aspetos e decisões tomadas para a construção do sistema.

4.3.1 Alternativas

De seguida, são apresentadas duas alternativas para a arquitetura do sistema, através de uma breve descrição de cada uma e da utilização de diagramas de componentes.

1) Alternativa 1

O diagrama de componentes ilustrado abaixo, na Figura 11, mostra como os componentes se ligam à aplicação *web* que se pretende criar. Trata-se de uma arquitetura com ligações ponto a ponto para cada uma das APIs que se pretende utilizar.

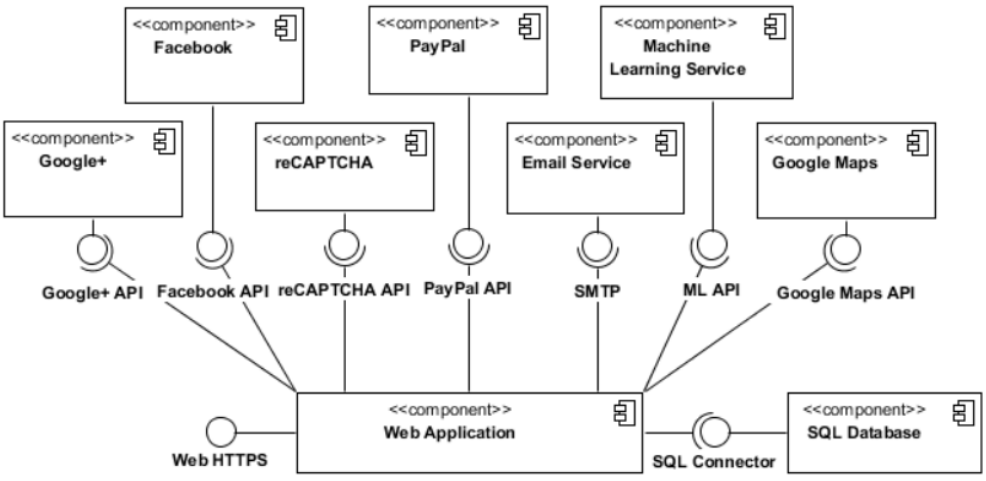

Figura 11 – Diagrama de componentes – Alternativa 1

Os componentes ilustrados na Figura 11, que compõe esta alternativa, são de seguida descritos brevemente por forma a perceber-se qual a sua função como parte do sistema.

Web Application

A *web application* representa a plataforma que será disponibilizada ao cidadão sob a forma de um conjunto de interfaces gráficos. Esta, por sua vez terá de ter acesso a uma base de dados SQL para efeitos de persistência de dados, bem como ter acesso a um conjunto de APIs que vão enriquecer o seu funcionamento.

SQL Database

Este componente representa a forma como são persistidos os dados do sistema, nomeadamente os anúncios, as fotografias, os dados dos utilizadores e outros dados de relevo para o processo.

Google Maps API

Para efeitos de disponibilização de um meio visual de marcação dos locais de desaparecimento e aparecimento de animais recorrer-se-á a uma API externa que proporcione este serviço. O serviço do Google Maps é apenas um exemplo de serviço que poderá ser utilizado.

Facebook API e Google+ API

Pretende-se fazer uso das redes sociais, tanto para o processo de autenticação dos utilizadores como para a partilha dos anúncios criados na plataforma. Nesse sentido será necessária a ligação a estas redes. É possível também que no futuro surjam outras redes sociais, pelo que estas são apenas as mais conhecidas na atualidade em Portugal.

reCAPTCHA

A validação dos dados de entrada colocados no sistema deverá ser feita recorrendo a mecanismos de proteção que evitem a introdução de dados através de *bots*. O reCAPTCHA é uma API disponibilizada pela Google para garantir que os formulários não são submetidos através de *bots*.

PayPal

Para efeitos de financiamento do projeto, apelar-se-á a doações de valores cedidas pelos cidadãos. Para o efeito serão disponibilizadas duas formas de o fazer, através de crédito em conta bancária (simples transferência bancária) ou através da utilização de uma conta PayPal. Neste último será necessária a interligação deste componente com o sistema.

ML API

A utilização de um sistema externo para avaliação das fotografias enviadas pelo utilizador é representada por este componente, habitualmente disponibilizada sobre a forma de uma REST API.

SMTP Server

Este componente permite representar o protocolo de envio de *e-mail* que será utilizado para o envio de informações sobre os casos submetidos, para efeitos de envio de publicidade ou outros.

2) Alternativa 2

O diagrama de componentes representado na Figura 12 pretende representar a forma como cada componente se liga para criar o sistema. O facto de o sistema ser modular permite a qualquer altura substituir componentes individuais, como se de um carro se tratasse e se pretendesse mudar por exemplo uma porta. Esta abordagem facilita o processo de alteração de módulos, pois se existir a necessidade de alterar o comportamento da aplicação bastará, à exceção da API do Google Maps, alterar a forma como o ESB mapeia a informação.

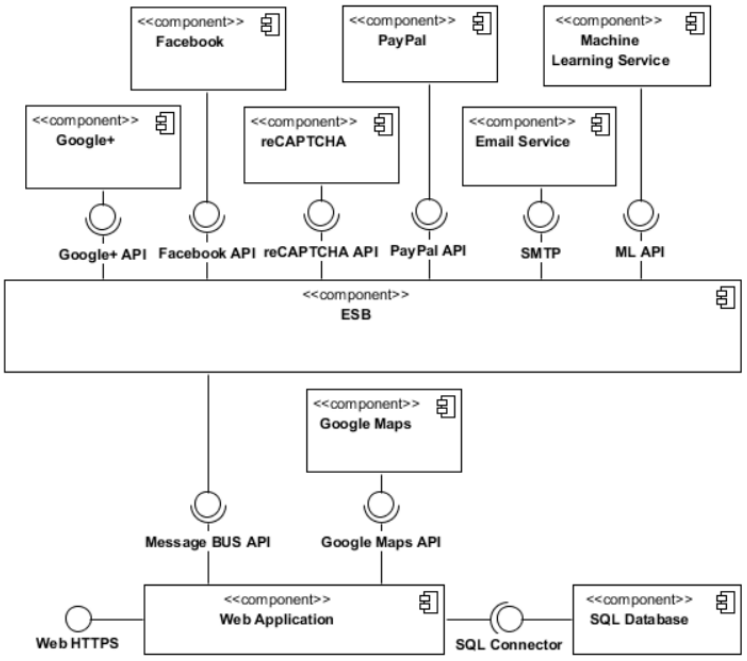

Figura 12 – Diagrama de componentes – Alternativa 2

O componente ESB, introduzido nesta alternativa, é descrito de seguida e tem como intuito a ligação de diversos componentes através de um sistema de comunicação por mensagens.

ESB

A utilização de um ESB permitirá o uso de um mecanismo de envio e receção de mensagens entre os diferentes componentes que compõe o sistema como meio de comunicação principal. Sendo que se existirem alterações às APIs externas, apenas será necessário adaptar o ESB para poder refletir as alterações desse sistema, não sendo necessário proceder a qualquer alteração na plataforma Web.

4.3.2 Arquitetura do *website*

A importância da modularidade dos sistemas é transversal a todas as arquiteturas, pois espera-se que a troca de uma simples peça não implique a alteração de todo o sistema. Isto aplica-se também ao *software*, logo, também às arquiteturas das plataformas *web*.

Para a aplicação deste conceito existem boas práticas que podem ser seguidas, como a aplicação de uma arquitetura do tipo MV*. Nesta família de padrões privilegia-se a segregação entre os diversos papéis existentes num sistema (*separation of concerns*), sendo os mais conhecidos o Model-View-Controller (MVC), Model-View-Presenter (MVP) e Model-View-ViewModel (MVVM).

Para este projeto pretende-se fazer uso de uma arquitetura que seja pouco complexa, escalável, facilmente testável e que se adeque às tecnologias utilizadas, como é o caso de uma arquitetura MVC.

Nesta arquitetura existem três grandes grupos, os *Models*, as *Views* e os *Controllers* que são de seguida descritos de acordo com a Microsoft em (Microsoft 2016b) no âmbito da sua *framework* ASP .NET Core MVC.

Model

O *model* numa aplicação MVC representa o estado de uma aplicação e qualquer lógica de negócio ou operações que devam ser feitas sobre este. A lógica de negócio deve ser encapsulada no *model*, juntamente com qualquer implementação de lógica para persistir o estado da aplicação.

View

As *views* são responsáveis por apresentar o conteúdo na interface com o utilizador, devendo conter o mínimo de lógica possível. No caso de existir lógica apenas deverá estar relacionada com a apresentação de conteúdo.

Controller

Os c*ontrollers* são os componentes que lidam com a interação do utilizador, que interagem com o *model* e, por fim, que selecionam a *view* a ser mostrada. Numa aplicação MVC a *view* apenas mostra informação, o *controller* lida com e responde aos *inputs* e interação do utilizador.

Na Figura 13 pode ver-se de que forma estão ligados os diversos papéis desta arquitetura.

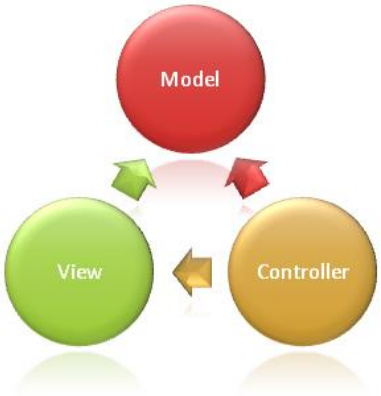

Figura 13 – Arquitetura MVC

4.3.3 Enterprise Service Bus

Segundo alguns autores (Hohpe & Woolf 2003), um *Enterprise Service Bus* (ESB) é uma solução *Middleware* que permite a interoperabilidade entre ambientes heterogéneos utilizando um modelo orientado aos serviços. Um ESB modela o *endpoint* de uma aplicação como um serviço podendo hospedar o agente do serviço localmente ou podendo o serviço ser executado remotamente. Em ambos os casos, o ESB fornece uma camada de abstração que virtualiza o serviço e separa-o das preocupações inerentes à infraestrutura. O ESB torna o serviço acessível a outras aplicações através de um ou mais protocolos de *middleware*. Regra geral, um dos protocolos que o ESB suporta é *Simple Object Access Protocol* (SOAP), mas o sistema não requer que todos os serviços comuniquem via este protocolo. O ESB media as interações entre os *endpoints* de serviço e permite que serviços diferentes interoperem.

Pretende-se fazer uso de um sistema que implemente ESB de forma a poder ligar a *Web Application* com as diversas APIs externas, como é o caso da Cloud Vision API ou o PayPal, de modo mais autónomo e modular, utilizando boas práticas de engenharia e simplificando futuras intervenções. Assim, caso as implementações destas APIs mudem ou caso se queira alterar o prestador de serviços, em princípio só será necessário alterar o ESB e não será necessário alterar a *Web Application*.

A aplicação deste tipo de modelo pretende evitar que se chegue a um dilema conhecido por *Spaghetti Integration*, isto é, um modelo demasiado confuso para ser facilmente percebido, mantido e gerido.

É comum os sistemas chegarem a uma arquitetura semelhante à representada pela Figura 14, onde cada parte está ligada a todas as outras. Com o aumento do número de pontos, é usual chegar-se a algo denominado de *Point-to-Point Hell.*

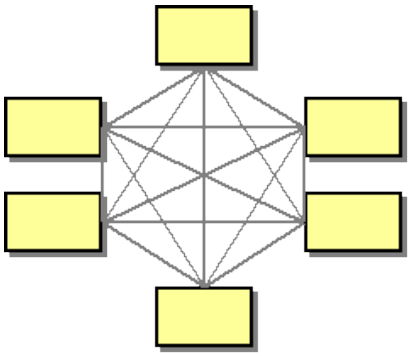

Figura 14 – Arquitetura Point-to-Point

4.3.4 Enterprise Integration Patterns (EIP)

A utilização de mensagens assíncronas no âmbito do desenvolvimento de *software* empresarial, era vista, por Martin Fowler em 2003, como tendo um papel com uma importância relevante, principalmente na integração de aplicações. A integração é indubitavelmente um ponto chave, pois o que se verifica é que as aplicações não podem "viver" isoladas umas das outras (Hohpe & Woolf 2003). Com a utilização de mensagens para integração de aplicações, surgem então alguns problemas frequentes cuja solução acaba por se tornar geral, o que por sua vez faz surgir os padrões.

De seguida são apresentados alguns exemplos de padrões de integração empresarial, que serão utilizados para responder aos requisitos do sistema, nomeadamente a entrega de *e-mails* ou a comunicação com algumas APIs externas utilizadas no projeto.

Guaranteed Delivery

A aplicação deste padrão empresarial permite garantir que a mensagem é entregue ao respetivo destinatário. Antes de a mensagem ser enviada, é em primeiro lugar armazenada fisicamente e só depois se procede ao seu envio. Isto acontece para que, caso exista algum erro na sua transmissão, a mensagem não se perca. Existem diversas situações em que a transmissão da mensagem pode não ser bem sucedida, o servidor de destino poderá estar *offline*, a rede poderá não estar disponível ou ainda poderão ocorrer *time outs* durante o processo (Hohpe & Woolf 2003) . Na Figura 15 está representado o referido padrão de integração empresarial.

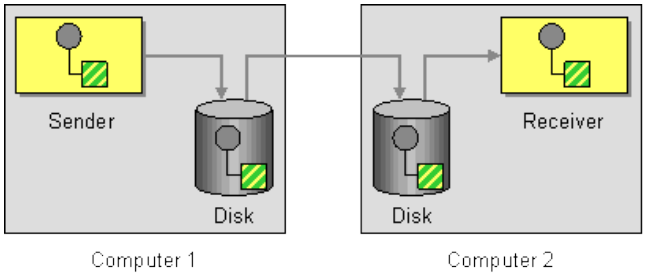

Figura 15 – Aplicação do padrão *Guaranteed Delivery*

Dead Letter Channel

Este padrão define a forma como os sistemas podem tratar as mensagens que não podem ser entregues ao seu destinatário por razões associadas a falhas dos sistemas, a problemas de comunicação ou outros. No caso de não ser possível entregar uma determinada mensagem, o sistema poderá encaminhá-la para um canal especial denominado *Dead Letter Channel* (DLC) para ser tratado *a posteriori* (Duvall et al. 2007). Uma possível apresentação deste padrão pode ser consultada na Figura 16.

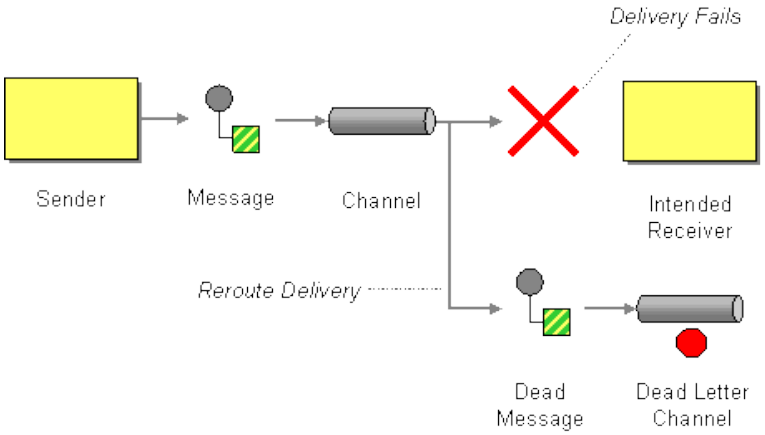

Figura 16 – Aplicação do padrão *Dead Letter Channel*

Message Broker

A utilização de uma unidade central, que possa receber mensagens de múltiplas origens e determinar qual o destino correto para rotear mensagens para os canais corretos, pode ser conseguida através de um *Message Broker*, também referido como o estilo arquitetural *hub-and-spoke*.

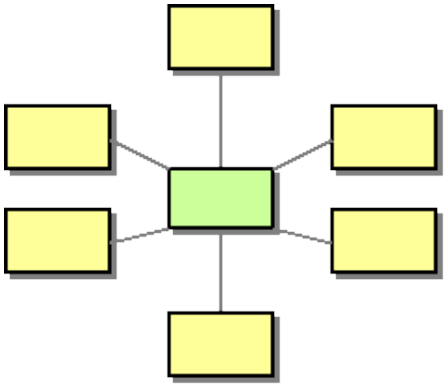

Figura 17 – Arquitetura Hub-and-Spoke

4.4 Processo de publicação de anúncio

A publicação de anúncios na plataforma terá de seguir um processo de negócio pré-definido por forma a que a informação possa ser processada, tratada e moderada. O processo de seguida ilustrado através da Figura 18, descreve por que passos e filtros passam os anúncios de animais desaparecidos.

Logo após a submissão do anúncio por parte do cidadão, são submetidas as fotografias do animal para análise por parte da ferramenta de *Machine Learning* selecionada. Esta por sua vez, quando termina, devolve um conjunto de informações tratadas que serão analisadas pelo sistema. No caso de existirem dúvidas quanto à presença de conteúdo das imagens, haverá necessidade da intervenção de um moderador para efetuar a avaliação manual das imagens. Só após este conjunto de passos será possível enviar o anúncio para publicação na plataforma, ficando então disponível para visualização.

Em *background* será iniciado um subprocesso que tentará fazer a correspondência entre o anúncio que o cidadão está a publicar e os anúncios previamente registados no sistema. No caso de este encontrar correspondência irá notificar o cidadão desta ocorrência. Poderão, obviamente, ser dadas sugestões de correspondência não válidas, pelo que cabe ao cidadão a sua validação.

No caso de não existir qualquer correspondência, o processo terminará automaticamente ao fim de 30 dias. Porém o cidadão poderá, em qualquer momento, terminar o processo através do cancelamento manual do mesmo.

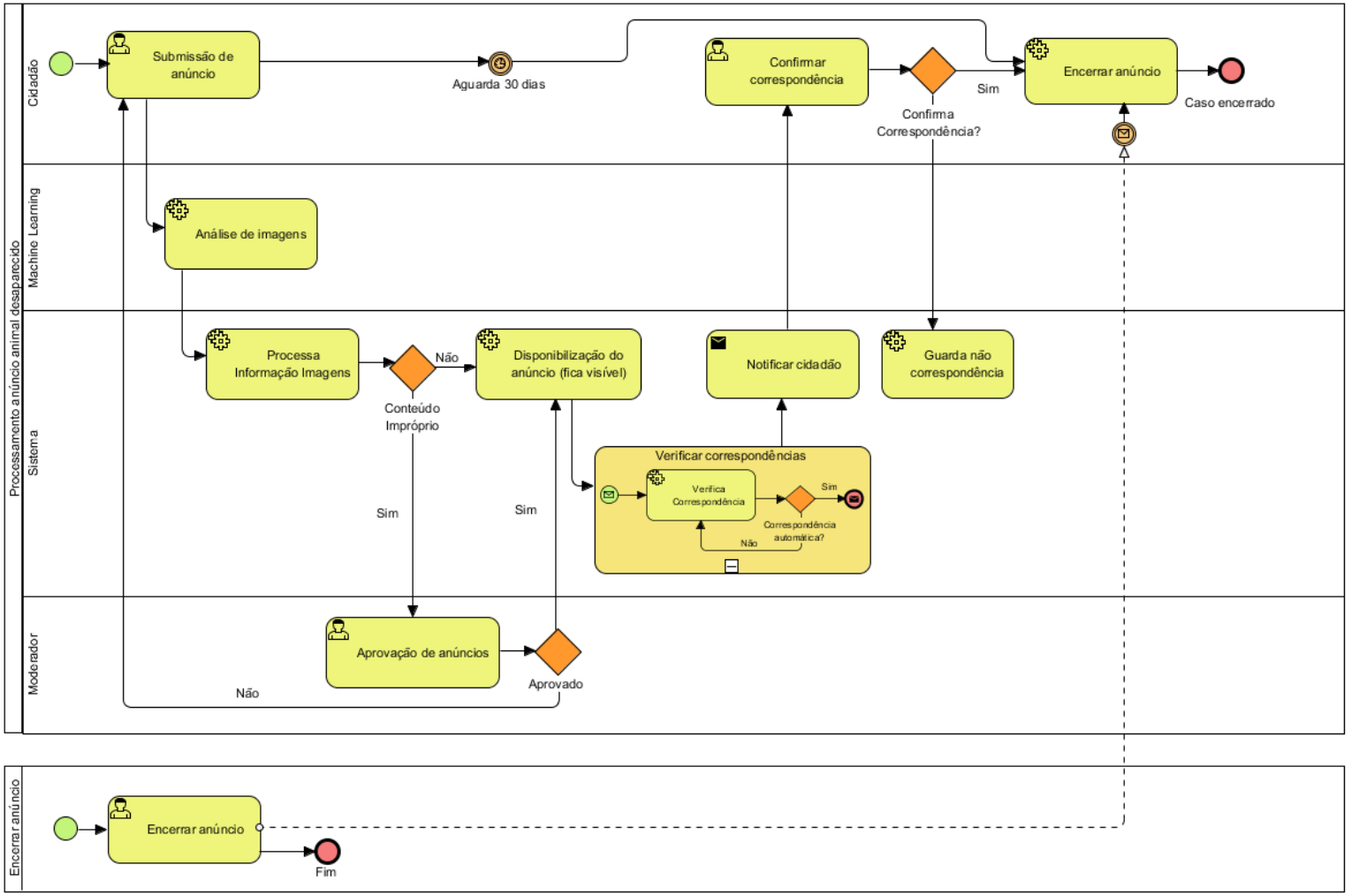

Figura 18 – Diagrama BPMN de processamento de anúncios

Após a submissão de um qualquer anúncio, quer seja ele de um animal desaparecido ou de um animal encontrado, deve ser seguido um conjunto de procedimentos para que se possa tentar perceber se existe algum anúncio similar correspondente.

Através da utilização do diagrama de sequência ilustrado pela Figura 19 pretende-se descrever o processo de publicação de anúncios até à invocação do algoritmo de análise de correspondência. É possível então verificar que após a gravação do anúncio, podem seguir-se dois caminhos.

No caso de existirem fotografias no anúncio é invocado o ESB, para que este invoque a ferramenta de ML. O ESB devolverá a informação para uma API da solução que executará o algoritmo.

No caso de não existirem fotografias nos anúncios, o algoritmo é executado de imediato.

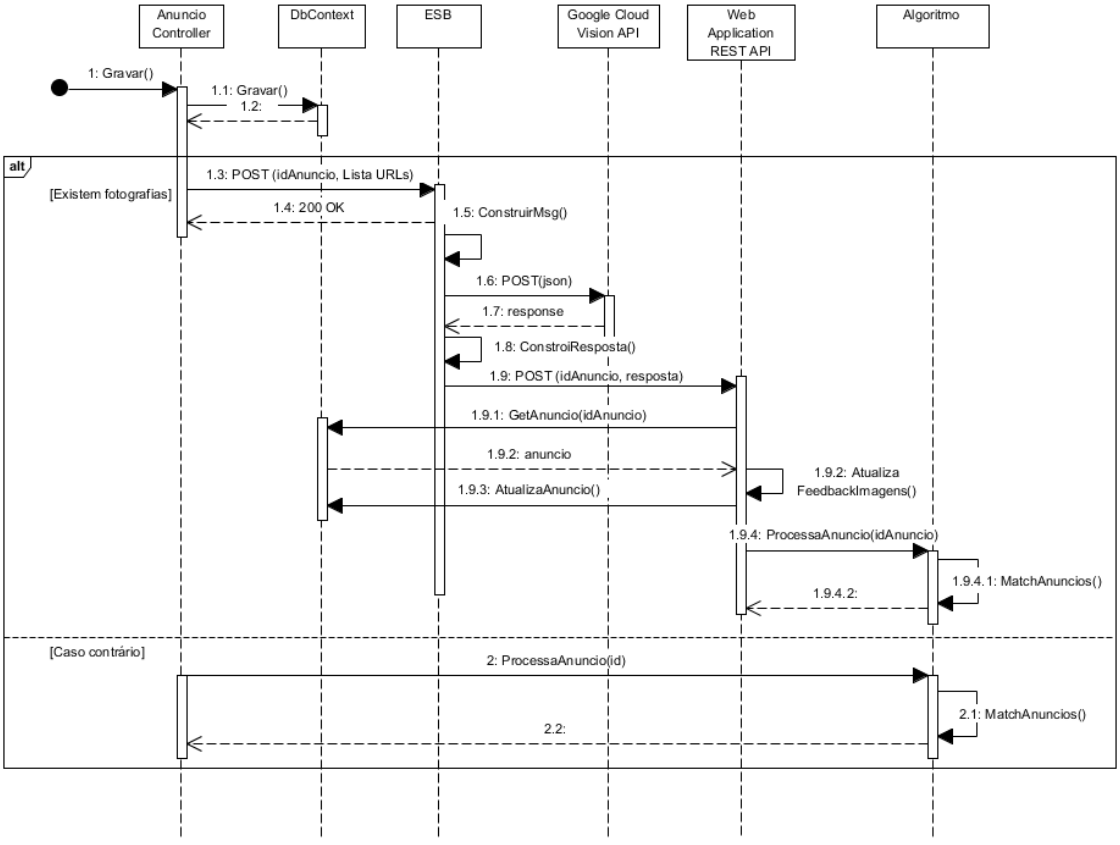

Figura 19 – Diagrama de sequência – Gravação e execução de algoritmo

A instância do algoritmo de processamento de correspondências recebe o identificador (id) do anúncio a processar e trata de obter todos os anúncios passiveis de serem correspondentes. De seguida deverá, através de um sistema de atribuição de pontuação ou peso, identificar quais os

anúncios com maior correspondência, caso exista algum. Através do diagrama de sequência ilustrado na Figura 20, é possível verificar o seu funcionamento.

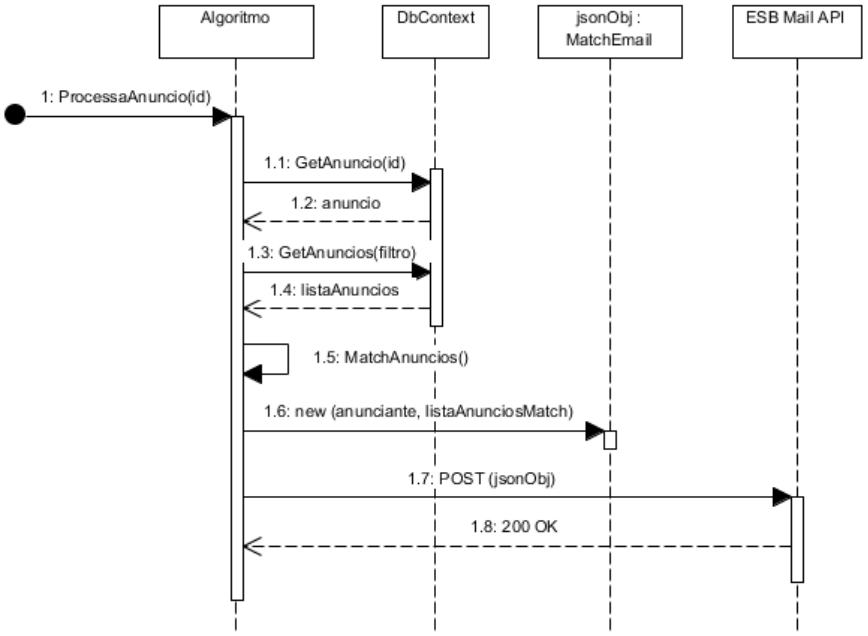

Figura 20 – Diagrama de sequência – Correspondência de anúncios

4.5 Base de dados

Se analisarmos a plataforma concorrente (www.encontra-me.org) podemos, através dos dados publicados na mesma, verificar que em 11 anos de existência foram publicados cerca de 63.900 anúncios. Isto é, em média, apenas 15 anúncios foram publicados por dia.

Os sistemas de gestão de base de dados (SGBD) relacionais existentes permitem, para o volume e taxas de utilização de dados referidos, a validação e integridade dos dados, bem como ter um bom desempenho. Pelo que se poderá concluir que a sua aplicação é adequada.

Para efeito de acesso a dados pretende-se utilizar um *Object Relational Mapping* (ORM) que esteja disponível na linguagem de programação selecionada (orientada a objetos) e que já implemente um conjunto de padrões para acesso a dados, tais como: *Domain Model, Data Mapper, Lazy Loading,* entre outros.

5 Metodologia

"*Success is a journey, not a destination. The doing is often more important than the outcome.*"
– Arthur Ashe

A construção de uma solução baseada na *web*, como abordagem primária ao problema colocado, foi uma das primeiras decisões a serem tomadas. O requisito de que a aplicação teria de estar disponível em diversos tipos de dispositivo, colocou duas hipóteses de solução. A criação de uma solução por dispositivo, entre eles os *smartphones*, *tablets* e *desktops*, ou a criação de apenas uma aplicação *web* que pudesse estar disponível para todos eles. Embora fosse de todo o interesse a criação de aplicações *mobile*, pelo seu fácil acesso a funcionalidades como GPS ou câmara, chegou-se à conclusão que esta solução não era exequível no tempo disponível.

Nesse sentido optou-se nesta fase por centralizar tudo numa só aplicação, de modo a diminuir o período de desenvolvimento, o número de aplicações a manter e a colocação da plataforma *online* mais rapidamente.

Neste capítulo serão apresentadas as abordagens, tecnologias e soluções para os problemas colocados.

5.1 Processo de desenvolvimento

Neste subcapítulo pretende-se abordar um conjunto de questões relacionadas com o processo de desenvolvimento da solução, tendo-se começado por referir algumas práticas ágeis de desenvolvimento de *software.* Isto, por um lado para uma menor duração e esforço a cada integração e por outro lado para ter a possibilidade de ter o *software* sempre num estado *releasable*.

Também é referida a utilização do serviço Bitbucket, sendo este um conhecido repositório de *software* que tem vindo a aumentar o número de funcionalidades, de forma quase sempre gratuita para pequenos projetos, conjugada com o *software* de controlo de versões *git*.

É ainda abordado o tema da realização de testes à solução, nomeadamente a realização de testes unitários para a tentativa de minimização de defeitos em funcionalidades da aplicação *web.*

Por fim são referidas algumas preocupações tidas em conta durante o processo de desenvolvimento.

5.1.1 Continuous Integration e Continuous Delivery

Em projetos de desenvolvimento de *software* era muito comum protelar-se a entrega do produto ao cliente até praticamente ao fim do seu desenvolvimento. No entanto, esta abordagem trazia um risco muito grande, nomeadamente a não aceitação do produto, por desvios ao que realmente era esperado pelo cliente.

Embora se deva integrar o cliente no processo de desenvolvimento de *software*, é ainda de enorme importância proceder-se a entregas do produto de forma regular. Esta metodologia pretende então evitar que aconteçam desvios àquilo que foi apurado na fase de levantamento de requisitos, e que caso estes ocorram sejam corrigidos o mais cedo possível.

Entregar o produto mais cedo ao cliente significa então obter *feedback* mais cedo, o que, como foi dito, ajuda a diminuir o fosso entre o que é pedido e o que é entregue ao cliente.

Estes conceitos poderão ser então aplicados através das abordagens de *Continuous Integration* (CI) e *Continuous Delivery* (CD) tal como os conceitos descritos por (Humble & Farley 2010) e (Duvall et al. 2007).

A abordagem de CI diz-nos que quaisquer alterações efetuadas ao código fonte, estrutura de base de dados, configurações ou outras, devem dar origem a um *commit* para o repositório de código fonte (Duvall et al. 2007). Estas alterações, por sua vez, são então capturadas por um sistema de CI que faz o *build* e que reporta possíveis falhas de integração. Outros autores (Humble & Farley 2010) (Martin Fowler 2006) vão um pouco mais longe defendendo que qualquer artefacto, documento, ou peça de *software* deverá estar incluída neste sistema. A ideia de que tudo o que é necessário para efetuar um *build* do sistema através de um único comando deve estar no sistema de controlo de versões é defendida por alguns autores (Humble & Farley 2010).

Por sua vez, quando se fala de CD refere-se uma abordagem que aplica o padrão *deployment pipeline*, que se foca na automatização dos processos de *build, deploy, test* e *release.* Devendo estes estar ligados entre si de forma sequencial, formando uma cadeia de processos necessários à produção de uma nova *release*. Através da aplicação deste padrão espera-se que qualquer

alteração que seja feita sobre o código fonte, estrutura de dados, configurações ou outras, despoletem uma nova instância do *pipeline* (Atlassian 2017). A execução da *pipeline* permitirá então manter todo o processo de *release* de *software* automatizado, alertando no caso de ocorrência de falhas.

Nesta abordagem defende-se que cada desenvolvimento seja feito sobre o *mainline*, ou seja, um ponto único de integração de *software*, que possa ser entregue ao cliente final a cada *commit*, mantendo-o sempre num estado *releaseable*. Porém há alturas em que tarefas maiores de desenvolvimento não permitem manter o *software* neste estado, podendo surgir situações em que não é possível efetuar uma *release*. Nestes casos pode justificar-se a criação de ramos (*branches*) para o desenvolvimento de tarefas. O surgimento de *bugs* de resolução urgente é um exemplo de uma tarefa que tem de ser executada de imediato, pelo que o desenvolvimento sobre a *mainline* faria com que a tarefa a ser executada fosse perdida ou efetuado *commit* sem esta estar terminada.

No caso deste projeto optou-se pela criação de um ramo de desenvolvimento para que se possa manter o *mainline* (ou ramo *master* no caso do *git*) disponível para qualquer eventualidade, tal como a correção de *bugs*.

5.1.2 Bitbucket Pipelines

Com o intuito de seguir as boas práticas descritas pelas abordagens anteriormente referidas – CI e CD – e evitar-se a utilização de processos manuais, optou-se, neste projeto pela utilização de *git* (*software* controlo de versões), tendo sido o Bitbucket a plataforma selecionada para efeitos de alojamento do repositório e utilização do serviço de pipeline.

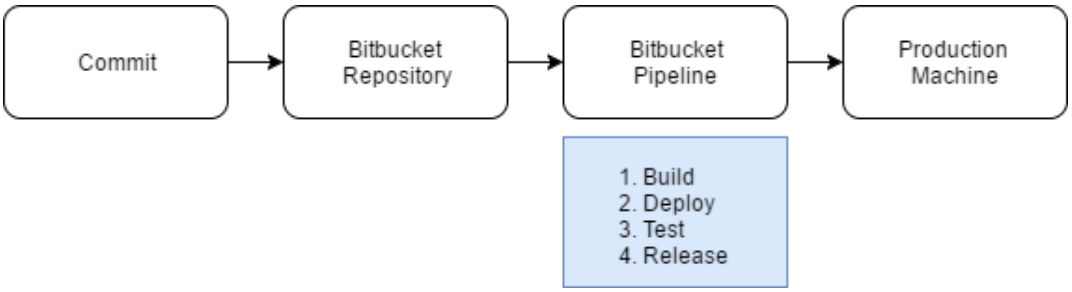

Figura 21 – Representação do *pipeline* implementado

A partir do momento em que é feito um *commit* no ramo *master*, o *software* percorre os passos ilustrados pelo *pipeline* representado na Figura 21. No caso de não existirem erros durante este processo, é feito um *package* que é enviado para o servidor de qualidade, sendo automaticamente colocado em funcionamento. Devido essencialmente à escassez de recursos financeiros, ainda só foram considerados os equipamentos de desenvolvimento e qualidade, no entanto, na finalização da 1ª iteração incluir-se-á o servidor de produção para a entrega do produto.

Ambiente Pipeline

O Bitbucket fornece, para a utilização da funcionalidade *pipelines*, uma imagem Docker por omissão onde é possível correr os *builds* das aplicações lá alojadas. É ainda possível a utilização de imagens colocadas em repositórios externos, como o Docker Hub ou outros. No âmbito deste trabalho foi utilizada uma imagem oficial da Microsoft, alojada no Docker Hub e especificada no parâmetro *image* do ficheiro de configuração bitbucket-pipelines.yml utilizado por esta funcionalidade. Neste ficheiro, que possui o formato Ain't Markup Language (YAML), é possível ainda definir qual o comportamento que o *pipeline* deve ter mediante a origem do *commit,* isto é, se é feito no ramo *develop* ou *master*. No ficheiro de configuração descrito pelo Código 1, é possível consultar a configuração realizada no âmbito deste trabalho.

```
image: microsoft/dotnet:1.1.2-sdk

pipelines:
  default:
    - step:
        script:
           # Restore packages
           - echo "This script runs allways."
  branches:
      develop:
      - step:
          script:
            - echo "This script runs only on commit to the develop branch."
            - dotnet restore
            - dotnet build
            - dotnet test ./SA.Tests/SA.Tests.csproj
      master:
      - step:
          script:
            - echo "This script runs only on commit to the master branch."
            - dotnet restore
            - dotnet build --configuration Release
            - dotnet test ./SA.Tests/SA.Tests.csproj
            - dotnet publish --configuration Release
            -                                                                cd
/opt/atlassian/pipelines/agent/build/SA.Web/bin/Release/netcoreapp1.1/publish/
&& tar -zcvf ../sa_web.tar.gz . && cd ..
            -  sftp  bitbucket@fuso.no-ip.org:/home/bitbucket/toproduction  <<<
$'put sa_web.tar.gz'
            - ssh bitbucket@fuso.no-ip.org 'sudo ./prod.sh deploy'
```

Código 1 – Ficheiro de configuração do *pipeline* Bitbucket

Informações sensíveis

Numa aplicação *web* como a descrita por este projeto, é comum aceder-se a outros componentes, quer sejam eles um SGBD ou uma API. Estes componentes, por norma, estão protegidos com sistemas de autenticação através de *username* e *password* ou por outro meio, como as chaves de aplicação. Estes dados, pela sua sensibilidade, devem ser guardados de

forma segura para que seja maximizada a privacidade. No entanto, estes dados devem estar disponíveis para a aplicação poder funcionar.

Existem então diversas formas de guardar estas informações, tanto em ambientes de desenvolvimento como de produção, entre elas estão a utilização de variáveis de ambiente, ficheiros de configuração ou Secret Managers[11].

Embora seja uma prática comum guardar este tipo de informação diretamente no código fonte, é considerada uma má prática, por trazer vários problemas ligados à segurança e manutenção deste tipo de informação. A manutenção, porque torna-se necessário recompilar o código fonte sempre que há alterações a estas informações e segurança porque todos os intervenientes no processo de desenvolvimento poderão ter acesso a estas informações. A utilização de repositórios de código fonte propicia também a falta de privacidade deste tipo de informação.

Uma forma de conseguir o armazenamento de dados sensíveis é a utilização de variáveis de ambiente do sistema operativo *host* ou hospedeiro. Embora haja alguns riscos que se evitam, como o envio de informações sensíveis para repositórios de código fonte, é necessário perceber-se que estas variáveis, habitualmente não encriptadas, estão acessíveis a qualquer utilizador do sistema *host* (Mozilla 2017a).

Outra forma é a inclusão deste tipo de informação em ficheiros de configuração; se forem tomadas as devidas precauções, é um método que pode ser utilizado. Para maximizar a segurança destas informações, uma das formas é a aplicação de encriptação sobre os dados e a colocação deste tipo de ficheiro diretamente nos servidores de produção.

Para a utilização deste método foi necessário, em primeiro lugar, configurar o ficheiro *.gitignore* que tem por objetivo ser utilizado para refletir todos os ficheiros que não devem ser enviados para o repositório de código fonte. Neste ficheiro foi então adicionado o caminho para o ficheiro JSON appsettings.json, que possui todas as configurações, *connection strings*, *keys* e *passwords* do *website*. Com esta configuração, o ficheiro referido será descartado em todos os *commits*.

No momento da colocação do *website* em produção, o *script* que é executado no servidor de produção é responsável também por utilizar a cópia constante no servidor para utilização do *website*.

Colocação em produção

Embora não seja de esperar que aconteça, por se tratar de uma fase adiantada do processo, caso haja necessidade de reverter o *software* para uma versão anterior, é disponibilizado no servidor de produção um *script* para o efeito. Este *script* coloca tanto o *software* como a respetiva base de dados na versão pretendida.

[11] Aplicação responsável por guardar informações sensíveis como *passwords* ou chaves

Após o *release* da solução, esta é compactada num ficheiro tar.gz que é enviado de seguida para o servidor de produção através de uma ligação SSH File Transfer Protocol (SFTP). Imediatamente a seguir é invocado o *script* constante referido pelo Código 2 com a opção *release* que irá efetuar o *backup* da aplicação *web* em produção, substituindo-a de seguida pela enviada pelo *Bitbucket*.

A *deployment pipeline* implementada permite que a cada *commit* no ramo *master* seja despoletado o processo de *build, tests, deploy* e *release* de forma completamente automática. O processo termina no servidor de produção, caso todo o procedimento seja executado sem a ocorrência de quaisquer erros. A existência de erros durante o processo despoleta também um alerta, enviado por e-mail, com o identificador do *commit* defeituoso.

```
#!/bin/bash

# Backup do diretório anterior
backup ()
{
 filename=$(date '+%Y%m%d_%H%M%S')_SA_Web.tar.gz;
 #tar cvzf ./backups/$filename -C /var/dotnetcore/SA/
 tar -C /var/dotnetcore -czvf ./backups/$filename SA

 if [ $? -ne 0 ]
 then
        echo "Erro a compactar diretorio"
        exit 1
 fi

 # Guardar o nome do ultimo ficheiro
 echo $filename > ./backups/lastbackup
}


deploy_service ()
{
 # Backup do servico em producao
 echo 'Deploying: Efetuar backup' > /dev/stderr
    backup

 # Parar Servico
 echo 'Deploying: Parar servico'  > /dev/stderr
 service kestrel-sa stop

 # Remover conteudo do diretorio
 echo 'Deploying: Remover conteudo anterior'  > /dev/stderr
 rm -Rf /var/dotnetcore/SA/* > /dev/null

 # Deploy do novo website
 echo 'Deploying: Copiar ficheiros novos' > /dev/stderr
 tar -xzvf ./toproduction/sa_web.tar.gz -C /var/dotnetcore/SA > /dev/null
 cp ./toproduction/appsettings.json /var/dotnetcore/SA > /dev/null

 # Remover ficheiro
 #rm sa_web.tar.gz
```

```
 # Start nginx
 echo 'Deploying: Start ao servico' > /dev/stderr
 service kestrel-sa start
}

restore_service ()
{
    # Parar Servico
 echo 'Restoring: Parar servico'  > /dev/stderr
 service kestrel-sa stop

 # Remover conteudo do diretorio
 echo 'Restoring: Remover conteudo anterior'  > /dev/stderr
 rm -Rf /var/dotnetcore/SA/*

 # Deploy do website que estava em producao
 echo 'Restoring: Restaurar ficheiros' > /dev/stderr
 cat  ./backups/lastbackup  |  xargs  -I{}  tar  -xzvf  ./backups/{}  -C
/var/dotnetcore/
 cp ./toproduction/appsettings.json /var/dotnetcore/SA > /dev/null

 # Start nginx
 echo 'Restoring: Start ao servico' > /dev/stderr
 service kestrel-sa start
}
…
```

Código 2 – *Bash script* de *publish/rollback* da aplicação *web*

5.1.3 Testes unitários

Segundo Edsger Dijkstra "*Program testing can be used to show the presence of bugs, but never to show their absence.*"

A utilização de testes de *software*, quer sejam eles unitários, de integração, aceitação ou outros, permite que sejam detetados defeitos no *software* construído. Numa primeira instância de construção de *software*, é de extrema importância a elaboração de testes unitários para a avaliação de funções básicas do sistema. O objetivo é que estes testes sejam isolados, pequenos e focados na funcionalidade e não em preocupações, como a lógica de negócio ou a infraestrutura. A sua utilização é particularmente útil na introdução de novas funcionalidades de *software,* mas também na alteração de funcionalidades já existentes. O aparecimento de erros em funcionalidades já existentes, nas quais os testes anteriormente passavam, poderá indicar a introdução de novos defeitos.

Com a utilização de ASP.NET Core MVC um dos componentes principais desta arquitetura são os *controllers* e é neles que deve ser depositada a atenção sobre o comportamento desejado para a aplicação. Com os testes unitários é possível testar o seu comportamento de modo a que possíveis defeitos de *software* não sejam colocados em produção (Steve Smith 2016).

Para a produção destes testes foi utilizada a ferramenta de testes xUnit, embora existissem outras como o MSTest ou nUnit para .NET Core.

O xUnit é uma ferramenta gratuita de testes, de código aberto e focada nas comunidades *online* para a .NET Framework.

Para a utilização de testes nesta ferramenta é necessário em primeiro lugar conhecer um pouco da forma como esta funciona. Esta assenta na utilização de dois tipos diferentes de testes, os Factos e as Teorias, devendo os métodos de testes possuir o atributo correspondente [Fact] ou [Theory].

Enquanto o atributo Fact tem por objetivo a realização de testes que sejam sempre verdadeiros, ou seja, testam condições invariantes, o atributo Theory é utilizado nos testes que apenas são verdadeiros para um conjunto particular de dados.

No exemplo a seguir, descrito pelo Código 3, testa-se o resultado do pedido de detalhes de um anúncio quando não é passado nenhum Id. O *controller* deverá retornar o código 404, ou seja, não encontrado.

```
[Fact]
public async Task DetailsReturnsNotFoundWhenIdIsNull()
{
      // Arrange
      var controller = new AnunciosController(null, null, null, null, null,
null);
      // Act
      var result = await controller.Details(null);
      // Assert
      var redirectToActionResult = Assert.IsType<NotFoundResult>(result);
      Assert.Equal(404, redirectToActionResult.StatusCode);
    }
```

Código 3 – Exemplo de teste a *controller*

5.1.4 Segurança

Uma das regras de ouro da segurança é não inventar segurança, isto é, embora se possa pensar que uma solução feita à medida, por não ser do conhecimento do público, seja a forma ideal de uma empresa se proteger, revela-se muitas vezes um desastre. Esta é uma prática completamente desaconselhada por comunidades *online*, como a *Open Web Application Security Project* (OWASP).

A segurança de uma solução informática deve ser uma preocupação transversal a todos os componentes que a constituem, devendo estar abrangidas nessa a confidencialidade, a integridade ou a disponibilidade.

Durante todo o processo de desenvolvimento houve uma grande preocupação com a segurança, tendo em vista tanto o processo em si como a solução final.

No que diz respeito ao processo de troca de informação com o repositório e o envio da solução para o servidor de qualidade, teve-se a preocupação de utilizar protocolos seguros como HTTPS, SFTP e SSH com o intuito de proteger a integridade e privacidade dos dados trocados.

Na sua plataforma, o Bitbucket possui uma ferramenta para geração de chaves SSH de modo a que o processo de autenticação em servidores remotos possa utilizar este método ao invés da utilização do método tradicional com *username* e *password*. Para isto basta gerar o par de chaves (pública e privada), copiar a chave pública para o servidor remoto e configurar as chaves autorizadas. Por fim é necessário também no Bitbucket adicionar o endereço do servidor remoto de forma a que seja capturado o *fingerprint* do servidor. Isto tem como finalidade proteger o acesso de ataques *man-in-the-middle*. A utilização deste mecanismo permitiu configurar a forma de comunicar entre o Bitbucket e o servidor de produção, sem que para isso tenha sido necessária a utilização de palavras-chave (Microsoft 2017a).

5.2 Aplicação Web

Pode dizer-se que a aplicação *web* é o *core* desta solução, pelo que todos os esforços foram no sentido da sua criação. Obviamente, e analogamente, um carro não anda, se apenas tiver o motor, no entanto esta é uma das peças fundamentais para o seu impulsionamento.

A utilização de *frameworks* maduras e tecnologicamente avançadas foi tida em conta por forma a ser criada uma aplicação baseada em boas práticas de engenharia sem, no entanto, descurar as mais recentes tendências e práticas no desenvolvimento de aplicações desta natureza.

Outro critério tido em conta foi a utilização de ferramentas que não comprometessem a viabilidade financeira do projeto, pelo que grande parte das tecnologias são *open-source* e podem ser utilizadas sem custos suplementares.

5.2.1 ASP.NET Core

Embora esta *framework* apenas tenha aparecido há relativamente pouco tempo, descende da .NET Framework com mais de 15 anos de história, tendo sido utilizada neste projeto por motivos relacionados com a sua flexibilidade, modularidade e por ser multiplataforma. De seguida é feita uma breve introdução às suas características e a aplicação de alguns módulos como o Identity.

O ASP.NET Core é uma *framework open-source* multiplataforma para a construção de aplicações *web*, mas também de aplicações para *Internet of Things* (IoT), *mobile* ou *on-premise*.

A sua arquitetura foi pensada e construída sob a forma de componentes modulares por forma a diminuir o *overhead*. Esta *framework* é uma evolução do ASP.NET que já conta com cerca de 15 anos de existência e milhões de programadores por todo o mundo, tendo sido redesenhada para a tornar mais leve e modular. (Microsoft 2017a)

Esta versão foi concebida com suporte para ser desenvolvida e executada tanto em Windows como Mac ou Linux.

Segundo a Microsoft (Microsoft 2017a), esta versão possui ainda algumas melhorias comparada com a versão ASP.NET:

- Construção de *user interfaces* (UI) e APIs web num só local;
- Integração de *frameworks client-side* modernas e *workflows* de desenvolvimento;
- Configuração do sistema baseado em ambiente preparado para *cloud;*
- *Dependency Injection* integrado;
- *Pipeline* de pedidos HTTP leve e modular;
- Capacidade de ser hospedado no IIS ou auto hospedado num processo próprio;
- Construído no .NET Core, que da aplicação;
- Pode ser executado no .NET Core, que dá suporte ao verdadeiro controlo de versões da aplicação lado-a-lado;
- Entregue inteiramente como *packages NuGet*;
- Adição de ferramentas que simplificam o desenvolvimento de aplicações *web* modernas;
- Desenvolver e executar aplicações ASP.NET Core multiplataforma em Windows, Mac e Linux;
- *Open-source* e focado na comunidade.

Como o ASP.NET Core é fornecido totalmente através de pacotes NuGet, permite que apenas sejam incluídos os pacotes estritamente necessários ao funcionamento das aplicações. Na aplicação criada tentou-se que apenas os módulos necessários fossem instalados, tal como o módulo referido à frente na secção "Autenticação e autorização". Esta modularidade permitiu reduzir o tamanho da aplicação e, por se tratar de um sistema modular, simplificar a sua manutenção.

A solução carece de um sistema de autenticação e autorização de acesso ao mesmo. Nesse sentido optou-se pela utilização de uma funcionalidade integrada com o .NET Core chamada de *Identity* (anteriormente *Membership*).

Autenticação e autorização – *Identity*

Esta funcionalidade disponibiliza já um conjunto de ferramentas de registo de utilizadores, de autenticação (*login/logout*) e autorização através de *claims* e *roles.* O *Identity* está preparado também para processos de recuperação de *password* e confirmação de registo de utilizador.

No entanto não são disponibilizadas funcionalidades de atribuição de *roles* a utilizadores, pelo que foi necessário implementá-las.

Para a aplicação optou-se por associar cada utilizador registado no sistema a um *role*, de modo a que todas as autorizações sejam baseadas nesta condição, podendo um utilizador pertencer

a mais do que um *role.* Assim é possível estender as permissões através da colocação dos utilizadores em mais do que um *role*.

Segurança

Também na implementação da solução foram tidas em consideração, questões relativas à segurança, nomeadamente a aplicação de mecanismos para a proteção contra os ataques do tipo *Cross-site request forgery* (também conhecido como XSRF ou CSRF).

Por forma a evitar a exploração deste ataque o .NET Core devolve ao cliente, juntamente com os formulários solicitados, um *token* que tem de ser enviado aquando da submissão da página. Um exemplo deste *token* poderá ser consultado de seguida no Código 4.

XSRF ou CSRF é um ataque contra aplicações *web* através do qual um *website* malicioso pode influenciar a interação entre o *browser* do cliente e o *website* que confia nesse browser. Estes ataques são possíveis porque os *browsers* enviam automaticamente alguns *tokens* de autenticação a cada pedido para o *website*. Esta forma de *exploit* é também conhecida como *one-click attack* ou *session riding*, porque o ataque tira proveito da sessão autenticada previamente pelo utilizador (Microsoft 2017c).

No âmbito do desenvolvimento de uma aplicação informática, uma prática aconselhada por especialistas de segurança é a utilização de *frameworks*, ferramentas e técnicas amplamente utilizadas, testadas e de confiança.

Nesse sentido, para uma proteção eficaz contra este tipo de ataque, várias recomendações são sugeridas pelo *website OWASP*, nomeadamente a verificação dos *headers* do protocolo HTTP e a utilização de *tokens* de validação.

Um ponto de partida para a defesa contra este ataque é a verificação dos *headers* "*Origin header*" e "*Referer header*" do pedido HTTP.

No caso de o "*origin header*" estar presente, é avaliado se o *website* do cabeçalho corresponde à sua origem. Se não estiver presente, pode verificar-se o "*referer header*" onde é possível verificar se o *hostname* corresponde ao *website* origem.

Poderá ainda verificar-se que nenhum dos dois está presente, ou que é simples fazer *spoofing* aos *headers* pelo que estas medidas são apenas uma primeira fase de proteção não sendo totalmente eficazes.

A abordagem mais comum para a defesa contra os ataques do tipo CSRF é a utilização de *synchronizer token pattern* (STP), sendo esta uma técnica utilizada quando o utilizador solicita uma página que contenha um formulário. O servidor envia um *token* associado à identidade do utilizador corrente para o cliente. O cliente envia o *token* de volta para verificação do servidor, caso este receba um *token* que não corresponda à identidade do utilizador autenticado, então o pedido é rejeitado. Este *token* é único e imprevisível. É ainda possível usar esta técnica para

assegurar a sequência de uma série de pedidos (assegurando que a página um precede a página dois e por aí em diante) (Osterwalder et al. 2010).

No trecho de código a seguir, identificado por Código 4, podemos ver um exemplo de como este *token* é transmitido para o cliente e como é guardado de forma temporária através de um *input field* escondido.

```
<input name="__RequestVerificationToken" type="hidden"
value="CfDJ8PeYaecib15Dk8o7a1XGf4dCHJNWp5hVO8mDfGc-
JJnnQRdWV4GzBrfKp7GCc_sFk3BhYORvhnOmyyFLV4FGRTIgSNv4qip7LFS6j1hXa6oSuz4-
rlE7JjKuezceQkp7lkZ6szhDi5uA_RtuZY6xea0GajXTezw9GEDF2rlRGrILx-
wNikFhtOIGWiISaUebAg">
```

Código 4 – Exemplo de *token* gerado pela *framework*

5.2.2 Kestrel Web Server e *reverse proxy*

Para a colocação de um *website online* é necessária a existência de uma aplicação servidora (*web server)* que forneça as páginas pedidas pelos utilizadores. A componente da aplicação *web* deste projeto tem a particularidade de utilizar a *framework* ASP.NET Core, sendo necessária a utilização de um *web server* que a suporte.

É ainda crucial que uma aplicação deste género, ligada à internet, esteja protegida por uma configuração que lhe permita responder a questões relativas à segurança, balanceamento de carga, limitação de pedidos ou até mesmo limitação da largura de banda utilizada.

Kestrel é um *web server* multiplataforma para .NET Core (Sourabh Shirhatti 2017), baseado na biblioteca *libuv* que tem como foco o *input*/*output* (I/O) assíncrono que apesar de poder servir conteúdo ASP.NET não é tão rico em funcionalidades como o IIS, Nginx ou Apache. Estes, por sua vez, podem ser responsáveis por diversos trabalhos, como: servir conteúdo estático, fazer *cache* e compressão de pedidos e utilização de SSL no servidor HTTP. Um servidor *reverse proxy* pode ainda estar alojado numa máquina dedicada ou ser colocado ao lado de um servidor HTTP (Microsoft 2017b).

Para a versão ASP.NET Core 1.x, utilizada neste projeto, a configuração aconselhada pela Microsoft é a inclusão de um *reverse proxy* tal como ilustrado na Figura 22.

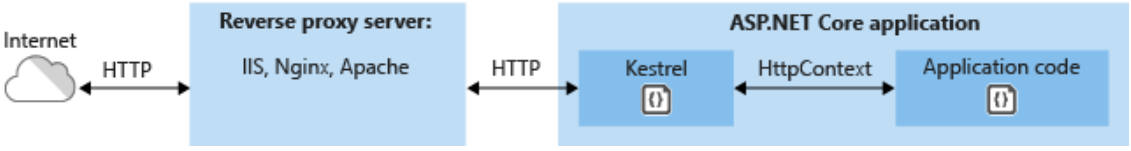

Figura 22 – *Reverse proxy* e *Kestrel web server*

Devido à utilização do sistema operativo Ubuntu, tanto no servidor de desenvolvimento como no de qualidade, apenas os *web servers* Nginx e Apache poderiam ser utilizados, pois o IIS apenas está disponível em sistemas operativos Windows. Assim, e por não existirem diferenças significativas, para o grau de desempenho esperado pelo sistema, optou-se pela utilização do *web server* Nginx.

Na configuração do *web server* foram aplicados certificados de segurança que possibilitam a cifra dos dados transmitidos entre o servidor e os clientes, tornando assim as comunicações mais seguras, tal como solicitado nos requisitos do sistema. Nas configurações foram ainda aplicados diversos *Headers* HTTP, cujo objetivo se prende com a proteção da solução contra diversos ataques conhecidos, nomeadamente ataques de Cross-site Scripting (XSS). De seguida são apresentadas os *Headers* implementados.

Headers HTTP

Os *headers* HTTP, tal como descritos pela OWASP (OWASP 2016), permitem ao cliente e ao servidor passarem informação juntamente com um pedido ou resposta. Um *header* de pedido consiste no seu nome insensível a maiúsculas seguido de dois pontos ":" e de seguida pelo seu valor sem quebras de linha. Os *headers* foram inicialmente definidos no RFC 4229 mantido pela *Internet Assigned Numbers Authority* (IANA), que gere ainda o registo de novos pedidos de *headers* HTTP.

Os *headers* podem ser agrupados segundo os seus contextos:

- *Header* genérico: *Headers* aplicáveis tanto a pedidos como a respostas, mas sem relação entre os dados eventualmente transmitidos no corpo;
- *Header* de pedido: *Headers* que contêm informação relativa ao recurso a obter ou sobre o próprio cliente;
- *Header* de resposta: *Headers* com informação adicional sobre a resposta, como a sua localização ou informação sobre o servidor como o nome ou versão;
- *Entity header*: Headers que contêm mais informação sobre o corpo da mensagem, tais como o seu tamanho ou o seu *Multipurpose Internet Mail Extensions* (MIME) type.

Com vista a proteger a aplicação de possíveis ataques foi definido um conjunto de *headers* HTTP de resposta, tendo sido configurado no *reverse proxy Nginx*, tal como descrito na Tabela 9. Deste modo todos os pedidos feitos por HTTPS conterão as configurações referidas.

Tabela 9 - *Headers* HTTP de resposta definidos no *Nginx*

Header	Valor
X-Frame-Options	DENY
Strict-Transport-Security	max-age=63072000; includeSubdomains;
X-Content-Type-Options	nosniff
X-XSS-Protection	X-XSS-Protection: 1; mode=block
Content-Security-Policy	add_header Content-Security-Policy "default-src 'self'; script-src 'self' 'unsafe-inline' 'unsafe-eval' https://ajax.aspnetcdn.com https://www.google-analytics.com https://maxcdn.bootstrapcdn.com https://code.jquery.com https://cdnjs.cloudflare.com; img-src 'self' data: http://www.w3.org https://www.google-analytics.com https://s-static.ak.facebook.com; style-src 'self' 'unsafe-inline' https://fonts.googleapis.com https://maxcdn.bootstrapcdn.com; font-src 'self' data: https://themes.googleusercontent.com; frame-src https://www.facebook.com https://s-static.ak.facebook.com; object-src 'none'";
Referrer-Policy	no-referrer-when-downgrade

5.2.3 Entity Framework Core

"O Entity Framework (EF) Core é um *object-relational mapper* (ORM) que permite aos programadores .NET utilizarem uma base de dados através da utilização de objetos .NET. A sua utilização elimina a necessidade de criação de código de acesso a dados que habitualmente os programadores necessitam de escrever, dando suporte a diversos motores de base de dados." (Microsoft 2016a)

"Com o EF Core, o acesso a dados é feito através de um *model,* que é composto por classes de entidade e um contexto que representa a sessão com a base de dados, permitindo consulta e gravação de dados." (Samuel Gibbs 2016)

As possibilidades para a criação do referido *model* são, nesta versão, através da utilização de duas abordagens diferentes. A abordagem que permite criar o modelo através de uma base de dados existente (*Database-First*) e a abordagem que permite através do *model* criar a base de dados (*Code-First*). Esta última permite a utilização de *EF Migrations* que, através do mapeamento existente com a base de dados, permite efetuar atualizações ao *schema*.

A utilização desta *framework* traz um conjunto de mais-valias a este projeto, nomeadamente na fiabilidade, agilidade e abstração da complexidade necessária nos processos atribuídos à persistência de dados.

A construção da base de dados com a utilização de *Entity Framework*, na sua versão 7 (ou Core 1.0) pode seguir, tal como dito anteriormente duas abordagens diferentes: *Database-First ou Code-First*. Em versões anteriores do *Entity Framwork* podia seguir-se também a abordagem

Model-First onde era possível, através de uma ferramenta de *design* construir o *model* que permitia gerar os *scripts* com as entidades, relações e constrangimentos para aplicação na base de dados e gerar as classes entidade *Plain Old CLR Object* (POCO) da aplicação. No entanto, esta funcionalidade foi descontinuada na sua versão Core.

Database-First

Tipicamente a abordagem *Database-First* é utilizada quando a base de dados já existe, quando é construída em separado ou por questões relacionadas com a facilidade de criação de SQL por parte de quem desenvolve a solução. Alguns SGBDs possuem inclusive ferramentas gráficas de construção de diagramas, abstraindo o programador da linguagem de definição de dados (LDD). São exemplo dessas ferramentas o SQL Management Studio da Microsoft ou o MySQL Workbench da Oracle.

Para a utilização desta abordagem o Entity Framework executa um processo de *reverse engineer* que consulta a base de dados. Em função das suas entidades gera, baseado em *templates*, as classes entidade, bem como o contexto a ser utilizado para mapear estas entidades.

Code-First

Tal como na abordagem anterior, para trabalharmos com Entity Framework é necessário que seja criado um contexto (DbContext) mapeando as entidades da base de dados e as classes entidade da aplicação. Nesta abordagem todo o código construído é criado pelo programador em vez de gerado automaticamente, o que acaba por trazer um maior controlo sobre o mesmo.

Habitualmente a abordagem *Code-First* é utilizada quando ainda não existe uma base de dados ou se está a lidar com uma base de dados vazia. Existe ainda a possibilidade de se recorrer, tal como na abordagem anterior, a um processo de *reverse engineer* para a criação das primeiras classes entidade e respetivo contexto.

Findo este processo poderá recorrer-se às EF Migrations por forma a que mudanças ao *model* possam produzir as respetivas atualizações de *schema* na base de dados.

A utilização da abordagem *Code-First* mostrou-se ser a mais indicada para este projeto por se tratar de uma solução construída de raiz e se pretender personalizar as entidades em código e não na base de dados.

Migrations

Com a evolução da solução é frequente a necessidade de inserção, modificação ou até remoção das entidades do modelo, sendo, por consequência, necessária também a atualização da estrutura de persistência de dados.

Para os casos referidos poderá recorrer-se à funcionalidade EF Migrations, que através do Entity Model inicial e um *snapshot* da base de dados, consegue verificar quais as diferenças existentes e aplicar os respetivos *upgrades* ao *schema* da base de dados.

Ao gerir automaticamente as alterações necessárias a serem aplicadas na base de dados, esta funcionalidade agiliza um processo habitualmente bastante sujeito a erro. A equipa de desenvolvimento delega assim a responsabilidade, tornando este processo mais fiável.

5.2.4 Interface gráfico

É sabido que nos últimos anos tem havido uma massificação de dispositivos móveis, sendo estes amplamente utilizados na navegação *web*, tendo já ultrapassado a navegação em computadores de secretária (Ethan Marcotter 2010). A oferta de produtos ou serviços adaptados às necessidades dos clientes é cada vez mais uma preocupação por parte das empresas. Tendo isto em consideração, as empresas têm optado pela disponibilização dos seus *websites* adaptados aos dispositivos dos seus clientes.

A construção de *designs* específicos para cada tipo de dispositivo, sendo disponibilizados sobre a forma de subdomínios do próprio *website* é uma das práticas aplicadas. Um exemplo atual desta realidade, é a rede social *facebook*, que para os dispositivos móveis disponibiliza o subdomínio https://m.facebook.com. No entanto, grande parte dos *websites* não estão otimizados para dispositivos como *smartphones*, *tables*, *tv* ou até *wearables*.

Com o aparecimento da versão 2.1 do *Cascading Style Sheets* (CSS) surgiram os *media types*, que permitem a seleção de CSS para os dispositivos ou funções para os quais foram solicitados (Ethan Marcotter 2010). Através da utilização de *media types* como "*screen*", "*tv*", "*print*", "*braille*" podemos tomar decisão do que mostrar e de como mostrar.

Mais tarde, com o aparecimento da especificação do CSS3 passa a ser possível a utilização de *media queries*, tornando possível ter a perceção de algumas medidas do tipo de dispositivo onde as páginas *web* correm, como a resolução e *viewport* (área disponibilizada para visualização da página) (Ethan Marcotter 2010).

```
<link rel="stylesheet" type="text/css"
   media="screen and (max-device-width: 480px)"
   href="shetland.css" />
```

Código 5 - Inclusão de CSS com *media types*

No caso do exemplo descrito acima, a utilização do *stylesheet shetland.css* só será aplicada se o *media type* for "*screen*" e se o ecrã onde está a ser apresentado tiver uma resolução até 480px (Saeed 2016).

A aplicação deste tipo de abordagem vem então permitir a adaptação de imagens, *grids* ou outros elementos de acordo com o dispositivo, criando páginas mais dinâmicas e abrangentes.

Sabendo isto, é importante, desde o início, a aplicação de estratégias que permitam abranger o maior número de dispositivos, proporcionando uma experiência mais agradável ao utilizador. Nesse sentido optou-se pela utilização de uma filosofia de desenvolvimento *Mobile first* de modo a que a informação seja adaptável em primeiro lugar a dispositivos com menor dimensão e de seguida aos restantes. Para lidar com este tipo de problemáticas recorreu-se à *framework* de *front-end bootstrap*.

5.2.5 Correspondência de anúncios

A correspondência entre anúncios é uma das peças chave na construção do sistema, esperando-se que seja também um dos elementos responsáveis pelo seu sucesso. Esta correspondência baseia-se na utilização de um algoritmo que cruza a informação constante nos anúncios com a informação obtida através do envio das fotografias para o sistema de ML.

A existência de características que possam identificar de forma unívoca um determinado animal é fundamental para obter a correspondência direta dos anúncios. A utilização do campo de *Microchip* permitirá fazer essa correspondência de modo automático. Nos restantes casos, é necessário proceder à correspondência dos anúncios através das características dos animais, que são especificadas no momento da submissão dos anúncios, bem como a utilização de informação obtida na API de visão computacional utilizada.

Sempre que existe um anúncio oposto ativo, em que as datas dos eventos sejam válidas – data do evento do desaparecimento menor do que a data do aparecimento – cuja espécie do animal e *Microchip* correspondam, o anúncio é automaticamente dado como correspondente, sem que o algoritmo prossiga para a restante avaliação.

Por outro lado, se a informação não for suficiente para fazer corresponder os anúncios é executada uma segunda avaliação através dos dados constantes nos mesmos e a informação vinda da API de visão computacional sobre as fotografias do animal. A correspondência destas características não pode ser feita de forma direta, isto é, por exemplo: se desaparecer um animal com uma trela, o algoritmo não pode apenas filtrar apenas os animais encontrados com trela. O risco de uma má avaliação de uma característica do animal por parte do anunciante, bem como o risco associado à perda ou remoção de características durante a ausência do animal poria em causa todo o processo de correspondência.

Para ser mais fácil perceber, de seguida são dados dois exemplos simples sobre o funcionamento do sistema à receção de anúncios.

Animal desaparecido

Aquando da submissão de um anúncio de um animal desaparecido, o sistema procurará na base de dados todos os anúncios de animais encontrados e no caso de ter um grau de certeza maior ou igual a 70% irá contactar o anunciante, tal como esquematizado na Figura 23.

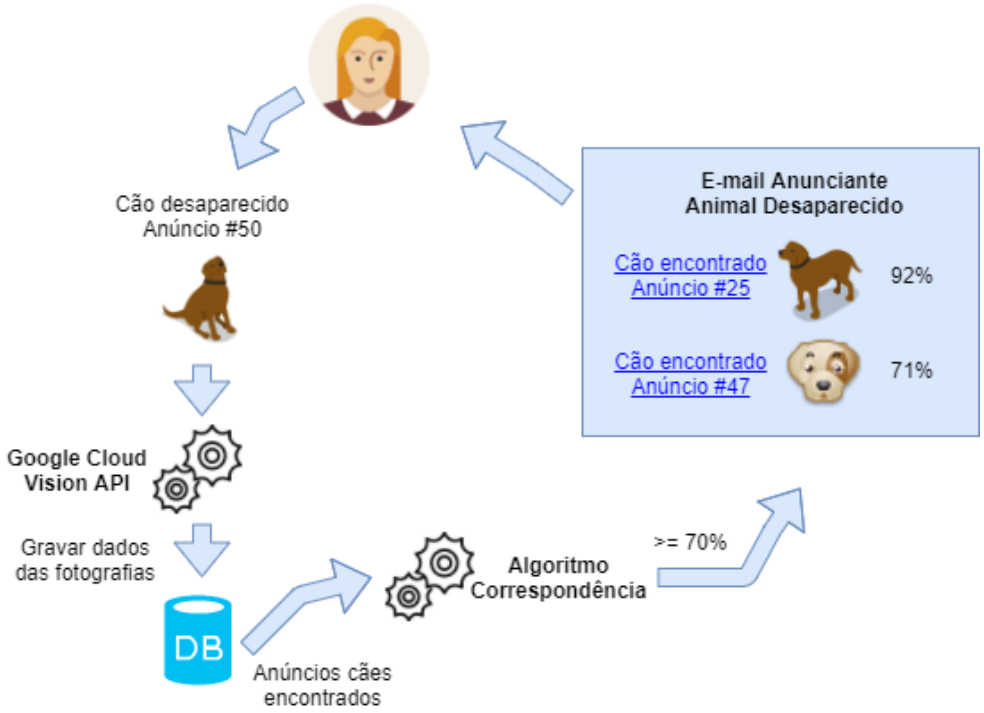

Figura 23 – Processo de correspondência aquando da submissão de animais desaparecidos

Animal encontrado

Por outro lado, caso seja submetido um anúncio de um animal encontrado, o sistema procurará na base de dados, os animais desaparecidos que possam corresponder e enviará para cada anunciante de um animal desaparecido um *e-mail* com a sugestão. Esta situação é ilustrada de seguida pela Figura 24.

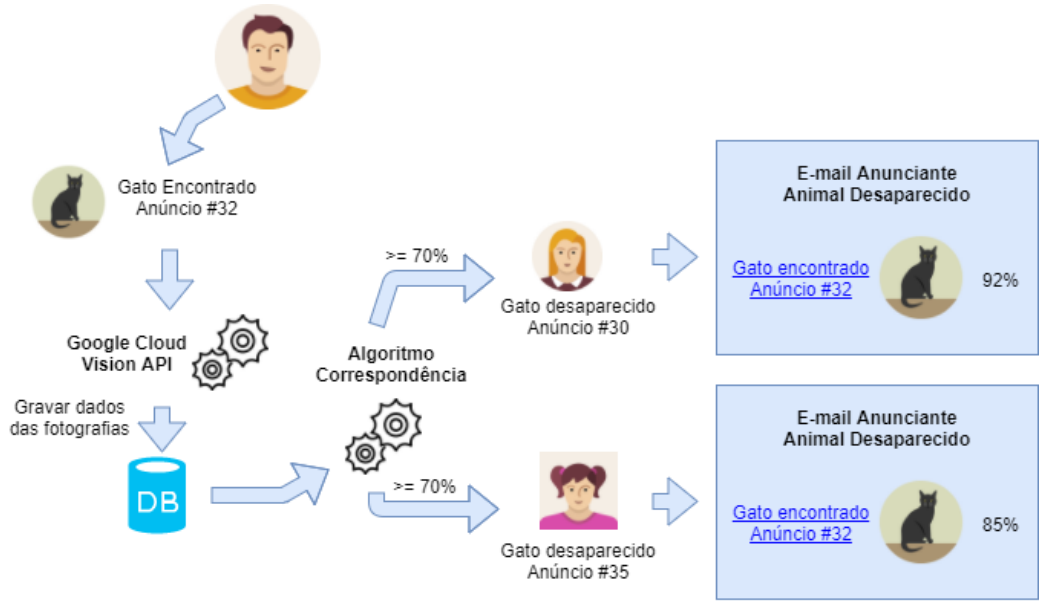

Figura 24 – Processo de correspondência aquando da submissão de animais encontrados

Tal como visto anteriormente, na secção de Métodos de identificação e na Identificação de características físicas e comportamentais do Capítulo 2, existem diferentes métodos e formas de identificação de animais. Estes, foram agora unidos em categorias, tendo-lhes sido atribuído um peso de acordo com a exatidão que se pensa que os dados terão, de forma a que possam ser usados no algoritmo da solução para calcular o grau de certeza de correspondência de anúncios.

Nesse sentido, a avaliação é feita através da atribuição de pesos a cada uma das características preenchidas pelos anunciantes e obtidas pela API de visão computacional, tendo como objetivo minimizar o risco de exclusão de anúncios válidos como correspondentes.

Os valores aqui definidos podem ser alterados na aplicação sempre que necessário, pelo que o algoritmo utiliza esses mesmos valores de forma completamente dinâmica.

Tal como referido, na Tabela 10 são definidos quais os pesos de cada categoria, de acordo com o que se espera serem as características cujo preenchimento será mais correto, tendo em conta também que há características que poderão deixar de existir, ou poderão ser muito variáveis com o passar do tempo.

Tabela 10 – Pesos atribuídos às características dos anúncios

Categoria	Peso
Características físicas	70%
Características comportamentais	5%
Localização	15%
Outras características	10%

De seguida serão apresentadas as características físicas utilizadas pelo algoritmo para o cálculo de correspondência de anúncios.

Características físicas

A primeira categoria pretende agrupar um conjunto de características físicas do animal, tanto inerentes ao próprio animal como as provocadas por ação humana, como por exemplo os entalhes/cortes de orelha ou o corte de caudas. Essencialmente, estas características poderão ser observadas de forma direta pelo anunciante. Na Tabela 11 é possível consultar estas características e verificar de que forma é que o seu peso influencia o algoritmo. O peso de cada categoria é calculado neste caso através da divisão do peso da categoria (70%) pelo número de elementos que a constituem.

Tabela 11 – Pesos atribuídos às características físicas do animal

Característica	Opções	Condições
Raça	Raças existentes na aplicação	Raça definida pelo ML – % total Raça definida pelo utilizador – 70%
Tamanho	Mini, pequeno, médio, grande, gigante	Se igual – % total Se valor adjacente – 50% do Peso
Idade	Cria, jovem, adulto, sénior	Se igual – % total Se valor adjacente – 50% do Peso
Pelo	Curto, médio, longo	Se igual – % total Se valor adjacente – 50% do Peso
Padrão Pelo	Atartarugado, bicolor, listas, liso, malhado, pintas, tigrado, tricolor	Se igual – % total
Sexo	Macho Fêmea Não sabe	Correspondência - Total
Cores	Cores definidas na aplicação	Se igual – % total Outro Match: 0%
Cauda	Curta, média, longa	Se igual – % total
Cauda Cortada	S/N	Se igual – % total
Orelha	Curta, média, longa	Se igual – % total
Orelhas Cortadas	S/N	Se igual – % total
Esterilizado	S/N/I	Se igual - % total Se indefinido – 0%

Apesar de se esperar que os responsáveis por anúncios de animais desaparecidos especifiquem sempre as raças dos seus animais, o mais comum será a inexistência ou raros os casos em que isso acontecerá por parte dos anunciantes de animais encontrados.
Aqui, entra então a avaliação das fotografias dos animais para que se tente encontrar a raça do animal presente no anúncio.
A Cloud Vision API foi a ferramenta que obteve melhores resultados durante a avaliação de ferramentas para o efeito, pelo que foi também utilizada para esta solução.

Cloud Vision API

No caso de o anunciante de animal encontrado não especificar a raça do animal, as fotografias do anúncio serão então submetidas à API de avaliação de conteúdo de imagens para se tentar apurar essa característica. Os resultados da avaliação serão então lidos e com base nas raças de animais desaparecidos, filtrados os resultados. Por poderem ser retornadas raças semelhantes, o que o algoritmo faz é o somatório das percentagens de cada raça presente, sendo escolhida no final a raça que possuir maior percentagem no conjunto de fotografias avaliadas.

Apesar das características físicas terem um grande peso na avaliação de correspondência dos anúncios, também outras características podem ser verificadas pela observação do animal, como é o caso das características comportamentais.

Características comportamentais

Além das características físicas do animal, pretende-se que também os aspetos comportamentais possam ser também especificados pelos anunciantes. Na Tabela 12 são enumeradas as características disponíveis. Todas as características possuem o mesmo peso dentro desta categoria.

Tabela 12 – Características comportamentais do animal

Característica	Opções
Agressividade	S/N
Docilidade	S/N
Adestramento	S/N
Hiperatividade	S/N
Apatia	S/N
Medo	S/N

Para além das características já enunciadas, há ainda fatores muito importantes para a análise de correspondência dos anúncios, como é o caso do número de quilómetros que distam os anúncios.

Distância entre anúncios

Para além das características já enunciadas, o algoritmo ainda tem em consideração a distância entre os anúncios. Isto é, por exemplo, se tiver desaparecido um cão no Algarve, é pouco provável que apareça em Bragança. Na Tabela 13 podem ser consultadas as condições para atribuição do peso total no âmbito desta categoria.

Tabela 13 – Peso atribuído à distância entre os anunciantes

Característica	Opções	Peso	Condições
Distância	N/A	15%	Até 1 km – 100%
			Até 5 km – 50%
			Até 10 km – 20%
			Mais de 10 km - 0%

Existem ainda características associadas a objetos colocados nos animais e que também poderão ser um fator de apoio na decisão de correspondência. Nesse sentido são apresentadas de seguida outras características que também serão levadas em consideração na avaliação de correspondência.

Outras características

Por fim, o algoritmo tem ainda em consideração outras características, associadas a acessórios ou objetos que muitas vezes os animais possuem no momento do seu

aparecimento/desaparecimento. A volatilidade das características desta categoria justifica o menor peso atribuído.

Tabela 14 – Peso atribuído a outras características

Característica	Opções	Peso	Condições
Trela	S/N	1%	
Coleira	S/N	2%	
Medalha	S/N	4%	Com o mesmo nome – 4% Nome diferente – 0%
Açaime	S/N	1%	
Roupa	S/N	1%	
Arnês	S/N	1%	

5.2.6 Partilha de anúncios em redes sociais

A utilização das redes sociais é hoje em dia uma realidade, pelo que estas ferramentas são uma forma amplamente utilizada para fins de divulgação de informação. A partilha dos anúncios de animais nestas redes, tanto desaparecidos como encontrados, é uma funcionalidade fundamental para a divulgação tanto da plataforma, como do caso em questão.

Para recorrer a esta funcionalidade é necessário que a plataforma esteja preparada para a partilha das suas páginas nas redes sociais, isto é, que possua identificadores (*meta tags*) sobre conteúdos como o título, a descrição ou as fotografias colocadas nos anúncios.

As redes sociais como o Facebook, Twitter ou Google+ possuem algoritmos capazes de ler as *meta tags* dos *websites* e assim produzir uma apresentação formatada do que se quer partilhar. No entanto é necessário utilizar protocolos próprios para a construção destas *meta tags*. A rede social Facebook implementou o seu próprio protocolo para a identificação de conteúdos, o Open Graph Protocol, tendo-se tornado um standard aberto (Saeed 2016).

Para a utilização deste protocolo, existem algumas propriedades que são obrigatórias e deverão ser colocadas dentro do elemento <head> da página, de forma a que possa ser criado um objeto *graph*. As propriedades obrigatórias são:

- og:title – O título do objeto para ser apresentado ex. “Cão desaparecido, caso #20031”;
- og:type – O tipo de objeto que estamos a instanciar ex. “website”;
- og:image – O URL de uma imagem que se pretenda associar;
- og:url – O URL para o objeto *graph*, que será usado como seu identificador permanente.

Existem ainda outras propriedades opcionais que poderão ser consultadas no *website* oficial deste protocolo The Open Graph protocol (www.ogp.me) e que complementarão a informação obrigatória.

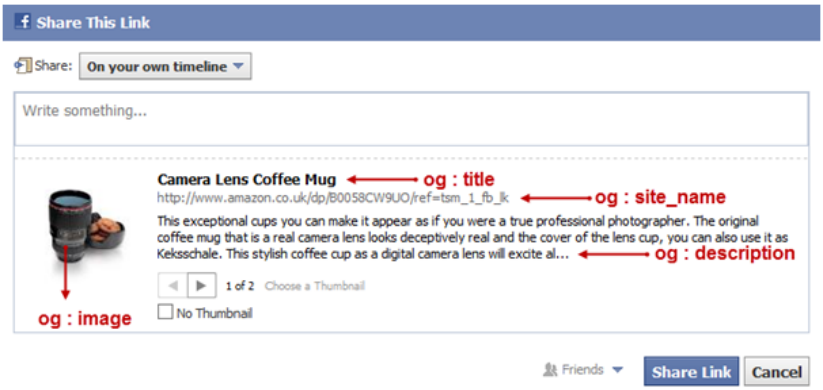

Figura 25 – [12] Facebook - Exemplo de recolha de informação de *metatags*

A rede social Twitter tem, por seu lado, uma forma idêntica para capturar as propriedades da página através do que esta chama de Twitter Cards. Trata-se, à semelhança do OGP, de um conjunto de *meta tags* que permitem identificar conteúdo a partilhar.

Para utilização dos Twitter Cards é também obrigatória a definição de um conjunto de propriedades que pode ser utilizada em conjunto com o OGP de forma a que não haja duplicação de *tags*. O exemplo do Código 6 reflete a conjugação de ambos os protocolos na identificação dos meta dados da página.

```
<meta name="twitter:card" content="summary" />
<meta name="twitter:site" content="@nytimesbits" />
<meta name="twitter:creator" content="@nickbilton" />
<meta  property="og:url"  content="http://bits.blogs.nytimes.com/2011/12/08/a-
twitter-for-my-sister/" />
<meta property="og:title" content="A Twitter for My Sister" />
<meta property="og:description" content="In the early days, Twitter grew so
quickly that it was almost impossible to add new features because engineers
spent their time trying to keep the rocket ship from stalling." />
<meta                                                    property="og:image"
content="http://graphics8.nytimes.com/images/2011/12/08/technology/bits-
newtwitter/bits-newtwitter-tmagArticle.jpg" />
```

Código 6 – [13] Exemplo de conjugação de Open Graph Protocol com Twitter Cards

A utilização destas duas formas de identificação de meta dados das páginas permite, por se tratar de algo já proliferado, que sejam utilizados também noutras redes sociais, como o Google+, o Pintrest e outros (Microsoft 2017b).

[12] Imagem retirada de https://rehansaeed.com/social-taghelpers-for-asp-net-core/
[13] Exemplo retirado de https://dev.twitter.com/cards/getting-started#started

5.2.7 Multilíngue

Na criação de produtos ou serviços, é necessário ter em consideração o mercado alvo para o qual são criados ou desenvolvidos. Apesar de o mercado alvo da aplicação a desenvolver ser neste momento essencialmente o mercado nacional, é importante à partida dotar a aplicação de tecnologias que possibilitem a sua internacionalização. Com a visita de milhares de turistas ao nosso país todos os anos, existe uma grande probabilidade de estes trazerem os seus animais de companhia e consequentemente também, existirem casos de desaparecimento no nosso território.

A pensar nisso, foi utlizada uma estratégia de utilização de *Resource files* para a tradução de termos, botões, títulos e outras *strings* que não constem da base de dados. Neste momento apenas foram criados dois *Resource files* partilhados, um para língua portuguesa e outro para língua inglesa.

O menu superior da página possui alguns termos que são traduzidos mediante a língua escolhida, tal como ilustrado nas Figura 26 e Figura 27.

Figura 26 – Menu de navegação em português

Figura 27 – Menu de navegação em inglês

5.3 Outras tecnologias utilizadas

Além das tecnologias anteriormente abordadas, que fazem parte dos processos *core* da solução, existem ainda outras que complementam a solução. São exemplos disso o WSO2 Integration que implementa um ESB permitindo o uso de mensagens entre o componente da aplicação *web* e os componentes externos; e também a API utilizada para fazer a avaliação do conteúdo das fotografias dos anúncios.

5.3.1 WSO2 Integration

A utilização do WSO2 Integration pretende dar parte da resposta aos princípios de baixo acoplamento e alta coesão da engenharia de *software* para a criação de uma arquitetura eficiente e sustentável, bem como aos requisitos de fiabilidade do sistema na entrega de mensagens, tanto às APIs externas como no envio de *e-mails*.

Como referido anteriormente, procurou-se que fossem utilizadas tecnologias gratuitas ou de baixo custo, de forma a poder limitar-se os gastos com o sistema. Com o intuito de cumprir com as boas práticas de engenharia, e com as limitações e requisitos do sistema, foi selecionada esta ferramenta que é disponibilizada gratuitamente e que é apresentada de seguida juntamente com a API criada para o envio de *e-mails*.

Segundo a própria empresa (WSO2 2017), o WSO2 é um Enterprise Service Bus totalmente *open source* compatível com os principais *standards* utilizados por *Web Services* como WF-*, SOAP, e REST. Possui um motor de mensagens baseado em padrões que permite utilizar o valor do envio de mensagens sem que para isso seja necessário escrever-se código.

De entre as inúmeras funcionalidades disponibilizadas pelo WSO destacam-se, no âmbito do trabalho desenvolvido nesta dissertação, a gestão de processos de negócio de longa duração com estado e a capacidade de *message broker* que pode ser utilizado para a entrega de mensagens de forma confiável (WSO2 2017).

Permite a integração de serviços *cloud*, *software* legado e armazenamento de dados, possibilitando a transformação de dados de forma transparente em diferentes formatos e tipos de protocolos de comunicação (Rodriguez 2016).

***Mail* API**

Tal como já referido anteriormente, a entrega de mensagens é algo crucial para o sucesso do sistema. Nesse sentido começou-se por aplicar os padrões *Dead Letter Channel* e *Guaranteed Delivery* no serviço de envio de *e-mail* da aplicação.

Para o efeito, foi criada uma API REST que através das propriedades definidas no *payload* da mensagem, envia *e-mails* para os respetivos destinatários. Sempre que a API recebe uma mensagem coloca-a na fila (*queue*) Java Message Service (JMS). O JMS Proxy está à escuta desta fila e sempre que deteta a chegada de uma nova mensagem processa-a. No caso da existência de algum erro na entrega, é feito um *Rollback* sendo a mensagem devolvida à *queue* anterior. Neste processo verifica-se a aplicação do padrão *Guaranteed Delivery.* Por outro lado, este processo não deve ser infinito, pelo que se define um número máximo de tentativas até que as mensagens passem para um outro canal, o *Dead Letter Channel*.

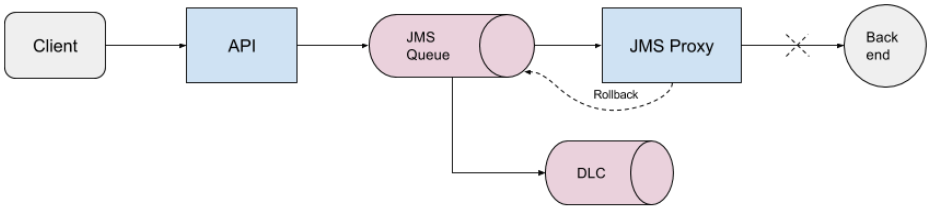

Figura 28 - [14] Aplicação dos padrões Dead Letter Channel e Guaranteed Delivery

[14] Imagem retirada de http://blog.lasindu.com/2016/07/how-to-expose-email-sending-api-with_6.html

No Código 7 é apresentada a Mail API, responsável pela receção de pedidos de envio de mensagens e colocação na JMS Queue. Do código apresentado foram retirados alguns eventos de *log*, bem como as propriedades de retorno de erros.

```
<api context="/email" name="EmailAPI" xmlns="http://ws.apache.org/ns/synapse">
    <resource methods="POST" protocol="http">
        <inSequence>
            <!-- Eventos de log -->
            <property     name="destQueue"     scope="default"     type="STRING"
value="MailQueue"/>
            <property   name="FORCE_SC_ACCEPTED"   scope="axis2"   type="STRING"
value="true"/>
            <property     name="OUT_ONLY"     scope="default"     type="STRING"
value="true"/>
            <property action="remove" name="REST_URL_POSTFIX" scope="axis2"/>
            <header  expression="fn:concat('jms:/',  get-property('destQueue'),
'?transport.jms.ConnectionFactoryJNDIName=QueueConnectionFactory&amp;java.nami
ng.factory.initial=org.wso2.andes.jndi.PropertiesFileInitialContextFactory&amp
;java.naming.provider.url=conf/jndi.properties&amp;transport.jms.DestinationTy
pe=queue')" name="To" scope="default"/>
            <send/>
        </inSequence>
        <outSequence/>
        <faultSequence>
            <-- Definição de propriedades para retorno de erros -->
        </faultSequence>
    </resource>
</api>
```

Código 7 – API para envio de *e-mails*

A *mail queue,* por sua vez, quando recebe uma mensagem, utiliza um *template* para a formatação da mesma, passando-a ao JMS Proxy que se encontra à escuta. Caso não tenha sucesso no envio da mensagem, é tentado a cada 5 segundos o seu reenvio, por no máximo 15 vezes. Caso não tenha sucesso é colocada a mensagem no canal *Dead Letter Channel* para ser tratado de forma manual.

Com a aplicação deste ESB foi possível garantir a entrega de mensagens, bem como dotar o sistema de um componente permeável à mudança e à escalabilidade.

5.3.2 Google Cloud Vision API

De modo a proceder-se à avaliação das fotografias enviadas para o sistema, foi utilizada a Google Cloud Vision API, que se mostrou ser a mais eficaz nos testes efetuados para a deteção de raças de animais.

Pretende-se então apresentar o funcionamento desta API referindo-se quais os custos inerentes à sua utilização, quais as questões de segurança, qual o *interface* da API e quais as opções utilizadas no âmbito deste projeto.

Em primeiro lugar é necessário proceder-se ao registo na *Google Cloud Platform* onde são disponibilizados os serviços da Google. Associado a este registo são atribuídos 300$ para serem gastos em serviços de base de dados, aluguer de máquinas virtuais, alojamento de aplicações entre outros (como a própria Cloud Vision API). Para utilização da Cloud Vision API em específico, a Google oferece planos gratuitos até 1000 pedidos mensais, pelo que se pensa que sejam suficientes numa primeira fase do projeto. Acima desse limite de pedidos, se assim se pretender poderá ser utilizado através do pagamento de 1.50$ a cada 1000 pedidos.

Em cada pedido à API é necessário adicionar um identificador único de forma a que a Google consiga associar o pedido feito a um determinado projeto e desta forma conceder autorização aos pedidos. O intuito da utilização destas chaves é ainda controlar o número de pedidos de cada cliente de forma a poder cobrar sobre a sua utilização em caso de ser excedida a quota atribuída.

Para aplicações que utilizam o protocolo OAuth 2.0 é possível gerar um *token* de acesso, contendo um identificador único do pedido. Não usando este protocolo, é possível ainda a utilização de uma chave (API key), que é gerada na consola de administração do Google Cloud Platform e que deve ser incluído na *query string* de cada pedido à API.

A invocação deve ser feita através de um pedido HTTP, utilizando o verbo POST para o endereço especificado de seguida. A chave da API deve ser incluída, tal como dito anteriormente na *query string* substituindo a expressão {API_KEY}.

Pedido HTTP

```
POST https://vision.googleapis.com/v1/images:annotate?key={API_KEY}
```

O *payload* da mensagem a enviar à API deve ser constituído por um objeto JSON com a especificação do tipo de avaliação – de acordo com o descrito na Tabela 15 – a ser feito a cada imagem, bem como deve ser indicada a imagem ou respetivo caminho para a obter.

De acordo com a sua documentação, as imagens a serem processadas podem ser passadas à API de três formas distintas:

- String codificada em base64;
- URI da Google Cloud;
- URL HTTP ou HTTPS de acesso público.

Como a publicação de anúncios disponibiliza as fotografias dos animais de forma pública, por forma a simplificar o processo e diminuir o tráfego de *upload* dos pedidos, o URL de cada imagem é passado diretamente à API.

Tabela 15 – Tipo de avaliação à imagem[15]

Enumeração	Descrição
TYPE_UNSPECIFIED	Funcionalidade não especificada
FACE_DETECTION	Execução da deteção de caras
LANDMARK_DETECTION	Execução da deteção de monumentos
LOGO_DETECTION	Execução da deteção de logotipos
LABEL_DETECTION	Execução da deteção de etiquetas
TEXT_DETECTION	Execução de Optical Character Recognition (OCR)
DOCUMENT_TEXT_DETECTION	Execução da deteção OCR de texto denso de um documento. Tem precedência quando ambos DOCUMENT_TEXT_DETECTION e TEXT_DETECTION estão presentes
SAFE_SEARCH_DETECTION	Execução de modelos de visão computacional para calcular as propriedades *safe-search*
IMAGE_PROPERTIES	Calcular um conjunto de propriedades da imagem, como as cores dominantes na imagem
CROP_HINTS	Execução das sugestões de corte de imagem
WEB_DETECTION	Execução de deteção na *web*

No âmbito deste trabalho, apenas serão utilizadas as funcionalidades LABEL_DETECTION e SAFE_SEARCH_DETECTION com o intuito de obter quais as entidades que se encontram presentes nas imagens, e se existe algum tipo de suspeita sobre o seu teor. Para efeitos de simplificação dos exemplos seguintes, apenas será considerada a funcionalidade LABEL_DETECTION. No exemplo referido pelo Código 8 é possível observar o *payload* de uma mensagem enviada à API, para uma só imagem utilizando o seu URL onde é solicitado que se avalie a imagem de acordo com a funcionalidade LABEL_DETECTION e que apenas sejam retornados no máximo seis resultados.

```
{
 "requests": [
  {
    "image": {
    "source": {
     "imageUri": "https://endereco.com/Chihuahua_3.jpg"
    }
   },
  "features": [
    {
     "type": "LABEL_DETECTION",
     "maxResults": 6
    }
   ],
  }
 ]
}
```

Código 8 – *Payload* do pedido submetido para avaliação à Google Cloud Vision API

[15] Retirado de https://cloud.google.com/vision/docs/reference/rest/v1/images/annotate

Na resposta – Código 9 – é possível encontrar o objeto definido por *labelAnnotations* onde são definidas três propriedades: *mid*, *description* e *score*.

A propriedade *machine-generated identifier* (mid) corresponde a uma entrada de entidade no Google Knowledge Graph. Isto é, em todos os casos em que seja detetada a mesma entidade, será retornado o mesmo identificador. Por exemplo, todas as entradas cuja entidade seja "dog" será retornado o identificador "/m/0bt9lr". A propriedade *description* contém, tal como o nome indica, a descrição do que a API foi capaz de reconhecer na imagem fornecida.

Por fim a propriedade score indica o grau de confiança na existência da entidade acima referida. Os valores apresentados estão num intervalo entre 0 e 1 e que significam respetivamente, sem confiança e confiança muito elevada. No JSON, que pode ser visto no Código 9, é possível consultar a resposta dada ao pedido feito à API, tal como descrito também pelo pedido visto no Código 8.


```
{
 "responses": [
  {
   "labelAnnotations": [
    {
     "mid": "/m/0bt9lr",
     "description": "dog",
     "score": 0.9867094
    },
    {
     "mid": "/m/01z5f",
     "description": "dog like mammal",
     "score": 0.96908855
    },
    {
     "mid": "/m/0kpmf",
     "description": "dog breed",
     "score": 0.94354135
    },
    {
     "mid": "/m/0khhs",
     "description": "chihuahua",
     "score": 0.9353954
    },
    {
     "mid": "/m/04rky",
     "description": "mammal",
     "score": 0.9238294
    },
    {
     "mid": "/m/05mqq3",
     "description": "snout",
     "score": 0.8230413
    }
   ]
  }
 ]
}
```


Código 9 – Resposta da Google Cloud Vision API

Por cada item do objeto *request* do pedido, será retornado um item do objeto *response*, pelo que esta funcionalidade é útil para a solução no sentido em que os pedidos são processados em lote *(batch).*

Com a utilização desta API quis-se obter informação das raças dos animais, mas também perceber-se de forma automática se o conteúdo das fotografias era próprio para ser apresentado ou não.

6 Avaliação

"*It always seems impossible until it's done.*" – Nelson Mandela

Por forma a validar se a solução apresentada é ou não uma mais-valia relativamente às plataformas existentes no mercado, pretende-se que esta seja avaliada. Cada vez que o cidadão colocar um anúncio, quer seja de um animal desaparecido ou de um animal encontrado, a plataforma terá de verificar se já existe no sistema algum anúncio correspondente. No caso de o sistema verificar a existência de correspondência entre anúncios irá notificar o cidadão detentor do animal desaparecido que este poderá ter sido encontrado. Se o utilizador confirmar que o animal lhe pertence, então temos uma correspondência acertada. Baseado nestas sugestões será possível saber se a plataforma acerta ou erra as suas previsões.

De seguida são apresentadas as grandezas avaliadas, as hipóteses testadas e a metodologia utilizada. Por fim são mostrados os resultados obtidos.

6.1 Grandezas a avaliar

Cada vez que a aplicação detetar uma correspondência entre anúncios fará então até três sugestões ao detentor do animal desaparecido. Este por sua vez validará se uma das sugestões feita pelo sistema é ou não verdadeira. Para se calcular a média do número de acertos da aplicação aplicar-se-á a seguinte fórmula:

$$\bar{x} = \frac{Acertos}{Total\ execuções\ do\ algoritmo\ com\ match}$$

6.2 Hipóteses a testar

A partir do momento em que se possui uma média de uma distribuição normal, é possível por intermédio de um intervalo de confiança, verificar quão estável serão as estimativas feitas para novas amostras. O intervalo de confiança pode definir-se através da seguinte fórmula:

$$\bar{x} \pm Z\frac{a}{2} \times \frac{\sigma}{\sqrt{(n)}}$$

6.3 Metodologia de avaliação

A cada submissão de um anúncio, no caso de o sistema encontrar uma possível correspondência, será sugerido ao cidadão um conjunto de até três anúncios para que ele proceda à validação da existência de correspondência. No caso de acerto, o cidadão indicará que o seu anúncio diz respeito ao anúncio sugerido e terminará assim o processo de correspondência para efeitos de avaliação da solução.

Tanto o número de sugestões dadas ao cidadão como o número de acertos feitos pela solução são registados para possibilitar a avaliação.

Apesar de a avaliação não poder ter sido feita num ambiente de produção, pelo facto do desenvolvimento do *website* ainda não se encontrar no fim da primeira iteração, foram criados num ambiente de desenvolvimento 60 anúncios, dos quais 30 são de animais desaparecidos e outros 30 de animais encontrados. Estes foram inseridos de forma a que o algoritmo só fosse executado uma vez por cada um. Isto é, foram colocados 30 anúncios desaparecidos e só depois os 30 anúncios de animais encontrados.

O objetivo passou por criar pares de anúncios desaparecido/encontrado que fossem similares, mas não iguais, que pudessem, no entanto, corresponder. E outros, cujas características não devessem produzir correspondência. Nos anúncios de animais desaparecidos estipularam-se três raças de cães e duas de gatos que foram utilizadas nestes anúncios. No que diz respeito aos cães optou-se pelas raças: Bulldog, Caniche, Galgo; e nos gatos as raças: Europeu comum e Devon Rex. Nos anúncios de animais encontrados não foram especificadas as raças para forçar a utilização da API de análise de fotografias e assim ver que impacto teria no algoritmo da solução.

6.4 Resultados

Tal como referido, foram criados 60 anúncios de forma simulada, o que deu origem a 30 execuções do algoritmo. Neste caso, a execução ocorreu aquando da submissão de novos anúncios de animais encontrados. Na Tabela 16 foi criado um resumo dos resultados obtidos na correspondência de anúncios.

Tabela 16 – Resultados da correspondência de anúncios

Total Anúncios	60
Execuções algoritmo	30
Acertos Cloud Vision API	30
Execuções algoritmo com resultados	26
Resultados corretos	23
Resultados Incorretos (Outros animais)	3

Verificou-se ainda que a estratégia implementada no algoritmo – apenas interessam as raças que constem em anúncios de animais desaparecidos – levou a que a Cloud Vision API acertasse na totalidade dos casos. Desta forma é possível concluir que esta estratégia influenciou de forma positiva o algoritmo que não sofreu desvios, por menor que fossem, vindos de informação da Cloud Vision API.

A média global de acertos da execução do algoritmo de correspondência foi então calculada tendo em conta o número total de notificações efetuadas e o número de acertos.

$$\bar{x} = \frac{23}{26} \cong 88.4$$

O intervalo de confiança para a média de acertos subjacente ao número de notificações feitas é de 88.4 ± 0,125236.

7 Conclusões

A realização de um projeto desta envergadura implica a existência de um conjunto de competências multidisciplinares provenientes, tanto de experiência anterior, como de competências adquiridas enquanto aluno do mestrado.

Apesar de bastante trabalhosa e desgastante, no âmbito desta dissertação, foi possível alcançar um conjunto de objetivos traçados inicialmente e outros, que surgiram no decorrer dos trabalhos que, por tratarem de questões importantes, foram incluídos na resolução do problema.

O cuidado com que foram abordados os constantes desafios que surgiram durante a elaboração deste projeto, permitiu alcançar um conjunto de objetivos importantes para a composição do sistema no seu todo.

De seguida são elencados quais os objetivos alcançados e quais as limitações e trabalho futuro para a concretização da solução.

7.1 Objetivos alcançados

Foi possível, com o trabalho realizado, criar uma estrutura de desenvolvimento baseada nos princípios de *Continuous Integration* e *Continuous Delivery* que agilizou o processo e tornou possível ao cliente ter acesso contínuo ao que ia sendo desenvolvido.

Para a construção das interfaces gráficas do *website*, foi utilizada a *framework* Bootstrap, que proporcionou a inclusão de *user* interfaces responsivos. A inclusão desta *framework* ajudou muito no processo de criação destas interfaces, no entanto foi também um grande desafio pela pouca experiência com a utilização da mesma.

Um dos pontos chave deste projeto, foi a criação e utilização do algoritmo de correspondência de anúncios, conjugado com as ferramentas de visão computacional da Google Cloud Vision tendo esta se mostrado bastante eficiente.

Esta veio permitir algo que era inicialmente pretendido: a deteção das raças dos animais; mas também veio trazer mais segurança à solução, no sentido em que é retornado o tipo de conteúdo apresentado nas imagens. No futuro espera-se ainda que possam ser utilizadas mais características retornadas por esta ferramenta, tais como a idade dos animais. É exemplo disso a etiqueta "*puppy*" ou a etiqueta "*kitten*" muitas vezes retornada em cães e gatos mais novos.

Embora não constasse dos objetivos iniciais, é, nos dias que correm, uma exigência a construção de soluções seguras que não ponham em causa os dados dos seus clientes. Nesse sentido existiu uma preocupação generalizada com questões relativas à segurança, tanto no processo de desenvolvimento como na solução final. Foram implementadas algumas funcionalidades e restrições que tornam a solução mais segura, como é o exemplo das comunicações através de HTTPS, a definição de alguns *headers* HTTP ou a utilização do *syncronizer token pattern* para evitar ataques de CSRF.

A decisão por uma arquitetura que incluísse um Enterprise Service Bus para a integração de sistemas veio trazer ao projeto uma componente importante no que diz respeito à modularidade, bem como permitiu responder de forma célere a requisitos definidos inicialmente. A elevada dependência de componentes externos e a sua mutabilidade foram fatores de preocupação neste projeto, porém, espera-se que os riscos daí provenientes sejam minimizados com a utilização deste sistema.

A aplicação de estratégias de internacionalização permitiu a inclusão de termos e textos tanto em língua portuguesa como em língua inglesa, pelo que a divulgação da associação e deste projeto poderá ser feita em ambas as línguas.

A utilização de ferramentas e tecnologias *open-source* permitiu a redução de custos associados com a aquisição de licenciamento de sistemas operativos, IDEs, ESBs, entre outros. A utilização por exemplo da *framework* .NET Core ou a utilização do WSO2 permitiu a sua execução num servidor com uma distribuição Linux gratuita.

7.2 Limitações e trabalho futuro

Os recursos escassos tanto a nível humano como financeiro continuarão a ser um dos desafios principais para o desenvolvimento de um projeto desta dimensão, pelo que se espera que por essa razão o projeto continue a ser desenvolvido de forma mais morosa do que aquilo que seria esperado.

Como trabalho futuro, finalizar-se-á a primeira iteração e a reavaliação da solução, seguido do desenvolvimento das duas restantes iterações. Nestas existem ainda processos que terão um papel especial para o sucesso deste projeto. São exemplo disso a inclusão de API do Google Maps, para o apoio na inserção e visualização dos anúncios e a visualização dos pontos de interesse no mapa; o PayPal para ajudar a componente financeira do projeto; e a autenticação através das credenciais nas redes sociais para facilitar o acesso à aplicação.

Por fim a construção de um *backoffice* para a recolha e visualização de indicadores – bem como para a gestão de todas as variáveis aplicadas ao sistema – é também outra das prioridades definidas para atribuir autonomia aos gestores do projeto, tirando do programador a responsabilidade dessa gestão.

Referências

Assembleia da República, 2003a. Decreto-Lei 313/2003 de 17 de Dezembro. In *Diário da República*. pp. 8440–8444.

Assembleia da República, 2003b. Decreto-Lei 315/2003 de 17 de Dezembro. In *Diário da República*.

Assembleia da República, 2016. Lei n.º 27/2016 de 23 de agosto. In *Diário da República*. Assembleia da República, pp. 2827–2828. Available at: https://dre.pt/application/file/75171217.

Assembleia da República, 2014. Lei n.º 69/2014 de 29 de agosto. In *Diário da República*. pp. 4566–4567.

Atlassian, 2017. Use SSH keys in Bitbucket Pipelines - Atlassian Documentation. Available at: https://confluence.atlassian.com/bitbucket/use-ssh-keys-in-bitbucket-pipelines-847452940.html [Accessed September 14, 2017].

Barth, A., Caballero, J. & Song, D., 2009. Secure content sniffing for web browsers, or how to stop papers from reviewing themselves. In *Proceedings - IEEE Symposium on Security and Privacy*. pp. 360–371. Available at: http://www.adambarth.com/papers/2009/barth-caballero-song.pdf [Accessed August 17, 2017].

Bishop, C.M., 2006. *Pattern Recognition and Machine Learning*, Springer. Available at: http://www.library.wisc.edu/selectedtocs/bg0137.pdf.

Blancou, J., 2001. A history of the traceability of animals and animal products. *Revue scientifique et technique (International Office of Epizootics)*, 20(2), pp.413–425.

Crabtree, P.J., 1992. Early Animal Domestication and Its Cultural Context. *Man*, 27(4), p.881. Available at: https://books.google.pt/books?hl=pt-PT&lr=&id=5eHoyZ0nTaoC&oi=fnd&pg=PA25&dq=sheep+domestication&ots=GpeYupWgGQ&sig=Z6M5xrgLvVbFaImwxA-CfDeGLoA&redir_esc=y#v=onepage&q=wool&f=false [Accessed January 17, 2017].

Driscoll, C.A., Macdonald, D.W. & O'Brien, S.J., 2009. From wild animals to domestic pets, an evolutionary view of domestication. *Proceedings of the National Academy of Sciences of the United States of America*, 106 Suppl(Supplement_1), pp.9971–8. Available at: http://www.pnas.org/content/106/Supplement_1/9971.full#sec-6.

Duvall, P., Matyas, S. & Glover, a, 2007. *Continuous integration: improving software quality and reducing risk*, Available at: http://medcontent.metapress.com/index/A65RM03P4874243N.pdf%7B%25%7D5Cnhttp://books.google.com/books?hl=en%7B&%7Dlr=%7B&%7Did=PV9qfEdv9L0C%7B&%7Doi=fnd%7B&%7Dpg=PP5%7B&%7Ddq=Continuous+Integration.+Improving+Software+Quality+and+Reducing+Risk%7B&%7Dots=mTh.

Ethan Marcotter, 2010. Responsive Web Design. , pp.1–9. Available at: http://alistapart.com/article/responsive-web-design.

GFK, 2015. Press release. Available at: http://www.gfk.com/pt/insights/press-release/portugal-e-um-pais-pet-friendly/ [Accessed December 16, 2016].

Google, 2016. Vision API - Image Content Analysis. *Google Cloud Platform*. Available at: https://cloud.google.com/vision/ [Accessed February 13, 2017].

Hageltorn, M. & Gustavsson, I., 1981. XXY-trisomy identified by banding techniques in a male tortoiseshell cat. *The Journal of Heredity*, 72(2), p.134. Available at: 0007276515.

Hohpe, G. & Woolf, B., 2003. *Enterprise Integration Patterns: Designing, Building, and Deploying Messaging Solutions*, Available at: http://www.amazon.com/Enterprise-Integration-Patterns-Designing-Deploying/dp/0321200683%5Cnhttp://www.eaipatterns.com/eaipatterns.html.

Humble, J. & Farley, D., 2010. Continuous Delivery. *Continuous delivery*, p.497.

Koen, P.A. et al., 1996. Fuzzy Front End : and Techniques. *Industrial Research*.

Konstantinova, N., 2014. Machine Learning explained in simple words. Available at: http://nkonst.com/machine-learning-explained-simple-words/ [Accessed February 11, 2017].

Manes, A., 2007. Enterprise Service Bus: A Definition. *Burton Group*, pp.1–35. Available at: http://www.gartner.com/id=1405237 [Accessed February 18, 2017].

Martin Fowler, 2006. Continuous Integration. Available at: https://martinfowler.com/articles/continuousIntegration.html [Accessed April 27, 2017].

Microsoft, 2016a. Entity Framework Core | Microsoft Docs. Available at: https://docs.microsoft.com/en-us/ef/core/index [Accessed July 28, 2017].
Microsoft, 2017a. Introduction to the C# Language and the .NET Framework | Microsoft Docs. Available at: https://docs.microsoft.com/en-us/aspnet/core/ [Accessed July 31, 2017].
Microsoft, 2017b. Kestrel web server implementation in ASP.NET Core | Microsoft Docs. Available at: https://docs.microsoft.com/en-us/aspnet/core/fundamentals/servers/kestrel?tabs=aspnetcore1x#when-to-use-kestrel-with-a-reverse-proxy [Accessed August 16, 2017].
Microsoft, 2016b. Model-View-Controller. Available at: https://msdn.microsoft.com/en-us/library/ff649643.aspx [Accessed August 9, 2017].
Microsoft, 2017c. Preventing Cross-Site Request Forgery (XSRF/CSRF) Attacks in ASP.NET Core | Microsoft Docs. Available at: https://docs.microsoft.com/en-us/aspnet/core/security/anti-request-forgery [Accessed August 16, 2017].
Mozilla, 2017a. HTTP headers - HTTP | MDN. Available at: https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers [Accessed August 30, 2017].
Mozilla, 2017b. Referrer-Policy - HTTP | MDN. Available at: https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Referrer-Policy [Accessed August 30, 2017].
OpenCV, 2016. OpenCV. Available at: http://docs.opencv.org/3.2.0/d1/dfb/intro.html [Accessed February 13, 2017].
Osterwalder, A. et al., 2010. *You ' re holding a handbook for visionaries , game changers , and challengers striving to defy outmoded business models and design tomorrow ' s enterprises . It ' s a book for the … written by co-created by designed by*, Available at: http://www.amazon.com/Business-Model-Generation-Visionaries-Challengers/dp/0470876417.
OWASP, 2016. OWASP Secure Headers Project - OWASP. Available at: https://www.owasp.org/index.php/OWASP_Secure_Headers_Project#tab=Headers [Accessed August 17, 2017].
Parlamento Europeu & Conselho da União Europeia, 2013. Regulamento (UE) N.º 576/2013. *Jornal Oficial da União Europeia*, 2013, pp.1–26. Available at: http://www.apmveac.pt/site/upload/5762013.pdf.
Parloff, R., 2016. The Deep-Learning Revolution. *FORTUNE.COM*.
Paul, D.R., Niewoehner, D.R. & Elder, D.L., 2007. *The Thinker's Guide to Engineering Reasoning*,
Pulli, K. et al., 2012. Real-time computer vision with OpenCV. *Communications of the ACM*, 55(6), p.61.
Rodriguez, L., 2016. Store & Protect Sensitive Data in ASP.NET Core. Available at: https://stormpath.com/blog/store-protect-sensitive-data-dotnet-core [Accessed April 19, 2017].
Saeed, M., 2016. Social TagHelpers for ASP.NET Core - Muhammad Rehan Saeed. Available at: https://rehansaeed.com/social-taghelpers-for-asp-net-core/ [Accessed September 19, 2017].
Samuel Gibbs, 2016. Mobile web browsing overtakes desktop for the first time | Technology | The Guardian. Available at: https://www.theguardian.com/technology/2016/nov/02/mobile-web-browsing-desktop-smartphones-tablets [Accessed May 10, 2017].
SNMV, 2016a. Comunicado. , p.2016.
SNMV, 2016b. SIRA - Sistema de Identificação e Recuperação Animal. Available at: http://www.sira.com.pt [Accessed December 16, 2016].
Sommerville, I., 2009. *Software Engineering*,
Sourabh Shirhatti, 2017. Host ASP.NET Core on Linux with Nginx | Microsoft Docs. Available at: https://docs.microsoft.com/en-us/aspnet/core/publishing/linuxproduction?tabs=aspnetcore1x [Accessed September 21, 2017].
Steve Smith, 2016. Testing controller logic in ASP.NET Core | Microsoft Docs. Available at: https://docs.microsoft.com/en-us/aspnet/core/mvc/controllers/testing [Accessed September 23, 2017].
Tomasz Malisiewicz, 2015. Tombone's Computer Vision Blog: Deep Learning vs Machine Learning vs Pattern Recognition. *2015-03-20*. Available at: http://www.computervisionblog.com/2015/03/deep-learning-vs-machine-learning-vs.html [Accessed February 7, 2017].
Woodall, T., 2003. Conceptualising "Value for the Customer": An Attributional, Structural and Dispositional Analysis. *Academy of Marketing Science Review*, 12(5), pp.1–42.
WSO2, 2017. WSO2 Enterprise Service Bus - Only 100% Open Source ESB | WSO2 Inc. Available at:

http://wso2.com/products/enterprise-service-bus/ [Accessed September 18, 2017].
WSPA, 2008. Identification methods for dogs and cats. Available at: http://www.icam-coalition.org/downloads/Identification methods for dogs and cats.pdf.

ANEXO 1

Headers HTTP Utilizados

Headers HTTP

A aplicação de boas práticas e técnicas amplamente discutidas é algo que não deve ser descorado de nenhum projeto. O descuido por questões relacionadas com a segurança pode levar a problemas graves tanto do foro jurídico como para a imagem de qualquer empresa ou instituição.

Uma das muitas características de segurança aplicadas a este projeto foi a inclusão de boas práticas no que diz respeito aos *headers* HTTP. De seguida é referido cada um deles bem como os valores que estes podem assumir.

X-Frame-Options

A aplicação deste *header* permite informar o *browser* se o *website* permite ser aberto dentro de um <frame> ou <iframe>, sendo particularmente útil para defesa contra os ataques do tipo *clickjacking*. Este *header* pode tomar três valores possíveis:

- DENY: Previne que qualquer domínio mostre a página como um frame;
- SAMEORIGIN: Apenas o próprio *website* pode mostrar a página como um frame;
- ALLOW-FROM uri: Apenas o Uniform Resource Identifier (URI) especificado é autorizado a mostrar a página dentro de frames.

Este header é adicionado automaticamente pelo .NET Core com o valor SAMEORIGIN, no entanto, este foi removido e adicionada a opção DENY, por não se fazer planos da inclusão de frames ou iframes.

Strict-Transport-Security (Barth et al. 2009)

HTTP Strict Transport Security (HSTS) é um mecanismo de politica de segurança que permite proteger um website contra os ataques de downgrade e de cookie hijacking. A sua utilização faz com que os browsers ou outros user agents compatíveis somente interajam com o *website* através de conexões HTTPS.

Os valores a definir são os seguintes:

- **max-age**: Tempo, em segundos, que o browser deve lembrar que o *website* só deve ser acedido através de HTTPS;
- **includeSubdomains**: Parâmetro opcional, que, se definido, aplica a regra também a todos os subdomínios do mesmo *website*;
- **preload**: Parâmetro opcional, que, se definido, permite incluir o domínio numa lista mantida pelo Chrome e utilizada também por outros *browsers* como o Firefox e Safari. Este parâmetro deve ser utilizado apenas se todos os domínios e subdomínios funcionem exclusivamente com HTTPS, caso contrário os *browsers* vão impedir o seu acesso.

X-Content-Type-Options

Os *browsers* possuem um algoritmo que inspeciona o conteúdo (MIME *sniffing*) que vem na resposta de um pedido HTTP, e que em determinados casos sobrepõe o MIME *type* fornecido pelo servidor (OWASP 2016).

A definição deste *header* com o valor *nosniff* previne que o *browser* tente deduzir automaticamente qual o MIME *type* da página prevenindo alguns ataques de XSS que tiram partido desta funcionalidade.

Com a publicação de anúncios de animais desaparecidos e encontrados, é pedido também que sejam enviadas fotografias dos animais. Esta funcionalidade pode ser utilizada por um atacante para injeção de conteúdo malicioso. Sem este *header* o browser tentará interpretar o ficheiro carregado como se fosse HTML sobrepondo-se ao MIME *type* definido.

X-XSS-Protection (OWASP 2016)

A utilização deste *header* permite que o *browser* ative o filtro contra XSS, permitindo ou negando a execução de *scripts*.

Os valores possíveis a atribuir a este *header* são:

- **0**: Filtro desabilitado
- **1**: Filtro habilitado. Se um ataque de XSS for detetado, o *browser*, no sentido de travar o ataque irá efetuar a limpeza de *tags* da página ou que possam envolver algum tipo de perigosidade para o *website*. Este processo é conhecido como *sanitizing*.
- **1; mode=block**: Filtro habilitado. Em vez de efetuar o processo de *sanitizing,* o *browser* ao detetar um ataque XSS irá bloquear o processo de construção da página (*rendering)*.
- **1; report=http://[YOURDOMAIN]/your_report_URI**: Filtro habilitado. O browser executará o processo de *sanitizing* da página e reportará a violação para o URI definido. Esta funcionalidade só está disponível no Chromium através da utilização dos *reports* de violação CSP.

A configuração aconselhada para este *header* e considerada uma boa prática é "X-XSS-Protection: 1; mode=block".

Content-Security-Policy

A experiência diz-nos que a utilização de *whitelists* é mais segura do que a utilização de *blacklists,* isto é, com uma *whitelist* é possível definir quem é que deve ter acesso a determinado recurso, enquanto que com uma *blacklist* é possível definir quem não tem acesso. A utilização de uma política assente na utilização de *whitelists* permite ter uma granularidade e clareza maior sobre quem está autorizado a aceder a um recurso. Com as *blacklists* é possível negar o acesso a um conjunto de entidades, existindo sempre a possibilidade de serem esquecidas outras de importância.

A aplicação do *header* HTTP Content Security Policy (CSP) é uma medida de segurança importante contra os ataques do tipo XSS, pois permite definir quais as fontes de conteúdo que estão autorizadas a serem utilizadas pelo *website*, prevenindo que conteúdo malicioso possa ser carregado de fontes não incluídas na *whitelist*.

Neste *header* é possível definir um conjunto de diretivas com as quais é possível definir que fontes é que estão autorizadas pelo *website*. São exemplos de diretivas os seguintes valores:

- **script-src**: Quais as fontes de scripts que estão autorizadas;
- **style-src**: Quais as fontes de estilos (CSS) que estão autorizadas;
- **img-src**: Quais as fontes de imagens que estão autorizadas;
- **font-src**: Quais as fontes de fontes de texto que estão autorizadas.

Referrer-Policy

Quando um determinado utilizador clica sobre um *hyperlink* na internet o que acontece habitualmente é o browser do utilizador enviar o URI que originou o pedido através da utilização do *header referrer*. É deste modo que por exemplo produtos como o *Google Analytics* conseguem saber a origem das visitas ao nosso *website*.

O *Referrer Policy* é um *header* recente que permite a um *website* controlar a quantidade de informação que o browser incluir na navegação para fora de um determinado documento

Com a utilização deste *header* é possível a um *website* autorizar quanta informação o *browser* pode passar na navegação para fora do documento atual (Mozilla 2017b).

O Referrer-Policy pode tomar os seguintes valores (Microsoft 2017c):

- **no-referrer**: O *header Referer* é inteiramente omitido. Não é enviada qualquer informação no *referrer* juntamente com os pedidos;
- **no-referrer-when-downgrade:** Esta é a opção por omissão dos *user agents*, se nenhuma política for especificada. A origem do pedido é enviada *a priori* no *referrer* se o destino for tão seguro quanto a origem (ex. HTTPS para HTTPS), mas não é enviado se o pedido for para um local menos seguro (HTTPS para HTTP);
- **origin:** Em todos os casos apenas é enviada a origem do documento no campo *referrer*. Isto é, se for pedido o documento https://example.com/page.html apenas é enviado no campo *referrer* https://example.com/;
- **origin-when-cross-origin**: É enviado o URL completo quando se trata de navegação para um local da mesma origem, mas quando a navegação é efetuada para o exterior apenas é enviado o endereço de origem;
- **same-origin**: Apenas no caso de se tratar do mesmo *website* é que é enviado o *referrer* preenchido. No caso de os pedidos serem para outros *websites*, nenhuma informação é enviada no *referrer*.

- **strict-origin**: Apenas é enviado a origem do documento como *referrer* para *a priori* um local tão seguro quanto a origem (HTTPS para HTTPS), mas não envia qualquer informação no caso de se tratar de um destino menos seguro (HTTPS para HTTP);
- **strict-origin-when-cross-origin**: É enviado o URL completo quando se tratar de pedidos da mesma origem. A origem do pedido é enviada *a priori* no *referrer* se o destino for tão seguro quanto a origem (ex. HTTPS para HTTPS), mas não é enviado se o pedido for para um local menos seguro (HTTPS para HTTP);
- **unsafe-url**: É enviado o URL completo tanto para pedidos na mesma origem ou para o exterior.