INSTITUTO SUPERIOR DE ENGENHARIA DO PORTO
MESTRADO EM ENGENHARIA INFORMÁTICA




# Tradutor de Língua Gestual Portuguesa Orientado a Serviços de Saúde

**GUSTAVO MOREIRA MAGALHÃES BASTOS FERREIRA**
Junho de 2022

POLITÉCNICO
DO PORTO

Instituto Superior de Engenharia do Porto

# Tradutor de Língua Gestual Portuguesa Orientado a Serviços de Saúde

**Gustavo Ferreira**

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

**Orientador: Dr. Nuno Bettencourt**

Porto, 30 de junho de 2022

# Dedicatória

A todos os que me acompanharam ao longo do meu percurso académico e estiveram sempre presentes para mim.

# Resumo

Graças aos vários avanços tecnológicos a qualidade de vida do ser humano melhorou exponencialmente, tendo afetado várias setores da sociedade. No entanto, esta evolução não acompanhou todas as comunidades, tendo alguma sido alienadas e marginalizadas em relação a ela. Este fenómeno, que reflete o contraste digital entre elementos da sociedade, denomina-se de *Digital Divide*. O *Digital Divide* pode ser observado na comunidade surda, que se debate diariamente com várias barreiras que poderiam ser facilmente ultrapassadas com recurso à tecnologia, nomeadamente no contexto da saúde. Os indivíduos surdos acabam por não aceder a serviços de saúde pelo facto de não se sentirem confortáveis para tal, uma vez que não possuem independência suficiente para o fazer devido à dificuldade de comunicação com os demais. Para além disto, já foram registados vários casos de diagnósticos errados à conta desta barreira de comunicação, que pode levar a problemas mais graves [1].

É com base nesta premissa que surge a necessidade de desenvolvimento de um tradutor digital capaz de auxiliar os surdos em serviços de saúde, de uma forma simples, através do telemóvel e sem necessidade de equipamento especial.

Neste documento, após a introdução ao projeto e sua contextualização, é possível encontrar informação relativa a soluções já existentes para o tema em questão, bem como tecnologias de destaque e relevância.

Posteriormente, é realizada uma análise de valor onde se verifica a validade e a necessidade da criação do presente projeto. Nesta análise, após identificação e exploração da oportunidade, em conjunto com uma proposta de valor, determina-se que o projeto possui um valor considerável para o seu público alvo. Profissionais que dedicam a sua carreira a acompanhar pessoas com deficiências auditivas relatam que aplicações digitais capazes de intermediar a comunicação entre surdos e os demais seria vantajoso ao ponto de revolucionar a qualidade de vida destas pessoas.

De seguida é construída a análise de domínio bem como o *design* arquitetural e consequente implementação de uma solução para um tradutor digital em tempo real, com base nas tecnologias apresentadas anteriormente e fazendo uso de algoritmos de *machine learning* e de reconhecimento de imagens.

Por último, de modo a avaliar a solução construída, apresentam-se métodos de avaliação que visam determinar o potencial funcional, o grau de usabilidade e de satisfação desta. Estes métodos baseiam-se em testes de *software*, inquéritos e testes de hipóteses.

**Palavras-chave:** Tradutor Digital, *Digital Divide*, Língua Gestual, Saúde, Análise de Imagens

# Abstract

Thanks to the multiple technological advances, the quality of life of the human kind has improved, affecting the various sectors of the society. Nonetheless, this evolution did not follow every community, given that some were marginalized in regards to it. This phenomenon, that reflects the contrast in between members of the society, is called Digital Divide. Digital divide can be observed in the deaf community, that finds daily barriers that could be easily surpassed with the use of technology, like the access to health care services. The deaf end up not being able to use health care services due to the fact that they do not possess enough independency to do it, due to communication issues. Beyond this, there were some cases where multiple diagnoses were made on account of this communication barrier, that can lead to more serious problems [1].

It is based on the premise exposed on the last paragraph that the need to develop a digital translator capable to help the deaf people in health care services in a simple way, through the phone and without the need to use special equipment is born.

In this document, after the introduction to the project and it's contextualization, it is possible to find an investigation on existing solutions to the current theme, as well as relevant technologies.

Posteriorly, a value analysis is made where it is verified the validity and necessity of the current project. In this analysis, after the identification and exploration of the opportunity, together with a value proposal, it is determined that the project has value for it's target audience. Professionals that dedicate their carreers to work with deaf people say that digital applications capable of intermediating the communication between deaf people and everybody else would be advantageous and would revolutionize their quality of life.

Afterwards, it is built the domain analysis as well as the architectural design and consequent implementation of a solution to a real time digital translator, based on the technologies presented before and making use of machine learning and image recognition algorithms.

Lastly, in a way to evaluate the developed solution, are presented some evaluation methods, whose goal is to determine the functional potencial, the degree of usability and satisfaction of it. These methods are based on software tests, inquiries and hypothesis tests.

**Keywords:** Digital Translator, Digital Divide, Sign Language, Health, Image Analysis

# Agradecimentos

Em primeiro lugar, agradeço aos meus pais, Miguel Ferreira e Cristina Ferreira, por tudo o que fizeram por mim ao longo da minha vida para que pudesse chegar até aqui.

Agradeço a todos os meus amigos, André Madureira, Eduardo Carneiro, João Dias, João Fonseca, José Ferreira e Tomás Santos. Sem eles não tinha conseguido alcançar os meus objetivos, ajudaram-me sempre ao longo do curso e pude contar sempre com eles.

Não posso deixar de agradecer ao meu orientador, Nuno Bettencourt, que esteve sempre disponível para me esclarecer todas as duvidas e que me acompanhou ao longo deste projeto.

Por último, agradeço à minha namorada, Catarina Moreira, pelo apoio incondicional e por me motivar nos momentos mais complicados. Sem ti não chegava tão longe, obrigado por tudo.

*"I wish there was a way to know you were in the good old days before you actually left them."*

— Unknown

# Conteúdo

# Lista de Figuras

# Lista de Tabelas

# Lista de Código

# Lista de Acrónimos

| | |
|---|---|
| AHP | *Analytic Hierarchy Process.* |
| API | *Application Program Interface.* |
| ASL | *American Sign Language.* |
| ATILGP | Associação de Tradutores e Intérpretes de Língua Gestual Portuguesa. |
| BSL | *British Sign Language.* |
| CSAT | *Customer Satisfaction Score.* |
| DL | *Deep Learning.* |
| ISEP | Instituto Superior de Engenharia do Porto. |
| LG | Língua Gestual. |
| LGP | Língua Gestual Portuguesa. |
| ML | *Machine Learning.* |
| NCD | New Concept Development. |
| OpenCV | *Open Source Computer Vision Library.* |
| PWA | *Progressive Web App.* |
| QFD | *Quality Function Deployment.* |
| SoC | *Separation of Concerns.* |
| TMDEI | Tese de Mestrado do Departamento de Engenharia de Informática. |
| UML | *Unified Modeling Language.* |

# Capítulo 1

# Introdução

Neste capítulo é feita uma breve apresentação ao leitor sobre o trabalho desenvolvido no âmbito da unidade curricular Tese de Mestrado do Departamento de Engenharia de Informática (TMDEI) do Mestrado em Engenharia de Software do Instituto Superior de Engenharia do Porto (ISEP).

Inicialmente, é apresentado o contexto do projeto, para que haja um melhor entendimento do tema desta dissertação. De seguida, é identificado o problema e os objetivos que se pretendem alcançar. É ainda analisada a metodologia de trabalho e as contribuições que o projeto pode oferecer. Por último, a estrutura do presente documento é detalhada de modo a facilitar a sua compreensão.

## 1.1 Contexto

Graças aos vários avanços tecnológicos a qualidade de vida do ser humano melhorou exponencialmente. A forma de trabalhar, de viver e de relaxar foi completamente revolucionada, uma vez que alguns dos problemas que outrora prevaleciam, como a falta de informação ou até de entretenimento, foram, na sua maioria, extintos. Este avanço tem afetado os diversos pilares da sociedade, como a educação, a comunicação e a saúde. Ainda assim, existe um grupo de pessoas que tem sido marginalizado em relação a todo este desenvolvimento. Este fenómeno, que reflete o contraste digital entre elementos da sociedade, denomina-se *Digital Divide* [2].

É com base nesta premissa que surge a necessidade de desenvolver o projeto descrito neste documento. O objetivo é construir um produto capaz de auxiliar a integração social de indivíduos surdos na área da saúde, permitindo-os ser independentes nos vários processos envolvidos numa ida a um hospital, por exemplo.

A principal motivação para o qual este tema foi selecionado, prende-se essencialmente com o facto de possuir o potencial para melhorar as vidas de uma grande franja populacional. O conhecimento deve ser aplicado para o bem-estar do ser humano e esta é uma oportunidade para tal.

## 1.2 Problema

Como foi mencionado na secção 1.1, apesar de todos os avanços tecnológicos, ainda é complicado integrar na sociedade pessoas com deficiências auditivas severas, devido

ao *Digital Divide* e também por não existir o conhecimento da língua gestual fora deste núcleo de indivíduos e daqueles que os acompanham regularmente.

Assim, torna-se uma tarefa (quase) impossível de conduzir uma simples conversa com uma pessoa surda. Após inquirir alguns profissionais de saúde, foi possível determinar que estes acreditam que não existem condições de integrar pessoas com estas dificuldades na sociedade, uma vez que não conseguem, ou sentem bastantes dificuldades, a aprender a ler e escrever, optando muitas vezes exclusivamente pela língua gestual, sendo esta a sua única forma de comunicação. A comunidade surda, devido à pandemia, vê a ser construída uma nova barreira na sua comunicação com os demais. Vários surdos conseguem entender certas palavras pelo movimento labial, mas o uso da máscara obriga ao desuso desta tática, uma vez que alguns nem notam que alguém possa estar a tentar falar com eles [3].

O que foi descrito no parágrafo anterior constitui um retrato generalista da atualidade, sendo demasiado abrangente para o contexto de uma dissertação. Deste modo, o problema aqui proposto representa apenas uma pequena parte disto, nomeadamente a dificuldade que os surdos sentem em comunicar em hospitais e centros de saúde. A saúde é um bem essencial e o acesso a ela deve ser fornecido a todos os cidadãos. No entanto, um indivíduo surdo, tal como já foi explicado, possui dificuldades em comunicar com os restantes no quotidiano e fazê-lo dentro de um hospital/centro de saúde não é uma exceção. O facto de num hospital existir um número muito reduzido ou até mesmo nulo de intérpretes faz com que se torne complicado para um surdo conseguir fazer algo tão simples como marcar uma consulta na receção ou simplesmente saber que alguém o está a chamar de uma sala de espera [4]. Muitas vezes, estas pessoas apenas conseguem ir a uma consulta quando acompanhadas por familiares ou intérpretes [4]. De modo a diminuir as barreiras e disparidades entre aqueles com impedições auditivas e o cidadão comum e garantir independência aos primeiros, é necessário que dificuldades ao acesso de serviços básicos (como o caso da saúde) sejam removidas. Assim, foi proposta a criação de uma aplicação capaz de aliviar este problema, sendo este tema abordado com maior detalhe na secção 1.3

## 1.3 Objetivos

De modo a diminuir o *Digital Divide* existente, bem como facilitar a compreensão de língua gestual para leigos, pretende-se o desenvolvimento de uma plataforma digital que contenha valor pela sua contribuição social, integrando os indivíduos com deficiências auditivas na área tecnológica. Assim, pretende-se que o projeto a construir seja capaz de funcionar como um intermédio entre o paciente (surdo) e o profissional de saúde, de modo a que ambos se entendam na totalidade, sem que seja necessário equipamento especial para o fazer. Posto isto, os objetivos propostos são:

- Desenvolver uma aplicação de fácil acesso, sendo possível utilizá-la tanto no computador como em telemóveis;
- A aplicação deve detetar gestos e traduzi-los corretamente para texto;
- Realizar traduções sem necessidade de equipamento especial, apenas a câmara do dispositivo tecnológico a ser utilizado.

## 1.4 Metodologia

A primeira etapa neste projeto passa por investigar e entender o modo de funcionamento da língua gestual portuguesa, bem como esta se estrutura.

De seguida, é necessário pesquisar sobre as tecnologias e programas já existentes que demonstrem ser relevantes para o projeto a desenvolver, sendo esta matéria abordada e aprofundada no capítulo 2.

Terminando a fase anterior, passa-se para a análise de valor, onde se estudam as necessidades dos possíveis utilizadores, bem como os ganhos de que estes vão usufruir.

O passo seguinte é a engenharia de requisitos, onde se elabora o modelo de domínio e se expõe os requisitos funcionais e não funcionais, e, posteriormente, o design arquitetural.

Para terminar, as próximas etapas são a implementação do projeto, que será constantemente acompanhada pelo orientador do ISEP e a avaliação da solução conseguida.

A informação necessária para a escrita deste documento será retirada de artigos, livros e páginas *web*, bem como de entrevistas realizadas a indivíduos que convivam e trabalhem com a comunidade surda.

## 1.5 Contribuições

O projeto em questão pretende, para além da presente dissertação, a criação de uma aplicação que funcione como um tradutor de Língua Gestual Portuguesa (LGP) para língua portuguesa escrita e ainda um artigo científico que retrate o seu desenvolvimento e eventuais conclusões.

## 1.6 Estrutura do Documento

O atual documento encontra-se dividido em oito capítulos distintos, cada um com o seu tema e respetiva importância para o seguimento do relatório.

No presente capítulo, capítulo 1, apresenta-se o problema que este projeto pretende resolver, bem como os objetivos propostos e a forma como o relatório se encontra estruturado.

No capítulo 2, para além de introduzir alguns conceitos fundamentais para o entendimento do trabalho realizado, faz-se ainda referência a trabalhos que estejam, de alguma forma, relacionados com o trabalho construído, quer seja por uma funcionalidade relevante ou pela lógica que o envolve. São ainda exploradas e analisadas tecnologias que se demonstram relevantes para a concretização do desafio proposto.

No que toca ao capítulo 3, apresenta-se a análise de valor elaborada, onde se aprofunda a oportunidade encontrada e o valor que o projeto proporciona aos utilizadores, entre outros tópicos relacionados.

O capítulo 4 expõe a análise de domínio, incluindo a exploração do modelo de domínio e da engenharia de requisitos, onde os requisitos funcionais e não funcionais são levantados.

O capítulo 5 foca-se no desenho arquitetural da solução.

No capítulo 7 ilustram-se os métodos de avaliação da solução, explicando as formas a ser utilizadas para determinar o sucesso ou insucesso do projeto.

Por último, no capítulo 8, realiza-se a conclusão do relatório, tanto a nível dos objetivos concretizados, como ao nível do trabalho a desenvolver no futuro. Nesta capítulo elabora-se, também, a apreciação final do desenvolvimento do projeto na sua totalidade.

# Capítulo 2

# Estado de Arte

Este capítulo aborda conceitos, tecnologias e projetos existentes que se revelam significativos para a obtenção da solução esperada.

Na primeira parte deste capítulo é feita uma introdução ao tema da Língua Gestual (LG), aprofundando depois uma pequena parte do seu universo, nomeadamente a LGP. De seguida, são explorados projetos semelhantes ao que se pretende desenvolver e, posteriormente, descrevem-se tecnologias de interesse, bem como os seus aspetos positivos e negativos. A pesquisa das tecnologias divide-se em três partes distintas:

- Tecnologias de *machine learning*;
- Tecnologias para reconhecimento de mãos e de gestos;
- Tecnologias de desenvolvimento.

Por último, são explorados conceitos de *design* arquitetural e analisadas as conclusões retiradas de toda a informação recolhida num formato de resumo, onde são salientadas as tecnologias e abordagens a serem utilizadas no desenvolvimento do presente projeto.

## 2.1 Língua Gestual

A língua gestual é já utilizada há vários anos e consiste na utilização das mãos para realizar gestos em conjunto com expressões faciais e movimentos do corpo para transmitir e comunicar ideias. Atualmente, é essencialmente utilizada por pessoas surdas ou por pessoas mudas que conseguem ouvir. No entanto, existem ainda outros usos para a língua gestual, como por exemplo no contexto militar ou em indústrias em que o ruído é predominante e utilizar a voz como meio de comunicação deixa de ser viável. O equívoco de que a língua gestual é meramente uma representação manual de um idioma oral ainda prevalece na sociedade. Na realidade, estas formas de comunicação pouco apresentam em comum. A língua gestual possui uma complexidade semelhante a um idioma oral mas são independentes entre si, tendo evoluído paralelamente. Um exemplo demonstrativo de tal é o facto de existirem línguas gestuais diferentes em cada país, até em países que utilizam o mesmo idioma, como o caso de Inglaterra e dos Estados Unidos da América. Estes países, onde a língua mais utilizada é o inglês, possuem duas línguas gestuais distintas, British Sign Language *(BSL)* e American Sign Language *(ASL)*, respetivamente. Assim,

indivíduos fluentes nestas línguas não conseguiriam comunicar entre si, ao contrário de um americano e inglês a falar nas suas línguas maternas [5].

### 2.1.1 Língua Gestual Portuguesa

A LGP tem origem no século XIX e é utilizada pela comunidade surda portuguesa para expressar a sua cultura e valores, pelo que a 15 de Novembro de 1997 foi aprovada pela constituição da república como uma das três línguas oficiais de Portugal, em conjunto com a Língua Portuguesa e o Mirandês [6]. Algo interessante é a existência de variedades dialectais, que variam conforme a região do país. Deste modo, existem diferenças lexicais entre os gestos da comunidade surda do norte e do sul de Portugal. Para além disto, também é possível comunicar num registo mais formal ou informal, consoante o contexto social em que o indivíduo surdo se insere. Assim, é possível concluir que a LGP, apesar das várias diferenças, também se assemelha à língua portuguesa, sendo que a primeira possui regras específicas que advêm do facto de se classificar como um idioma espácio-visual [7]. Uma vez que a LGP é uma língua bastante recente é alvo de uma constante evolução e renovação [8].

#### Estrutura da LGP

A base da LGP são os queremas. Um querema pode ser explicado recorrendo à língua portuguesa. Nesta última, as palavras são constituídas por sons (fonemas), a unidade mínima de qualquer idioma oral e utilizado em todas as palavras. Desta forma, um querema está para a LGP como um fonema está para a língua portuguesa [7]. Cada gesto na LGP é constituído por cinco queremas distintos:

- Configuração da mão: forma que a mão assume;
- Movimento: direcionalidade ou movimento da mão/dedos;
- Expressão não manual: certos gestos possuem alguma marcação feita com a bochecha, boca, língua ou dentes;
- Localização: sítio do espaço ou corpo onde os gestos se situam quando executados;
- Orientação: orientação para a qual a palma da mão está virada.

Uma vez que basta alterar um dos queremas para dar um novo significado ao gesto, a LGP possui uma infinidade de combinações possíveis. Tal como na língua portuguesa onde apenas um som de uma palavra pode alterar o seu significado ou até originar erros, o mesmo é possível com na LGP com a modificação de um dos cinco queremas base [7].

Depois de formados os gestos, é importante ainda mencionar que estes se dividem em três categorias distintas: gestos icónicos; gestos referenciais e gestos arbitrários. Os primeiros são utilizados para representar um objeto, normalmente fazendo-o delineando o próprio, como que o tentando desenhar com o dedo. Uma alternativa é com as mãos representar o objeto ou uma ação sobre ele, auferindo-lhe dimensão e forma. Os gestos referenciais representam os pronomes pessoais, partes do corpo ou até da roupa. Por último, os gestos arbitrários são todos aqueles que não espelham qualquer relação direta com o conceito que representam [9].

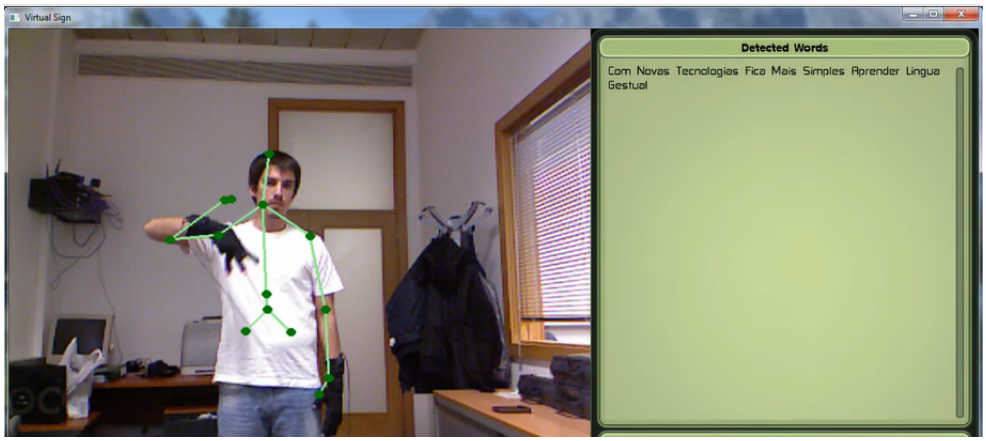


FIGURA 2.1: Interface Gráfica do Virtual Sign ()

## 2.2 Soluções Existentes

Nesta secção são abordadas soluções capazes de traduzir língua gestual para um idioma oral (em texto ou voz) e/ou vice-versa, ou sejam captar texto e/ou voz e traduzi-lo para língua gestual.

### 2.2.1 Virtual Sign

O *Virtual Sign* é uma aplicação web que tem como principal objetivo simplificar o processo de aprendizagem e comunicação entre todos os que comunicam através da língua gestual portuguesa. Deste modo, proporciona uma acesso mais equitativo à educação, diminuindo assim o *digital divide* existente na sociedade [10].

Esta aplicação funciona como um tradutor biderecional de LGP, conseguindo captar texto e apresentar a sua tradução em língua gestual através de um avatar, sendo também capaz de assimilar gestos e traduzi-los para texto. Para que esta última tradução seja possível é necessária a utilização de uma câmara cinética (*Kinect*) e também de luvas de sensores. Assim, o *Kinect* fornece informações sobre o movimento e a posição das mãos e do corpo e as luvas fornecem informações precisas sobre a configuração das mãos e dos dedos. Na figura 2.1 é possível verificar a interface gráfica do *Virtual Sign* onde um utilizador faz uso das luvas de sensores para que os seus gestos sejam traduzidos [10]. É importante também referir que as traduções são todas feitas em tempo real [10].

Para além do tradutor em tempo real, o *Virtual Sign* também oferece uma implementação do conceito de *gamificação* para auxiliar a aprendizagem de uma nova língua. Este jogo apresenta três níveis, em que cada um aborda uma parte diferente da LGP. O primeiro nível leva o utilizador a interagir com o alfabeto, o segundo possui várias das palavras mais utilizadas e o terceiro e último nível engloba as frases mais comuns. O processo de aprendizagem baseia-se na observação do avatar e da reprodução de gestos pela parte do utilizador, fazendo também uso das luvas de sinais e da câmara cinética [11].

### 2.2.2 Hand Talk

A *Hand Talk* é uma aplicação móvel disponível para os sistemas Android e iOS que já conta com mais de quatro milhões de *downloads* e que se especializa em traduzir voz e texto para ASL e Libras (língua gestual oficial do Brasil) [12].

A aplicação possui uma interface amigável, como é possível verificar na figura 2.2, onde basta introduzir a mensagem que queremos traduzir através da voz ou texto (selecionando o campo de texto ou o microfone, respetivamente) e de forma imediata o intérprete (avatar que se encontra no centro do ecrã) reproduz a tradução na língua gestual especificada [12].

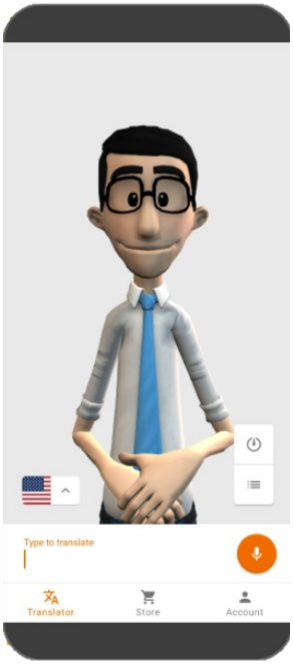


FIGURA 2.2: Interface Gráfica da aplicação Hand Talk ([12])

### 2.2.3 English to Sign Language Translator

Este tradutor é uma ferramenta *online* que traduz inglês escrito para BSL ou para ASL em tempo real. No entanto, ao contrário dos exemplos mencionados anteriormente, este tradutor não possui um avatar que demonstra como efetuar os gestos, optando apenas por exibir ao utilizador um conjunto de imagens estáticas que representam os gestos. Ainda assim, o tradutor limita-se a traduzir cada letra para o seu correspondente em LG, como pode ser verificado na figura 2.3, semelhante ao que seria soletrar cada palavra. Deste modo, esta tradutor apesar de rápido e capaz de fornecer ajuda momentânea é "pobre" no que toca a vocabulário.

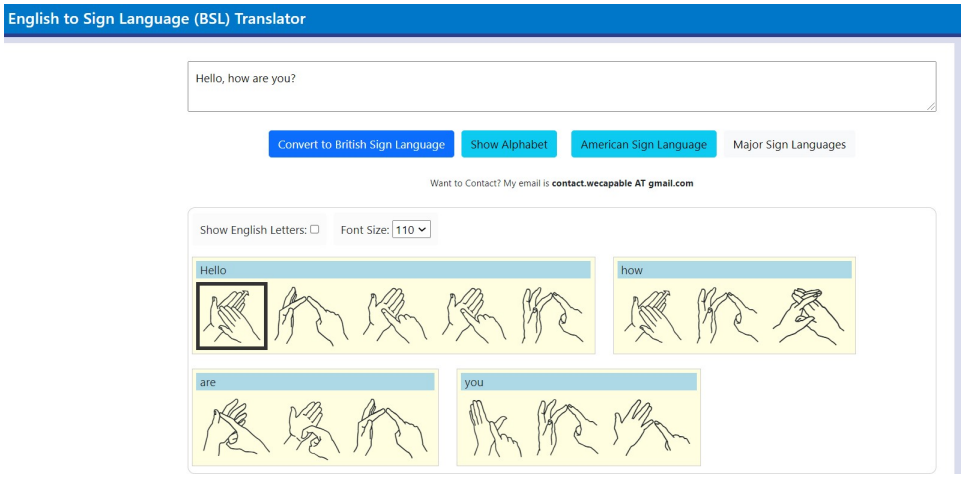


FIGURA 2.3: Tradutor de inglês para ASL/BSL

### 2.2.4 SignAll Chat

O SignAll Chat é um programa que permite a um indivíduo surdo comunicar com um indivíduo que não tenha qualquer conhecimento de ASL em tempo real. O objetivo é permitir que qualquer negócio, como um hotel ou banco por exemplo, possam contratar indivíduos surdos para que estes consigam comunicar com os clientes [13].

A interface gráfica da aplicação é semelhante a *chats* utilizados nas várias redes sociais, como se pode verificar na figura 2.4.

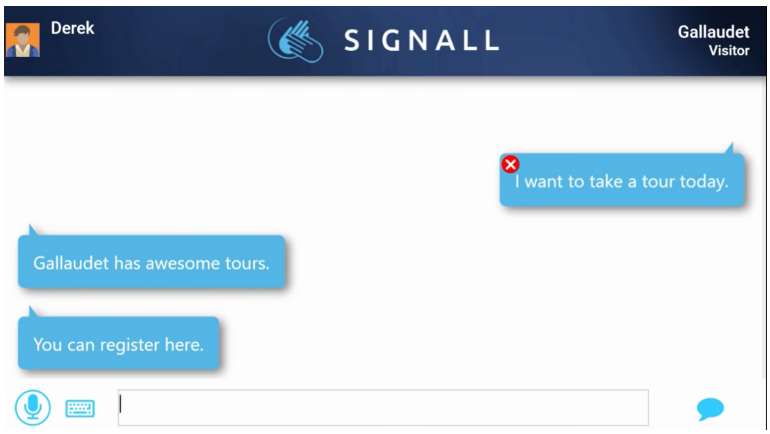


FIGURA 2.4: Interface Gráfica da aplicação SignAll Chat ([13])

De um lado o utilizador que não entende língua gestual fala para um microfone (em inglês), que capta a mensagem e a transforma em texto. Do outro lado, com o auxílio de quatro câmaras distintas, o indivíduo surdo efetua os gestos (apenas funciona com ASL) e estes serão também transformados em texto [13], como pode ser observado na figura 2.5.

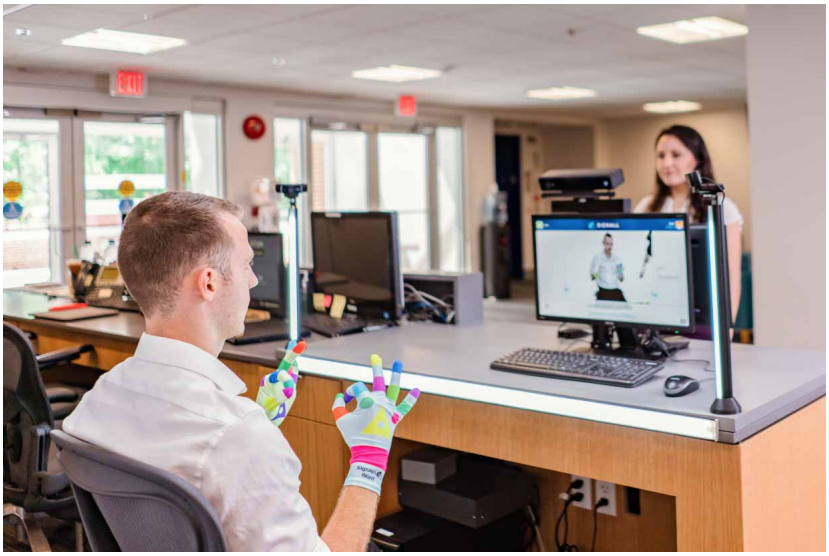

FIGURA 2.5: Indivíduo surdo a utilizar SignAll Chat ([13])

No entanto, o SignAll Chat não traduz o idioma oral para língua gestual, sendo apenas um tradutor unidirecional.

## 2.3 Comparação Soluções Existentes

Esta secção tem o intuito de comparar as soluções apresentadas ao longo deste capítulo. A tabela 2.1 expões as várias soluções com base em quatro critérios, relevantes para o tema em que se inserem.

TABELA 2.1: Comparação de soluções existentes

| | **Input** | **Output** | **Bidirecional** | **Idiomas** |
|---|---|---|---|---|
| **Virtual Sign** | Gestos e Texto | Gestos e Texto | ✓ | Língua portuguesa e LGP |
| **Hand Talk** | Texto | Gestos | X | Língua portuguesa, língua inglesa, Libras e ASL |
| **English to Sign Language Translator** | Texto | Gestos Estáticos | X | Língua inglesa, ASL e BSL |
| **SignAll Chat** | Gestos e Texto | Texto | X | Língua inglesa e ASL |

## 2.4 Inteligência Artificial

A inteligência artificial (IA) possui várias definições, sendo que para Jhon McCarthy, da universidade de Stanford, é a ciência de criar máquinas inteligentes, nomeadamente programas computacionais inteligentes. A IA não se prende apenas ao conceito dos computadores entenderem a inteligência humana, mas sim não se confinarem a métodos biologicamente observáveis [14].

Ainda assim, décadas precedentes à anterior definição ter sido publicada, Alan Turing, também conhecido como o pai da IA, iniciou a discussão sobre este mesma, onde propôs o teste de Turing. Neste teste, um humano terá o papel de interrogar um computador e um outro humano através de mensagens escritas num computador, sendo que o objetivo final será o interrogador conseguir decifrar quem é quem. Caso o interrogador não consiga distinguir entre o computador e o humano, segundo Turing, alcançamos a verdadeira inteligência artificial [15].

Numa abordagem mais moderna do tema, Stuart Russel e Peter Norvig, defendem que a IA pode ser dividida nos quatro pontos seguintes, que diferenciam sistemas computacionais na base da racionalidade e na dualidade do pensar em contrapartida com agir [16]:

- Sistemas que pensam como humanos;
- Sistemas que agem como humanos;
- Sistemas que pensam racionalmente;
- Sistemas que agem racionalmente.

A definição de Turing pertenceria à segunda categoria, "Sistemas que agem como humanos". De uma forma simples, a IA combina a ciência computacional com *datasets* (coleção estruturada de informação [17]) com o intuito de resolver vários tipos de problemas. A IA contém outros campos mais específicos, como *Machine Learning* (ML) [18].

## 2.5 Machine Learning

ML, como foi mencionado na secção 2.4, constitui um sub-grupo da IA, podendo ser definido como o estudo científico de algoritmos e modelos estatísticos que sistemas computacionais utilizam para realizar uma dada tarefa sem a necessidade de serem programados para tal explicitamente [19]. De uma forma mais simples, ML serve para ensinar aos computadores a forma mais eficiente de tratar a informação e aprender com ela.

Existem vários algoritmos de ML, que são utilizados conforme a necessidade e o tipo de problema, sendo que alguns acabam por se tornar mais populares que outros. A figura 2.6 representa uma árvore com as várias abordagens de implementar ML:

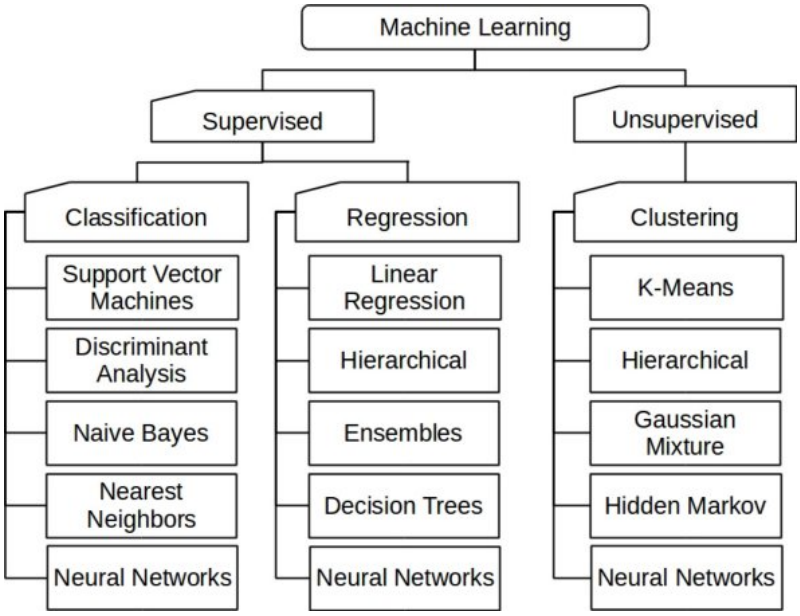


FIGURA 2.6: Algoritmos de *Machine Learning* [20]

De todos os algoritmos espelhados na figura 2.6, o mais popular e mais comummente utilizado é o que se baseia na utilização de *Neural Networks* (NN) (ou redes neuronais). As NN, também conhecidas por *Artificial Neural Networks*, são a base de algoritmos de *Deep Learning* (DL), um sub-conjunto de ML. O nome e estrutura das NN baseia-se no cérebro humano, tentando imitar o sistema neurológico e o modo como os neurónios enviam sinais entre eles [21].

As NNs são constituídas por camadas, existindo a camada de entrada (*input layer*), uma ou mais camadas escondidas (*hidden layers*) e uma camada de saída (*output layer*), sendo que cada camada possui um conjunto de nós (ou *nodes*). Estes ligam-se a nós da camada seguinte, tendo um peso associado e um limite (*threshold*). Caso o *output* de cada nó seja superior a um dado *threshold*, o nó a este ligado vai ser ativado, passando para a próxima camada da rede neuronal. Caso contrário, a informação não passa para a próxima camada da rede. Este processo é repetido até se alcançar a última camada, onde o nó com o maior valor é considerado o "vencedor", sendo este o resultado obtido [21]. A figura 2.7 espelha o aspeto mais comum de uma NN.

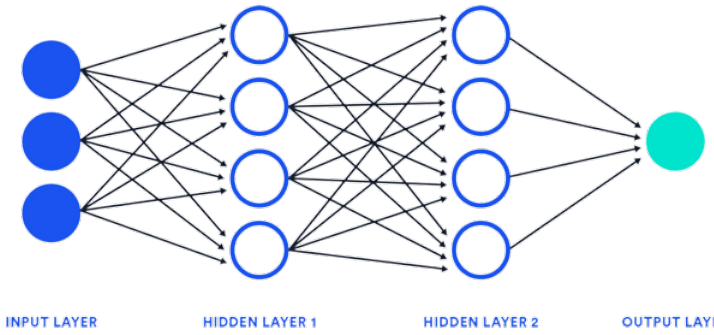


FIGURA 2.7: Representação comum de uma Rede Neuronal [19]

As NNs precisam de ser treinadas com bastante informação para melhorarem e se tornarem mais precisas com o passar do tempo, sendo necessário um investimento neste processo. No entanto, assim que a rede neuronal se encontrar afinada, já pode ser utilizada com mais confiança, realizando tarefas de computação com uma velocidade elevada. Acabam por ser muito utilizadas no reconhecimento de imagens e de sons, sendo que até o algoritmo de pesquisa da "Google" se baseia neste algoritmo [21]. Para além disto, existem muitas mais diversas formas de aplicar redes neuronais, tornando este método eficaz e genérico. Existem ainda diferentes tipos de redes neuronais que vão ser explorados de seguida.

### 2.5.1 Multi-Layer Perceptrons

As NN do tipo *Multi-Layer Perceptrons* (MLP), também conhecidas por *Feedforward Neural Networks*, são o exemplo mais típico de modelos de ML. As redes MLP são constituídas por várias funções, estando este modelo associado à utilização de grafos direcionados sem ciclos, que descrevem o modo como as funções se interligam. Utilizando o exemplo apresentado na figura 2.7, cada camada representa uma função. Na realidade, esta figura espelha o conceito de uma *neural network* do tipo MLP. Como foi mencionado na secção 2.5, existe uma camada de entrada, uma de saída e ainda um número ilimitado de camadas escondidas que se encontram entre as duas primeiras. O número de camadas que uma rede MLP possui constitui a sua profundidade (*depth*) [22].

### 2.5.2 Convolutional Neural Networks

As *Convolutional Neural Networks* (CNN) distinguem-se das restantes pelo seu desempenho superior no reconhecimento de imagem e áudio [23]. Existem três tipos de camadas (*layers*) distintas nas CNNs:

- *Convolutional layer*
- *Pooling layer*
- *Fully-connected (FC) layer*

A *convolutional layer* é a primeira camada de uma CNN, dado que podem seguir-se de novas *convolutional layers* ou então de *pooling layers*. Esta camada é o bloco principal das CNNs e é onde a maioria do esforço computacional se encontra. Uma imagem, por exemplo, que entre nesta camada vai ser transformada em valores numéricos, permitindo à NN interpretar e extrair padrões relevantes, sendo este processo denominado de *convolution* [22].

As *pooling layers* têm a função de reduzir o número de parâmetros de entrada. Esta camada aplica um filtro no *input* sem pesos ao *input* recebido. Existem assim dois tipos de *pooling*:

- *Max pooling*: À medida que o filtro percorre o *input*, os maiores valores são selecionados e enviados para a lista de saída. Esta abordagem tende a ser a mais utilizada [22];
- *Average pooling*: Enquanto o filtro percorre o *input*, o valor médio dentro de um dado intervalo é calculado e depois enviado para a lista de saída [22].

Apesar de muita informação ser perdida nestas camadas, este processo reduz a complexidade e aumenta a eficiência de todo o processo [22].

A *fully-connected layer* é sempre a última de uma CNN, possuindo a tarefa de classificar os resultados com base nos recursos extraídos ao longo das camadas anteriores [22].

Com cada camada a CNN aumenta em complexidade. As primeiras camadas focam-se em tarefas simples. No caso de o *input* ser uma imagem, as camadas iniciais deveriam focar-se nas cores e arestas existentes, por exemplo. Continuando com esta linha de pensamento, conforme a imagem fosse sendo processada pelas várias camadas da CNN, seria possível começar a reconhecer elementos maiores e até formas, até que no fim, na camada FC se identifica o objeto que se encontra representado na figura. A figura 2.8 ilustra o funcionamento de uma CNN, e como a informação de entrada passa pelas várias camadas.

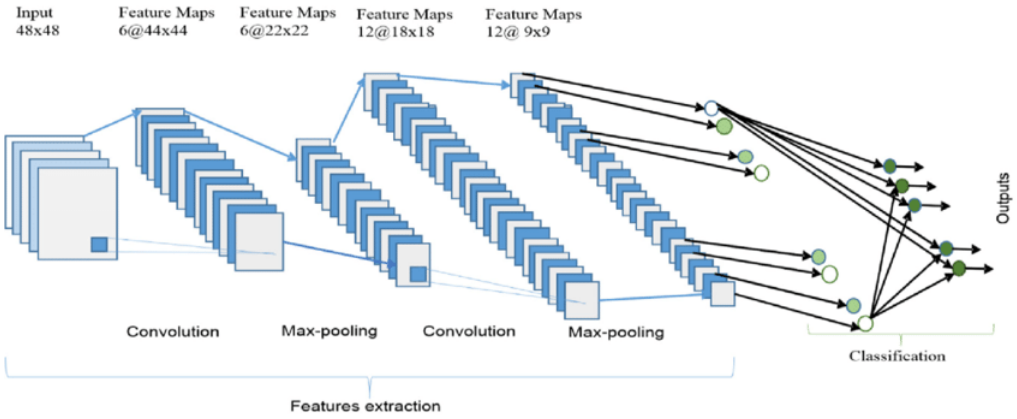


FIGURA 2.8: Representação de uma *Convolutional Neural Network* [24]

Este tipo de NN é muito utilizado nas novas tecnologias e nas mais distintas áreas profissionais, sendo possível dar uso a CNNs no segmento do *Marketing* (plataformas de redes sociais, por exemplo) e até em aplicações destinadas à saúde, chegando até ser possível auxiliar os médicos a identificar tumores malignos [22].

### 2.5.3 Recurrent Neural Networks

As *Recurrent Neural Networks* (RNN) são mais um tipo de NN que utilizam a informação numa sequência lógica, utilizando *datasets* que acompanham a informação ao longo do tempo. Este tipo de rede neuronal é mais utilizado em contextos onde o eixo temporal é uma variável importante, como a tradução de idiomas, processamento de língua natural, reconhecimento de voz e captação de imagens, estando mesmo incorporado em aplicações como a "Siri" e o "Google Translate" [25].

De forma semelhante aos tipos de NN abordados nas secções anteriores, as RNNs necessitam de informação para treinar e aprender. No entanto, distinguem-se destas pela capacidade memorizar informação recolhida anteriormente, uma vez que o resultado de um dado *input* pode ser influenciado por resultados obtidos previamente [25].

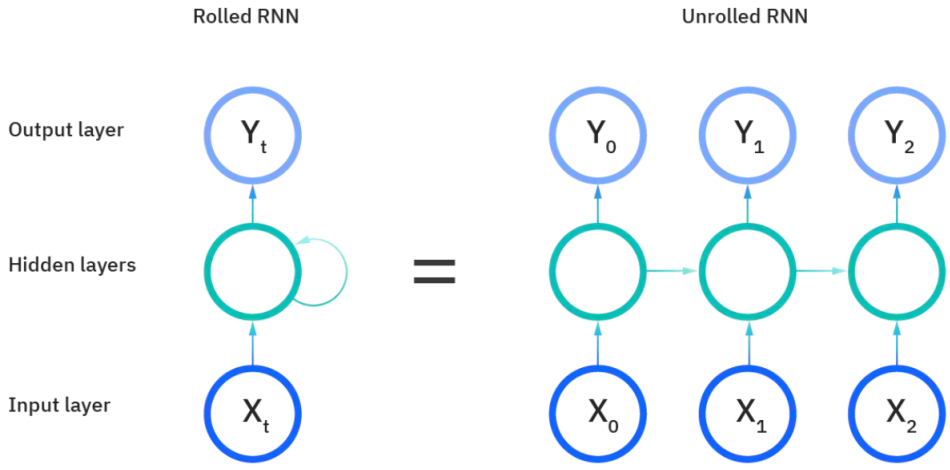


FIGURA 2.9: Representação de uma *Recurrent Neural Network* [25]

Na figura 2.9 é possível observar dois modos de retratar uma RNN. O modelo à esquerda representa uma RNN de uma forma mais simples, sendo que o modelo à direita ilustra a sua extensão completa. Nesta última versão o leitor é capaz de identificar as camadas individuais. De modo a melhor explicar o modo de funcionamento das RNNs, será exemplificado um pequeno caso prático, recorrendo à frase "Vi-me grego para entender o problema", utilizada quando se deseja expressar dificuldade em entender algo. Para que a frase faça sentido, é preciso que seja expressa nesta ordem específica, pelo que a rede neuronal deve ter em atenção a ordem em que cada palavra aparece para que o resultado final tenha nexo. Regressando à figura 2.9, as camadas no modelo à direita mapeiam individualmente as palavras constituintes da frase utilizada como exemplo, enquanto que o modelo à esquerda pode

ser considerado como a representação da frase na sua totalidade. A computação inicia-se com a palavra "vi-me", que é processada desde X0 a Y0. De seguida, em X1 dá entrada a palavra "grego", sendo que nas *hidden layers* que se seguem, os cálculos vão ter em conta *inputs* anteriores, neste caso apenas "vi-me", gerando o *output* Y1. Deste modo, na terceira camada, as palavras “vi-me” e “grego” estariam também contempladas, para que seja possível prever a próxima palavra da frase e assim consecutivamente ao longo da RNN, até se concluir a frase.

Para além disto, as RNNs também se distinguem das restantes pelo uso do algoritmo *backpropagation through time* (BPTT). O algoritmo BPTT possui os mesmos princípios que o tradicional *backpropagation*, onde o modelo de ML se treina autonomamente calculando erros existentes. O BTPP diferencia-se pelo facto de somar os erros a cada camada, enquanto que nas restantes abordagens (CNN e MLP) não necessitam deste trabalho extra, do mesmo modo que não partilham parâmetros entre camadas, ao contrário da rede neuronal em questão [25].

Por outro lado, as RNNs tendem a encontrar dois tipos de problemas, causados pelo tamanho do gradiente. O primeiro problema designa-se *exploding gradients*, sendo que este ocorre quando o gradiente é muito elevado, levando a que os pesos do modelo cresçam demasiado até que eventualmente possuam um valor inválido. O segundo é conhecido por *vanishing gradients*, que é essencialmente o oposto do problema anterior. Neste, os pesos vão ficando cada vez mais reduzidos até que se tornem insignificantes. Uma possível solução para estes problemas pode ser reduzir a complexidade do modelo, removendo algumas das *hidden layers* [25].

As RNNs podem ser construídas com base em distintas arquiteturas, como as apresentadas de seguida:

- ***Bidirectional recurrent neural networks* (BRNN):** Este tipo de arquitetura possibilita que para além das RNNs fazerem uso da informação processada no passado, possam também fazer uso de informação futura de modo a melhor a precisão do algoritmo [25];
- ***Long short-term memory* (LSTM):** Esta é provavelmente a arquitetura mais popular, tendo surgido como uma solução para o problema do *vanishing gradient*. Nesta abordagem utilizam-se células nas *hidden layers* da NN, as quais possuem três portões - *input*; *output*; *forget*. Estes portões controlam o fluxo de informação que é necessária para prever o resultado da rede. Assim, resolve-se o problema das RNNs não conseguirem fazer previsões sem ser com informação existente no passado recente, uma vez que os portões vão deixando sempre passar a informação que possa vir a ser útil entre as camadas, removendo também a que for repetida, de modo a não ocupar espaço desnecessário [25];
- ***Gated recurrent units* (GRU):** Esta arquitetura é semelhante à LSTM, visto que também se debruça sobre o problema de memória a curto prazo das RNNs. Todavia, em vez de utilizar células, utiliza *hidden states*, usando dois *gates* ao invés de 3, um *reset gate* e um *update gate*. Em semelhança à LSTM, estes controlam a qual a informação e a respetiva quantidade desta que se mantém [25].

### 2.5.4 Comparação de Redes Neuronais

Nesta secção é feita uma comparação entre os vários tipos de redes neuronais abordados ao longo da secção 2.5. Esta comparação encontra-se explícita na tabela 2.2, onde cada tipo de NN é analisada em relação a um dado critério.

Tabela 2.2: Comparação dos tipos de redes neuronais analisados

| | **MLP** | **CNN** | **RNN** |
|---|---|---|---|
| ***Input Data*** | Tabular | Imagens | Sequencial |
| **Conexões Recorrentes** | X | X | ✓ |
| **Partilha de Parâmetros** | X | ✓ | ✓ |
| **Relação espacial** | X | ✓ | X |
| ***Vanishing/Exploding Gradient*** | ✓ | ✓ | ✓ |

## 2.6 Tecnologias para Machine Learning

Nesta secção são exploradas tecnologias criadas com o intuito de facilitar o desenvolvimento de aplicações de *machine learning*.

### 2.6.1 Tensorflow

"Tensorflow" é uma plataforma *open-source* que tem o objetivo de criar aplicações de ML, oferecendo diferentes níveis de abstração para que o utilizador possa selecionar o que se adequa melhor às suas necessidades. Esta ferramenta fornece modelos pré-treinados, com base em redes neuronais, que podem ser utilizados tanto no desenvolvimento *Web*, compatível com linguagens de programação como *Javascript* ou *Python*, mas também no desenvolvimento móvel (*Android* e *iOS*). Para além disto ainda fornece a possibilidade ao utilizador de criar e treinar os seus próprios modelos com recurso a APIs poderosas e intuitivas. É também disponibilizada a "TensorBoard", que essencialmente é um conjunto de ferramentas que auxilia na visualização de métricas, de grafos representativos de modelos e muito mais [26]. Um exemplo deste recurso pode ser observado na figura 2.10:

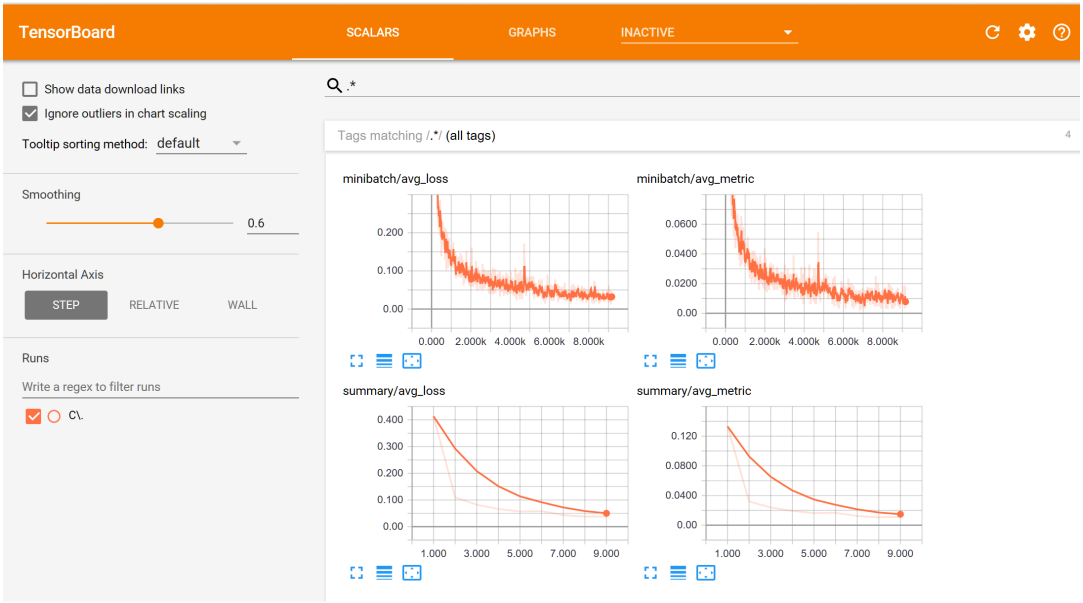


FIGURA 2.10: Interface Gráfica da TensorBoard [27]

Todas estas vantagens levam a que esta seja a plataforma de ML mais famosa da atualidade, sendo que a própria "Google" (empresa fundadora do "TensorFlow"), a utiliza em quase todos os seus produtos.

### 2.6.2 OpenCV

*Open Source Computer Vision Library* (OpenCV) é uma biblioteca *open-source* de ML e *computer vision*.*Computer vision* é o ramo da IA que se baseia na interpretação e entendimento do mundo através de imagens e vídeos [28]. Assim, este ferramenta foi construída para fornecer uma infraestrutura comum para aplicações baseadas em *computer vision*, de modo que o desenvolvimento desta área fosse acelerado. Esta biblioteca possui extensos algoritmos de ML otimizados, que podem ser utilizados para detetar caras, identificar objetos e até classificar ações de humanos captadas em vídeos, sendo muito utilizada em aplicações onde é necessário captar esta informação em tempo real. É multi-plataforma, fornecendo *interfaces* em várias linguagens de programação e suporte para vários sistemas operativos [29].

## 2.7 Frameworks de Reconhecimento de Gestos

Esta secção analisa *frameworks* [1] para o desenvolvimento de aplicações de ML, mas ao contrário das abordadas na secção 2.6 que são para uso genérico, estas são especializadas na identificação de mãos e gestos.

[1] Estrutura base que contém ferramentas e componentes que agilizem o processo de desenvolvimento de software [30].

### 2.7.1 MediaPipe

MediaPipe é uma *framework* multi-plataforma desenvolvida pela Google, que disponibiliza soluções otimizadas para ML completamente customizáveis [31]. Esta ferramenta foi incluída nesta secção pelo facto de possuir uma solução treinada para identificar mãos de forma precisa, sendo que também possui modelos treinados para reconhecer e identificar caras, olhos, cabelo e até poses feitas com o corpo inteiro. No entanto, neste trabalho, apenas será dado foco a soluções capazes de interpretar mãos, uma vez que se enquadra no tema presente.

Na realidade, apesar de existir um módulo do MediaPipe destinado especificamente a mãos (MediaPipe Hands), esta secção irá aprofundar outra solução, nomeadamente MediaPipe Holistic. Esta caracteriza-se por constituir uma combinação de três outras soluções: *Hands*, *Face Mesh* e *Pose*. Como foi explorado na secção 2.1.1, certos gestos da LGP necessitam de movimentos da cara e do corpo em conjunto com os gestos das mãos para que seja possível transmitir a mensagem correta. Por esta mesma razão é que foi optado por analisar esta abordagem, que é capaz de detetar mãos, elementos da cara e ainda braços, como ilustra a figura 2.11.

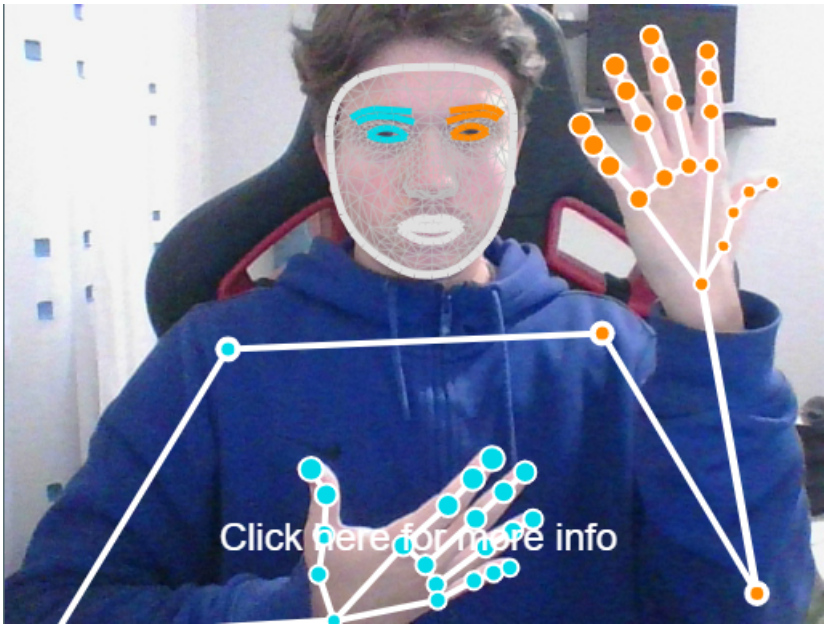


FIGURA 2.11: Utilização da *framework* MediaPipe Holistic

Analisando a figura 2.11 é possível notar que vários pontos do corpo estão a ser identificados, onde os coloridos a azul identificam a parte esquerda do corpo e os laranjas a parte direita.

### 2.7.2 Fingerpose

Fingerpose é uma ferramenta criado para identificar diferentes gestos em imagens e vídeo em tempo real. Esta *framework* é customizável, sendo possível acrescentar a quantidade de gestos que consegue interpretar [32]. No entanto, este processo é

pouco automatizado. Esta *framework* deteta os diferentes gestos através de ficheiros com formato *json* [2] onde se regista a posição que os vários dedos da mão devem ter para reproduzir o gesto em questão. A figura 2.12 exemplifica um destes ficheiros:

```
// thumb:
victoryDescription.addDirection(Finger.Thumb, FingerDirection.VerticalUp, 1.0);
victoryDescription.addDirection(Finger.Thumb, FingerDirection.DiagonalUpLeft, 1.0);
victoryDescription.addDirection(Finger.Thumb, FingerDirection.DiagonalUpRight, 1.0);

// index:
victoryDescription.addCurl(Finger.Index, FingerCurl.NoCurl, 1.0);
victoryDescription.addDirection(Finger.Index, FingerDirection.VerticalUp, 1.0);
victoryDescription.addDirection(Finger.Index, FingerDirection.DiagonalUpLeft, 1.0);
victoryDescription.addDirection(Finger.Index, FingerDirection.DiagonalUpRight, 1.0);
victoryDescription.addDirection(Finger.Index, FingerDirection.HorizontalLeft, 1.0);
victoryDescription.addDirection(Finger.Index, FingerDirection.HorizontalRight, 1.0);

// middle:
victoryDescription.addCurl(Finger.Middle, FingerCurl.NoCurl, 1.0);
victoryDescription.addDirection(Finger.Middle, FingerDirection.VerticalUp, 1.0);
victoryDescription.addDirection(Finger.Middle, FingerDirection.DiagonalUpLeft, 1.0);
victoryDescription.addDirection(Finger.Middle, FingerDirection.DiagonalUpRight, 1.0);
victoryDescription.addDirection(Finger.Middle, FingerDirection.HorizontalLeft, 1.0);
victoryDescription.addDirection(Finger.Middle, FingerDirection.HorizontalRight, 1.0);

// ring:
victoryDescription.addCurl(Finger.Ring, FingerCurl.FullCurl, 1.0);
victoryDescription.addCurl(Finger.Ring, FingerCurl.HalfCurl, 0.9);

// pinky:
victoryDescription.addCurl(Finger.Pinky, FingerCurl.FullCurl, 1.0);
victoryDescription.addCurl(Finger.Pinky, FingerCurl.HalfCurl, 0.9);
```

FIGURA 2.12: Ficheiro *json* utilizado na ferramenta Fingerpose

Analisando a figura 2.12, é necessário indicar para cada dedo da mão a orientação no espaço deste ou então se está semi ou completamente dobrado. Depois do preenchimento destes ficheiros a ferramenta trata do resto e imediatamente identifica o novo gesto. É importante notar que apenas é possível reconhecer gestos estáticos com recurso a esta ferramenta.

Ao contrário do exemplo anterior, presente na secção 2.7.1, com esta ferramenta apenas é possível utilizar uma mão, não sendo capaz de identificar mais nada, como se conclui analisando a figura 2.13, que espelha o uso da *framework* em estudo.

[2] formato de texto que agrupa a informação em pares nome/valor [33].

FIGURA 2.13: Utilização da *framework* Fingerpose

### 2.7.3 HandTrackJS

A *framework* HandTrackJS possibilita a deteção de vários gestos a partir de uma imagem ou de um vídeo em tempo real, tendo sido desenvolvida para ser utilizada em aplicações *Web*, uma vez que apenas pode ser usufruída com a linguagem de programação *javascript*. Esta ferramenta utiliza redes neuronais otimizadas, construídas com recurso à biblioteca "TensorFlow", que foi explorada na secção 2.6.1. É capaz de detetar ambas as mãos e também caras, tendo a vantagem de conseguir discernir mãos fechadas, abertas e com um dedo levantado. Em semelhança às restantes *frameworks* apresentadas ao longo deste capítulo é de fácil implementação, apresentando uma precisão elevada. A figura 2.14 apresenta o uso de HandTrackJS para detetar uma mão aberta e outra fechada.

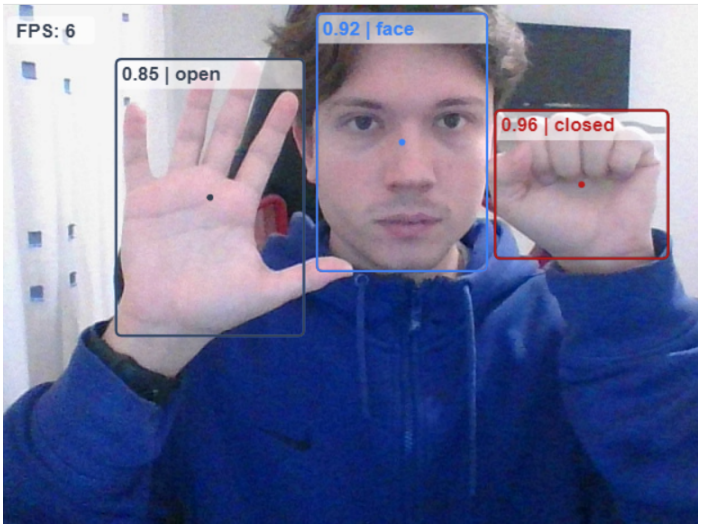


FIGURA 2.14: Utilização da *framework* HandTrackJS

### 2.7.4 Comparação de Frameworks de Reconhecimento de Gestos

A presente secção serve para comparar as ferramentas analisadas nas secções 2.7.1, 2.7.2 e 2.7.3. A tabela 2.3 analisa as *frameworks* com base em quatro critérios distintos que são considerados relevantes para a elaboração do presente projeto.

Tabela 2.3: Comparação de *Frameworks* de reconhecimento de gestos

| | **MediaPipe** | **FingerPose** | **HandTrackJS** |
|---|---|---|---|
| **Multi-Plataforma** | ✓ | X | X |
| **Possibilidade de utilizar duas mãos** | ✓ | X | ✓ |
| **Sem necessidade de equipamento extra** | ✓ | ✓ | ✓ |
| **Capacidade de detetar caras** | ✓ | X | ✓ |

## 2.8 Frameworks de Desenvolvimento de Progressive Web Apps

Tendo em conta o problema que este projeto tenta resolver, faz sentido que a aplicação a construir seja de fácil acesso, podendo ser utilizado através de um computador ou de um telemóvel. Para que seja possível cumprir este requisito, a aplicação será construída como uma *Progressive Web App* (PWA). Uma PWA pode ser considerada como uma versão móvel otimizada de uma aplicação *web* capaz de ser utilizada nos *smartphones* a partir do *browser* [3]. São semelhantes a outras aplicações móveis, mas sem o incoveniente de necessitarem de ser descarregadas através de uma loja de aplicações. Para além disto, estas aplicações encontram-se disponíveis mesmo em modo *offline* e possuem um custo reduzido de desenvolvimento, uma vez que deixa de ser necessário desenvolver aplicações distintas para *web* e para dispositivos móveis [35]. Existem ainda muitos outros benefícios que demonstram o porquê do sucesso das PWAs.

Assim, nesta secção são exploradas algumas das *frameworks* que possibilitam o desenvolvimento deste tipo de aplicações, tanto na vertente de *frontend*[4], como de *backend*[5].

### 2.8.1 Angular

Angular é uma plataforma para a construção da interface gráfica de aplicações. Esta tecnologia foi criada pela ´´Google" e utiliza um conjunto de linguagens de

[3] programa que fornece uma forma de interação com toda a informação na internet [34]
[4] módulo responsável por interagir com o utilizador e gerir a interface gráfica
[5] servidor da aplicação, responsável por gerir os componentes internos do sistema

programação. O Angular proporciona a criação de *Single-Page Applications* (SPA), aplicações executadas dentro de um *browser* que não necessitam de ser atualizadas durante o seu uso [36].

### 2.8.2 React

React é uma biblioteca da linguagem de programação *Javascript* para o desenvolvimento de interfaces gráficas. De forma semelhante ao Angular, também é utilizada para a construção de SPAs. No entanto, quem criou a *framework* em questão foi o "Facebook" [37].

### 2.8.3 Comparação de Frameworks de Desenvolvimento de PWAs

Começando pelas ferramentos de desenvolvimento de *frontend*, tanto React como Angular são bastante populares no universo da construção de interfaces gráficas, possuindo bastantes semelhanças, inclusive a arquitetura baseada em componentes, que torna todo o processo de desenvolvimento mais rápido, possibilitando a reutilização de código. Ainda assim, existem diferenças notáveis entre ambas. Em primeiro lugar, o React fornece mais liberdade na organização do código, enquanto que o uso de Angular restringe um pouco este aspeto. Ainda assim, a segunda oferece vantagens que a primeira não tem, como o uso de *dependency injection*.

Concluindo, ambas as tecnologias são bastantes populares e utilizadas mundialmente, pelo que o suporte disponível é enorme para as duas. No entanto, o desenvolvimento com React é mais rápido, sendo esta uma variável importante no projeto em questão, pelo que foi a selecionada.

## 2.9 Frameworks de Desenvolvimento de Backend

A secção 2.8 explora fundamentalmente ferramentas para a interface gráfica que irá interagir com o utilizador, mas é essencial que exista também um servidor (*backend*) capaz de fazer o processamento ´´invisível" dos dados e de os fornecer ao *frontend*. Desta forma, esta secção analisa *frameworks* próprias para o desenvolvimentos do lado do servidor de soluções tecnológicas.

### 2.9.1 .Net Core

Esta é uma plataforma de intuito geral, desenvolvida pela "Microsoft", que pode ser utilizada para construir projetos em vários sistemas operativos, com várias linguagens de programação [38].

### 2.9.2 Flask

Flask é uma *framework* de desenvolvimento *web* escrita em *python*, que foi desenvolvido por Armin Ronacher. É uma ferramenta simples mas extensível, onde o utilizador está livre para tomar as decisões que quiser (como a escolha da base de dados a usar, por exemplo) [39].

### 2.9.3 Spring

O Spring constitui uma *framework* que fornece um modelo abrangente de programação e configuração para aplicações desenvolvida em *Java*, sem restrições no que toca a ambientes de implantação. Spring oferece um leque de funcionalidades que facilita a construção de projetos de raiz [40].

### 2.9.4 Comparação de Frameworks de Desenvolvimento de Backend

No que toca às *frameworks* de *backend*, foi decidido que seria utilizado o Flask e Spring. Foram selecionadas duas *frameworks*, pelo facto de haver vários módulos na parte do servidor, sendo esta matéria melhor explorada ao longo do capítulo 5. A parte associada à tradução propriamente dita de LGP será construída com recurso a Flask, pela razão de se basear em *python*, linguagem mais utilizada e com mais suporte para ML. Para mais, Flask foi construído com o intuito de fornecer um desenvolvimento simples e rápido, enquadrando-se com a escolha feita para o *frontend*. Spring será utilizado no módulo ligado a *analytics*, uma vez que é uma aplicação orientada a objetos mais básica e que não precisa de ML. Visto que o Spring possibilita a construção rápida deste tipo de projetos, foi a *framework* escolhida.

## 2.10 Tecnologias de Bases de Dados

Tal como se verifica na maioria das soluções tecnológicas existentes, é necessário armazenar diversa informação. Deste modo, nesta secção são exploradas algumas ferramentas que potenciam a conservação de dados.

### 2.10.1 MongoDB

MongoDB é uma base de dados que armazena a informação em documentos JSON flexíveis, onde os campos podem variar de documento para documento e a sua estrutura pode ser alterada com o passar do tempo [41]. O excerto de código 2.1 representa um exemplo de como a informação fica guardada com o MongoDB.

```
{
    "_id": "6wt75dvcd7svds",
    "firstName": "first",
    "lastName": "last",
    "adress": {
        "street": "Street 1",
        "city": "City 1",
        "zipCode": "90404"
    }
}
```

Listing 2.1: Exemplo estrutura de dados com MongoDB.

O modelo empregue pelo MongoDB é capaz de mapear os objetos existentes no código construído, facilitando o processo de persistência destes [41].

É importante notar que MongoDB é de utilização gratuita e possui uma disponibilidade elevada, bem como uma curva de aprendizagem reduzida [41].

### 2.10.2 SQL Server

SQL Server é um sistema de bases de dados relacional desenvolvido pela "Microsoft". Foi construído sobre *SQL*, sendo esta a linguagem de programação utilizada para interagir com as bases de dados aqui criadas [42].

### 2.10.3 H2

H2 é uma base de dados relacional com suporte em SQL desenvolvida em *Java*. Estas bases de dados caracterizam-se por serem rápidas, *open source* e com elevada disponibilidade, sendo muitas vezes embutidas em aplicações *Java* pela sua fácil implementação [43].

### 2.10.4 Comparação de Tecnologias de Bases de Dados

Visto que é necessário armazenar dados para a avaliação do sistema (*analytics*), como é explorado mais a fundo no capítulo 5, e também de utilizadores, foi decidido que seriam utilizadas duas ferramentas de bases de dados distintas, uma para cada um destes propósitos. MongoDB foi selecionada para conservar os dados dos utilizadores e H2 para armazenar os dados de *analytics*.

## 2.11 Conceitos de Design

Nesta secção são explorados alguns conceitos de *design* que foram adotados para a análise e desenho do problema, presentes nas secções 4 e 5, respetivamente.

### 2.11.1 Modelo de Domínio

Segundo o autor Eric Evans, o modelo de domínio é definido como a ideia que é suposto ser transmitida com a construção do próprio diagrama. O modelo de domínio é uma abstração rigorosamente organizada e seletiva do conhecimento dos peritos do domínio [44]. O uso do modelo de domínio é um costume na Engenharia de Software, sendo muito utilizado para formular o negócio, tendo em conta que o seu objetivo é representar uma aproximação do domínio em questão.

Nestes diagrama (considerando que a linguagem de modelação é UML e o diagrama é de classes), identificam-se classes concetuais, bem como objetos reais do domínio de interesse, que refletem conceitos de domínio e as interações que estes possuem entre si. Estes conceitos podem ou não converter-se em elementos de software. O conceito de classe concetual é distinto dos conceitos de entidade e conceito de negócio.

### 2.11.2 Modelo 4 + 1

Uma forma de descrever a arquitetura de *software* implementada é através do modelo 4+1. Este modelo consiste em cinco vistas distintas:

- **Vista lógica** – modelo de objetos do *design* (quando o *design* é orientado a objetos) [45];

- **Vista de processos** – vista que demonstra os aspetos de concorrência e de sincronização do *design* [45];
- **Vista física** – vista que descreve o mapeamento do *software* no *hardware* [45];
- **Vista de implementação** (ou de desenvolvimento) – vista que descreve a organização estática do *software* no seu ambiente de produção [45];
- **Vista de cenários** – vista que representa os vários casos de uso [45].

As várias vistas deste modelo podem ser observadas na figura 2.15.

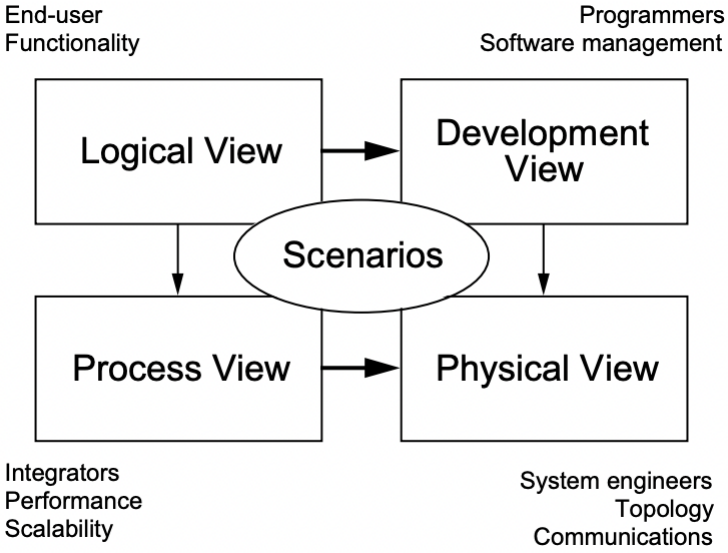


FIGURA 2.15: Vistas do modelo 4+1 [45]

É importante notar que nem sempre é necessário apresentar todas as vistas do modelo 4+1, uma vez que tal depende sempre da arquitetura construída e do que se está a tentar demonstrar. Por exemplo, numa arquitetura onde o *software* se encontra completamente mapeado num só servidor, provavelmente não existe a necessidade de uma vista física [45].

### 2.11.3 Modelo C4

Para se entender com que detalhe se devem apresentar as várias vistas expostas na secção 2.11.2 é utilizado o modelo C4, que acaba por complementar o modelo 4+1. Este modelo pressupõe quatro níveis distintos:

- **Context (Nível 1)** – nível mais elevado e mais abstrato que inclui as dependências chave do sistema e os seus atores [46];
- **Container (Nível 2)** - os diagramas deste nível demonstram as escolhas de tecnologia de alto nível, a forma de como as responsabilidades foram distribuídas e como os *containers* comunicam entre si [46];

- **Component (Nível 3)** - o *component* é um elemento integrante de um *container*. Os diagramas de nível três permitem observar a lógica dos *components* e as suas relações [46];
- **Classes (Nível 4)** - os diagramas de nível quatro são os mais próximos da implementação, sendo geralmente utilizados para explicar como um dado padrão ou componente vai ser implementado [46].

A figura 2.16 ilustra as várias camadas definidas pelo modelo C4.

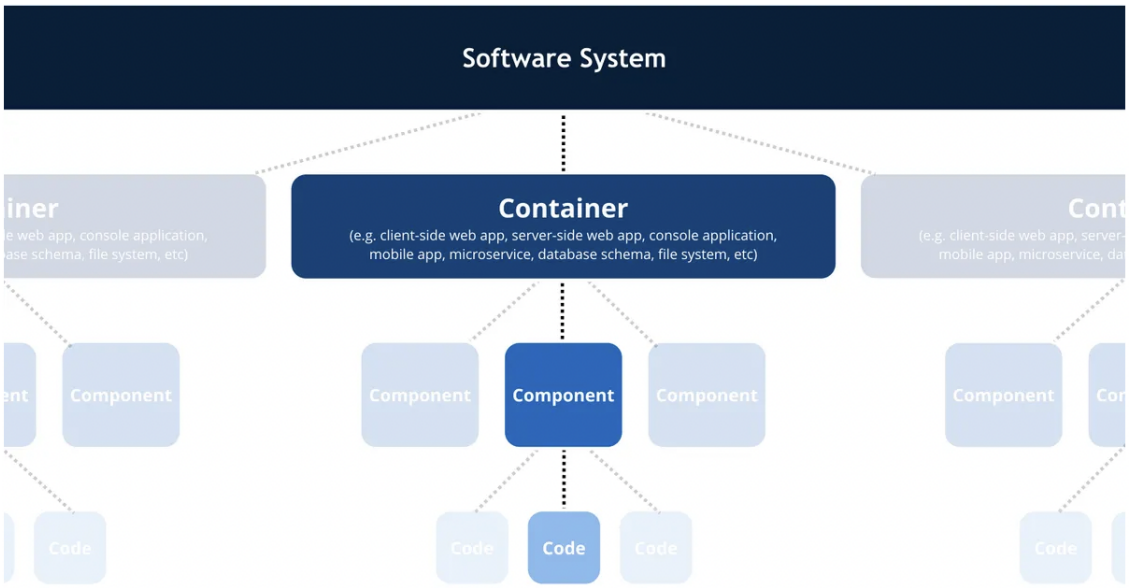


FIGURA 2.16: Modelo C4 [47]

## 2.12 Sumário

Neste projeto, para o desenvolvimento do *backend*, será utilizado Flask e Spring. Flask será utilizado para o desenvolvimento dos módulos que façam uso de ML, enquanto que Spring foi selecionado para a construção do módulo de *analytics*.

Para o desenvolvimento do *frontend* será utilizado React, visto que permite um desenvolvimento mais ágil e, consequentemente, mais rápido.

Em relação às tecnologias de ML, para efetuar a tradução de língua gestual para língua portuguesa, serão utilizadas redes neuronais do tipo RNN com uma arquitetura de LSTM, que serão construídas com recurso a "Tensorflow". Foi decidido utilizar "Tensorflow", visto que é uma ferramenta que oferece uma interface amigável para a construção de redes neuronais, que é *open-source* e disponibiliza imenso suporte, o que, mais uma vez, é crucial para que o desenvolvimento não sofra atrasos indesejados. No que toca ao tipo de NN, as redes RNN, baseadas na arquitetura LSTM, são as que se adequam de melhor forma ao problema em questão, uma vez que outras soluções para realizar traduções, como o "Google Translate", também optam por esta abordagem.

Por último, dado que já existem tecnologias capazes de identificar e reconhecer mãos, não será necessário treinar um modelo para que faça isto, tal como foi analisado na secção 2.7. No entanto, a *framework* a ser utilizada apenas será decidida no capítulo 3, com recurso ao método AHP, que lá será aprofundado.

# Capítulo 3

# Análise de Valor

Não é novidade que o consumidor de um produto ou serviço deseja sempre que este possua o menor custo possível, sem perder o seu valor. Deste modo, este capítulo descreve o processo de análise de valor (AV) para que se identifiquem todos os custos desnecessários, evitando assim investimentos que prejudicam o valor final do produto. Este processo é apresentado e detalhado com base em artefactos que foram construídos recorrendo a várias técnicas próprias, com o intuito final de determinar se existe ou não valor no produto a desenvolver.

## 3.1 Inovação, New Concept Development (NCD)

O termo "inovação" remete para mudança, seja esta consequência da necessidade de algo novo ou da insatisfação com o estado da arte presente. Estas mudanças podem ser estruturais e revolucionárias, originando em conceitos nunca antes vistos ou, simplesmente, alterando processos já existentes com o objetivo de os tornar mais eficientes, por exemplo. No momento da escrita do presente documento, é possível verificar uma inovação tecnológica acentuada, o que leva a uma constante procura de inovação de produtos e serviços que forneçam um aumento de valor ao cliente, pelo que constitui um processo algo crucial na área da informática.

Segundo o livro *The PDMA ToolBook for New Product Development*, o processo da inovação pode ser separado em três partes distintas, como se pode observar na Figura 3.1 [48].

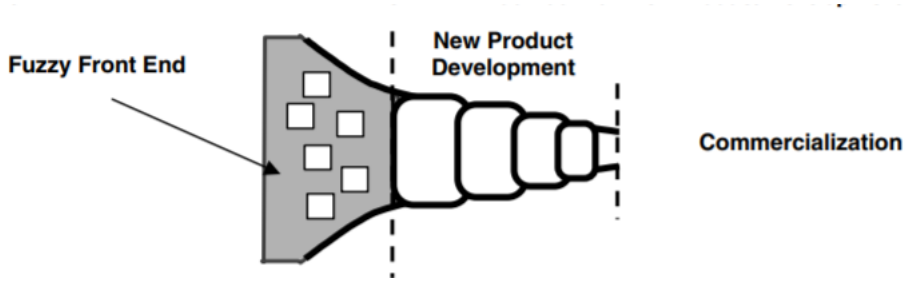


Figura 3.1: Processo de Inovação ([48])

Este processo acompanha o produto desde a sua criação até à sua fase final, a comercialização, sendo que deve ser seguido de forma sequencial, passando por todas as fases pela seguinte ordem [49]:

- *Front End of Innovation (FEI)*
- *New Product Development*
- Comercialização

A primeira etapa no processo de inovação é o FEI. Foi decidido adotar o termo FEI à priori do termo *Fuzzy Front End* presente na Figura 3.1, uma vez que segundo Peter Koen, o segundo implica que o FEI é misterioso, incontrolável e impossível de gerir, o que por sua vez não é verdade [50]. O FEI é o período em que uma nova ideia para um novo produto é gerada e posteriormente desenvolvida. As atividades no FEI são menos estruturadas e menos previsíveis em comparação com as restantes etapas [49].

O segundo passo é o NPD. Esta fase engloba o desenvolvimento do produto, bem como o *marketing* envolvido e outros processos que têm como objetivo de satisfazer os requisitos da estratégia competitiva de uma empresa [51].

Por último, a etapa de comercialização baseia-se na venda do produto/serviço criado, sendo este algo novo ou então uma reinvenção/alteração de outro já existente.

Para fornecer uma terminologia e linguagem comum capaz de otimizar a etapa de FEI, foi desenvolvido o modelo *New Concept Development* (NCD), que se encontra representado na figura 3.2 [50].

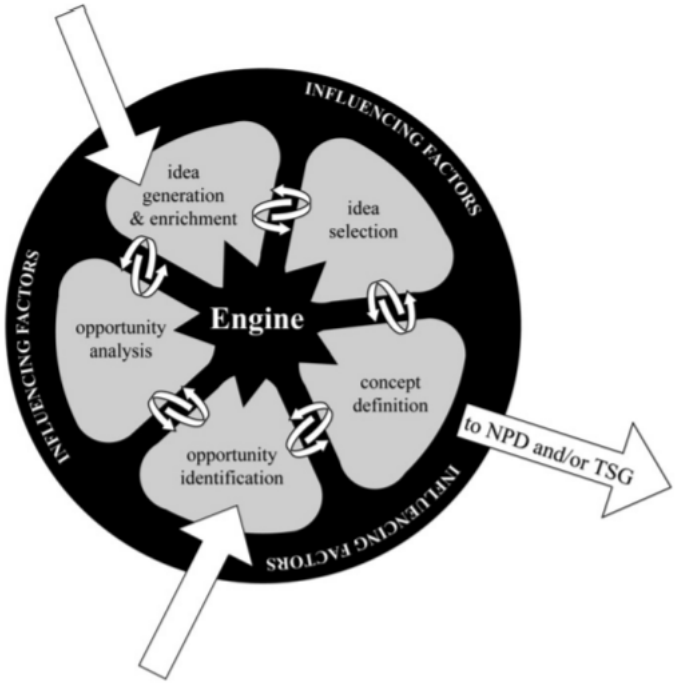


FIGURA 3.2: *New Concept Development Model* ([48])

A partir da figura 3.2 é possível destacar três partes distintas do modelo NCD:

- *Engine* - Representa a liderança, a cultura e a estratégia de negócio da organização, impulsionando os elementos chave [48];
- Cinco elementos chave - Correspondem às totalidade das fases do processo desde a análise de oportunidades até à definição de um conceito (são atividades controladas) [48];
- Fatores Influenciadores - Constituem fatores externos (não controláveis) que afetam o processo de inovação na sua totalidade [48].

Nos pontos seguintes deste capítulo exploram-se em detalhe alguns dos elementos chave que fazem sentido aplicar ao projeto em questão.

### 3.1.1 Identificação de Oportunidade

Os vários avanços tecnológicos possibilitaram que a qualidade de vida do ser humano melhorasse significativamente, tendo revolucionado a nossa forma de viver. No entanto, ainda existem distintos segmentos da sociedade que são marginalizados em relação a todo este desenvolvimento, como é o caso dos surdos (*digital divide*).

A saúde é um bem essencial e o acesso a ela deveria ser fornecido a todos os cidadãos. Ainda assim, a comunidade surda, numa grande maioria, não consegue usufruir dos serviços prestados em hospitais/centros de saúde por várias razões. Este é um problema que existe em vários países por todo o mundo, mesmo naqueles mais desenvolvidos como os Estados Unidos da América . Vários estudos corroboram a afirmação de que os surdos enfrentam bastantes barreiras de comunicação a aceder a serviços de saúde, como medo, desconfiança e até frustração. Existem ainda indivíduos que descrevem um cenário de incomunicabilidade, dificultando a troca de informação e, consequentemente, o tratamento aos pacientes. Um estudo realizado nos Estados Unidos da América concluiu que é recorrente profissionais de saúde fornecerem diagnósticos médicos errados a pacientes surdos, devido a estas barreiras de comunicação, o que pode levar a problemas mais graves. No Reino Unido, um outro estudo demonstra que 44% dos pacientes surdos acredita que o contacto mais recente que tiveram com serviços de saúde foi acompanhado de muitas dificuldades de comunicação, enquanto que apenas 17% da população geral sentiu o mesmo. Outras táticas de comunicação como a leitura de lábios ou até a troca de ideias escrita não são fiáveis, ao contrário do que comummente se faz acreditar. Na realidade, os indivíduos surdos que conseguem ler lábios apenas detetam cerca de 30% a 45% das palavras. No que toca à troca de ideias escritas, certos estudos indicam que a maioria do surdos apresentam níveis de leitura baixos ou até inexistentes [1].

Aliado a estas dificuldades, a comunidade surda tem mais problemas a aceder a informação (pela falta de material disponível em língua gestual), nomeadamente a informação ligada à saúde, o que leva a que estes indivíduos se tornem mais ignorantes nesta vertente. Em algumas línguas gestuais existem até certas doenças que não possuem tradução, como o caso da BSL, onde não existe nenhum gesto para a palavra “colesterol” [1].

Encontram-se ainda outros problemas na comunicação de surdos com profissionais de saúde, relacionados com a vergonha, muito associados a surdos no período da adolescência. Estes, quando possível, são acompanhados por intérpretes mas admitem ter vergonha e embaraço de fazer questões ou revelar problemas quando na presença destes [1].

Com base na informação apresentada ao longo desta secção, é possível concluir que são vários os obstáculos e barreiras existentes que diminuem a independência da comunidade surda, alienando-a de um contexto social essencial para a vida humana. Deste modo, é com base na conjugação desta informação e daquela apresentada na secção 1.2 que se identifica a oportunidade, tentando diminuir o *digital divide* através de uma solução capaz de assistir na tradução da língua gestual para texto, para que os profissionais de saúde consigam entender de forma clara a mensagem a ser passada.

### 3.1.2 Análise de Oportunidade

Depois de identificar a oportunidade, torna-se possível analisar a mesma para que seja possível retirar conclusões sobre a sua implementação. Uma vez que a oportunidade em questão é de natureza digital é necessário primeiro entender-se se os indivíduos surdos oferecem algum tipo de resistência face à sua utilização.

Recorrendo aos exemplos analisados na secção 2.2, é possível notar que já existem várias soluções tecnológicas com o intuito de facilitar a vida da comunidade surda, bem como fornecer-lhes um maior nível de independência. O tradutor móvel "Hand Talk", no momento de escrita deste relatório, conta com mais de trinta e sete mil (37000) *downloads* na *play store* com avaliações bastante positivas, o que demonstra que efetivamente as pessoas surdas recorrem a estas ferramentas. Foram também questionados indivíduos que trabalham diariamente com surdos, como médicos e professores, que relatam que toda a ajuda capaz de facilitar a vida destas pessoas é crucial. Uma professora de escola básica e secundária de Lordelo, indica que "apesar de termos assistido a mudanças que permitiram melhorar as acessibilidades para os surdos nas diferentes áreas da sua vida, considero que há muito a fazer para que se atinja a igualdade de direitos no acesso a bens, serviços e informação. Nem todos os serviços se encontram adaptados por natureza para o atendimento a surdos, de forma a que este possa ser feito sem o auxílio de terceiros. São necessários mais esforços no sentido de dotar os serviços com capacidade de resposta para atender a todas estas necessidades". O médico José Pinto acrescenta ainda, com base na sua experiência, que "todas as ferramentas ao dispor das pessoas surdas para perceber com objetividade e clareza no dia-a-dia as suas necessidades é de estrema relevância e importância. Certamente mais importante e relevante se torna no contexto da saúde, onde a pessoa surda tem de ser cuidada e entendida convenientemente para solucionar o seu problema. Como não é possível ter umas pessoas em todos os serviços que entendam LG, haver aplicações informática que facilitam a vida quer aos surdos, quer aos ouvintes é um passo importante para para a melhoria da comunicação e melhoria dos serviços de saúde aos cidadão surdos". Estas afirmações complementam a informação até aqui recolhida e demonstram que os problemas identificados são a realidade, sendo que até os profissionais de saúde acreditam que a solução para eles devem passar pelo mundo digital.

Pela informação apresentada ao longo desta secção, é possível entender que um tradutor digital de LGP é uma solução pertinente, pelo que faz sentido analisar as tendências deste conceito. Deste modo, o gráfico presente na figura 3.3 apresenta a quantidade de pesquisas *online* por tradutores de LG.


FIGURA 3.3: Gráfico representativo da tendência de pesquisas *online* pelo conceito de tradutor de língua gestual [**gogleTrends**]

Para obter os dados espelhados na figura 3.3 foi utilizada a ferramenta "Google Trends" [52]. Esta ferramenta, apesar de apenas conter informação de um subconjunto do total das pesquisas realizadas no motor de busca "Google" (5 anos), considera-se que este é suficiente para se retirar conclusões fiáveis, tendo em conta os milhões de pedidos diários que este *browser* possui. Assim, a partir da figura 3.3, conclui-se que nos últimos cinco anos a população global apresenta um interesse constante na solução proposta, tendo existido um ligeiro pico em 2018. Deste modo, nasce mais um argumento positivo face à implementação da oportunidade identificada.

## 3.2 Seleção da Framework a utilizar

Como foi mencionado no fim do capítulo do Estado da Arte (2), é necessário decidir entre três *frameworks* distintas: MediaPipe; FingerPose; HandTrackJS, qual a melhor a utilizar para o presente projeto. Uma vez que esta é uma escolha crucial e tendo em conta também a complexidade desta, foi decidido que seria utilizado um método de apoio à decisão multicritério, neste caso o *Analytic Hierarchy Process* (AHP). Este foi o método utilizado, visto que se baseia numa abordagem racional, recorrendo a técnicas matemáticas e a cruzamentos entre alternativas e critérios existentes [53].

Neste método existem sete fases e nesta secção serão todas executadas para o problema em questão:

1. Construção da árvore hierárquica de decisão

2. Comparação das alternativas e critérios
3. Prioridade relativa de cada critério
4. Avaliar a consistência das prioridades relativas
5. Construção da matriz de comparação para cada critério
6. Obter a prioridade composta para as alternativas
7. Escolha da alternativa

### 3.2.1 Construção da árvore hierárquica de decisão

Na primeira fase o objetivo é definir o problema e estruturá-lo num diagrama hierárquico. Como já foi mencionado, o objetivo da utilização deste método é definir qual a *framework* utilizar para o reconhecimento dos gestos. Assim sendo, esta decisão tem de ser feita com base nos seguintes critérios:

- Possibilidade de utilizar a *framework* em várias linguagens de programação;
- Capacidade de detetar duas mãos;
- Capacidade de reconhecer caras;
- Possibilidade de utilização sem recorrer a equipamento especial (como luvas ou câmaras específicas).

O primeiro ponto não é obrigatório mas sem dúvida que é uma mais valia, fazendo com que haja menos limitações futuras no que toca a tecnologias. O segundo ponto é essencial, todo o projeto se baseia na tradução de língua gestual e sem a capacidade de identificar pelo menos duas mãos deixa de ser possível alcançar tal objetivo. A terceira é também muito relevante. Como foi já mencionado na secção 2.1.1, a LGP utiliza também a boca e outras partes do corpo para complementar as mãos, pelo que a possibilidade de reconhecer expressões faciais tornaria a aplicação mais completa. Por último, a não utilização de equipamentos especiais acaba por não ser tão importante como os dois últimos critérios mas é também relevante para a natureza deste relatório. A ideia seria que os utilizadores conseguissem utilizar a aplicação a qualquer momento e a necessidade de utilizar luvas, por exemplo, estragaria este propósito.

As alternativas a considerar são as que foram mencionadas na secção 2.7: MediaPipe, FingerPose e HandTrackJS.

Tendo em conta o objetivo final e todos os critérios enunciados, foi construída a árvore hierárquica que pode ser observada na figura 3.4.

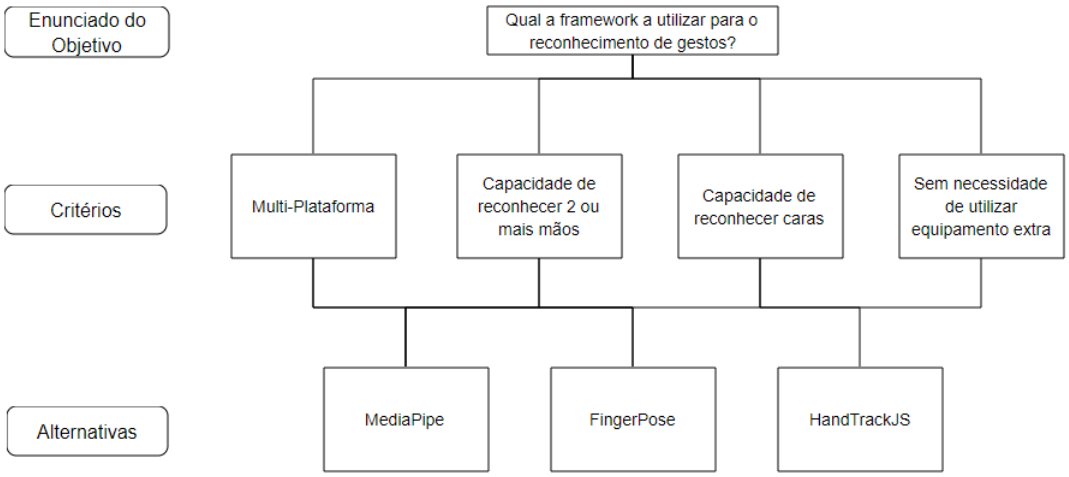


FIGURA 3.4: Estrutura hierárquica do método AHP

### 3.2.2 Comparação das alternativas e critérios

A segunda fase do AHP consiste em estabelecer prioridades entre os elementos para cada nível da hierarquia, através de uma matriz de comparação. Para tal, é necessário utilizar uma escala, sendo que neste caso foi optado por utilizar a escala de Saaty [1]. Deste modo, foi possível construir a matriz de comparação entre critérios, que se encontra representada na tabela 3.1, onde:

- A: Possibilidade de utilizar a *framework* em várias linguagens de programação;
- B: Capacidade de detetar duas mãos;
- C: Capacidade de reconhecer caras;
- D: Possibilidade de utilização sem recorrer a equipamento especial.

TABELA 3.1: Matriz de comparação entre critérios

| | **A** | **B** | **C** | **D** |
|---|---|---|---|---|
| A | 1 | 0.11 | 0.14 | 0.17 |
| B | 9 | 1 | 3 | 5 |
| C | 7 | 0.33 | 1 | 2 |
| D | 6 | 0.2 | 0.5 | 1 |

Analisando a tabela 3.1, é possível concluir que os critérios de reconhecimento de pelo menos duas mãos é de importância absoluta em relação à possibilidade de utilizar a ferramenta em várias linguagens de programação. É também levemente mais importante que a capacidade de reconhecer caras e mais importante que a não utilização de equipamentos especiais. Por sua vez, a capacidade de detetar caras é de muito forte importância em comparação ao primeiro critério, sendo levemente mais importante que o último. Para terminar, é possível também inferir que a não

[1] Escala proposta por Thomas L. Saaty para a utilização do método AHP [53]

utilização de equipamento especial acaba por ser mais importante que a possibilidade de ser multi-plataforma.

### 3.2.3 Prioridade relativa de cada critério

Na terceira fase do método AHP, o objetivo é identificar a prioridade relativa de cada critério. Para tal, é preciso normalizar os valores da matriz de comparações, representada na tabela 3.1 e, posteriormente, obter-se o vetor de prioridades respetivo. A tabela 3.2 apresenta a matriz já normalizada, seguida do vetor de prioridades, calculado através da média aritmética de cada linha da matriz.

Tabela 3.2: Matriz normalizada de comparação entre critérios e vetor de prioridades relativas

| | **A** | **B** | **C** | **D** | **Prioridade Relativa** |
|---|---|---|---|---|---|
| A | $\frac{1}{23}$ | $\frac{5}{74}$ | $\frac{1}{32}$ | $\frac{1}{49}$ | 0.04 |
| B | $\frac{9}{23}$ | $\frac{45}{74}$ | $\frac{21}{32}$ | $\frac{30}{49}$ | 0.57 |
| C | $\frac{7}{23}$ | $\frac{15}{74}$ | $\frac{7}{32}$ | $\frac{12}{49}$ | 0.24 |
| D | $\frac{6}{23}$ | $\frac{9}{74}$ | $\frac{7}{64}$ | $\frac{6}{49}$ | 0.62 |

### 3.2.4 Avaliar a consistência das prioridades relativas

Esta etapa implica a avaliação da consistência das prioridades relativas. Para isso, é necessário calcular a razão de consistência (RC) para medir o quanto os julgamentos foram consistentes em relação a grandes amostras de juízos completamente aleatórios, que pode ser calculada com base na fórmula:

$$RC = \frac{IC}{IR}$$

Caso o valor de RC seja inferior a 0,1 (ou 10%), pode-se concluir que os valores estão consistentes. Na fórmula anterior, IC corresponde ao índice de consistência, enquanto que IR representa o índice aleatório obtido a partir da tabela de índice aleatório, que se encontra ilustrada na tabela 3.3:

Tabela 3.3: Valores de IR para matrizes quadradas de ordem n

| **Dimensão da matriz** | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|---|---|---|---|---|---|---|---|---|
| **Índice de consistência** | 0.00 | 0.00 | 0.58 | 0.90 | 1.12 | 1.24 | 1.32 | 1.41 |

No caso presente, a ordem da matriz é 4, pelo que o valor do IR é 0.90.

Para o cálculo do IC, é necessário aplicar a seguinte fórmula:

$$IC = \frac{(\lambda max - n)}{(n-1)}$$

Com esta informação é possível realizar os cálculos que restam nesta etapa. Primeiro, multiplica-se a matriz de comparação pelo vetor de prioridades, obtendo-se a matriz presenta na tabela 3.4:

Tabela 3.4: Matriz resultante da multiplicação do vetor prioridade com a matriz de comparação

| | |
|---|---|
| **A** | 0.16 |
| **B** | 2.40 |
| **C** | 1.01 |
| **D** | 0.62 |

A partir da média dos valores espelhados na tabela 3.4 e consequente divisão destes pelos valores do vetor de prioridades relativas, é possível calcular o valor de $\lambda max$, como demonstra a seguinte equação:

$$lambdamax = \frac{0.16/0.04 + 2.40/0.57 + 1.01/0.24 + 0.62/0.15}{4} = 4.13$$

Com o $\lambda max$, é possível calcular o IC:

$$IC = \frac{(4.13 - 4)}{(4 - 1)} = 0.04 = 4$$

Por último, aplica-se a fórmula da razão de consistência:

$$RC = \frac{(IC)}{(IR)} = \frac{0.04}{0.90} = 0.04$$

Como $0.04 < 0.1$, é possível concluir que os valores de prioridades relativas se encontram consistentes.

### 3.2.5 Construção da matriz de comparação para cada critério

A quinta fase do AHP baseia-se na construção de matrizes de comparação cruzando todas as alternativas existentes com cada critério de forma individual, onde:

- A': Utilização de MediaPipe
- B': Utilização de Fingerpose
- C': Utilização de HandTrackJS

Assim, para o critério A obtém-se a matriz representada na tabela 3.5:

Tabela 3.5: Matriz de comparação para o critério A

| | **A'** | **B'** | **C'** | **Vetor Prioridade** |
|---|---|---|---|---|
| A' | 1 | 4 | 4 | 0.67 |
| B' | $\frac{1}{4}$ | 1 | 1 | 0.17 |
| C' | $\frac{1}{4}$ | 1 | 1 | 0.17 |

Para o critério B obtém-se a matriz representada na tabela 3.6:

Tabela 3.6: Matriz de comparação para o critério B

| | **A'** | **B'** | **C'** | **Vetor Prioridade** |
|---|---|---|---|---|
| A' | 1 | 7 | 1 | 0.47 |
| B' | $\frac{1}{7}$ | 1 | $\frac{1}{7}$ | 0.07 |
| C' | 1 | 7 | 1 | 0.47 |

Para o critério C obtém-se a matriz representada na tabela 3.7:

Tabela 3.7: Matriz de comparação para o critério C

| | **A'** | **B'** | **C'** | **Vetor Prioridade** |
|---|---|---|---|---|
| A' | 1 | 7 | 4 | 0.68 |
| B' | $\frac{1}{7}$ | 1 | $\frac{1}{5}$ | 0.07 |
| C' | $\frac{1}{4}$ | 5 | 1 | 0.25 |

Para o critério D obtém-se a matriz representada na tabela 3.8:

Tabela 3.8: Matriz de comparação para o critério D

| | **A'** | **B'** | **C'** | **Vetor Prioridade** |
|---|---|---|---|---|
| A' | 1 | 1 | 1 | 0.33 |
| B' | 1 | 1 | 1 | 0.33 |
| C' | 1 | 1 | 1 | 0.33 |

Analisando as tabelas 3.5, 3.6, 3.7 e 3.8, é possível entender a importância relativa de cada uma das alternativas apresentadas. No que diz respeito à possibilidade de integrar a *framework* com várias linguagens de programação, a alternativa A' seria a mais adequada. Em relação ao critério da capacidade de reconhecer pelo menos duas mãos, tanto a alternativa A' como a B' seriam adequadas. Para o critério do reconhecimento facial, a alternativa A' acaba por apresentar uma ligeira vantagem. Por último, para o critério relativo à utilização sem equipamentos especiais, todas as alternativas seriam adequadas, tendo resultado numa matriz uniforme.

### 3.2.6 Obter a prioridade composta para as alternativas

A sexta e penúltima fase, consiste em obter as prioridades compostas das alternativas, multiplicando os valores obtidos na secção 3.2.5 pelos valores das prioridades relativos, obtidos no início do método AHP. Para isto, o primeiro passo é juntar os vetores de prioridades provenientes das matrizes de comparação das alternativas por critérios numa só matriz, representada na tabela **??**.

Tabela 3.9: Matriz de prioridades relativas das alternativas

| | **A** | **B** | **C** | **D** |
|---|---|---|---|---|
| **A'** | 0.67 | 0.47 | 0.68 | 0.33 |
| **B'** | 0.17 | 0.07 | 0.07 | 0.33 |
| **C'** | 0.17 | 0.47 | 0.25 | 0.33 |

Multiplicando a matriz retratada na tabela 3.9 pelo vetor de prioridades relativas dos critérios, presentes na tabela 3.2, obtém-se o vetor de prioridades representado na tabela :

Tabela 3.10: Vetor de Prioridades Compostas

| | |
|---|---|
| **A'** | **0.51** |
| **B'** | 0.11 |
| **C'** | 0.38 |

### 3.2.7 Escolha da alternativa

A última fase do AHP consiste simplesmente na escolha da alternativa mais apropriada para o problema a resolver. Deste modo, com base nos critérios definidos, é possível concluir que a utilização da *framework* MediaPipe seria a mais adequada para a identificação de gestos.

## 3.3 Proposta de Valor

Uma Proposta de Valor (PV) é utilizada com o intuito de expressar as motivações que os clientes/utilizadores possuem para darem preferência a um determinado produto/serviço, tendo em conta as alternativas existentes no mercado. Uma PV deve ser clara, sucinta e intuitiva, de modo que quando esta é lida por alguém toda a mensagem é transmitida imediatamente. O objetivo é que seja fácil de memorizar, devido à brevidade e objetividade dos pontos levantados. Para além disto, uma PV deve ser única, sendo construída sempre de raiz tendo em conta o caso específico a ser avaliado. Assim, é possível distinguir o produto/serviço em questão de outros que lhe façam concorrência [54].

Para representar a proposta de valor aplicada ao contexto em questão, foi utilizada a ferramenta *Value Proposition Canvas*, que pode ser analisada na figura 3.5.

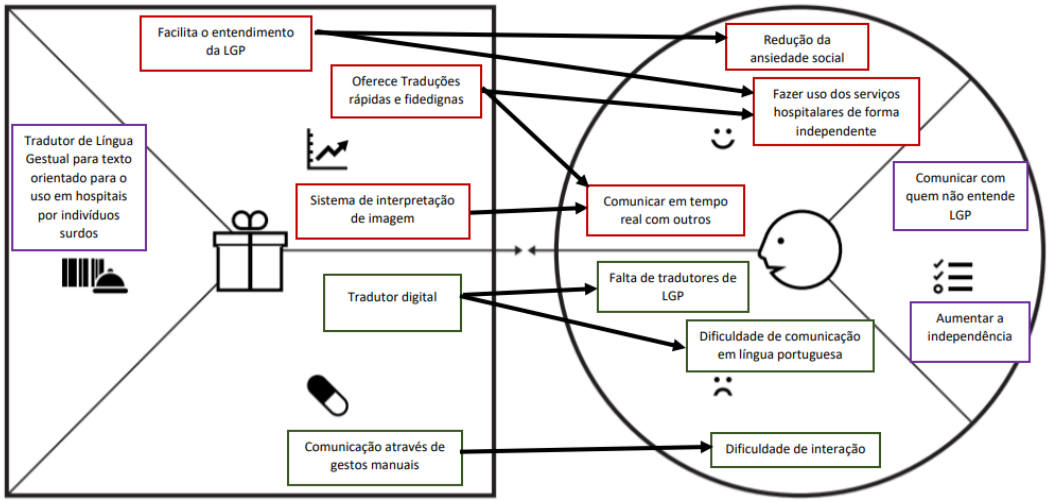


FIGURA 3.5: Proposta de Valor

A partir da 3.5 é possível concluir que existe uma relação evidente entre o produto a desenvolver e as necessidades do cliente. Assim, tudo leva a querer que o projeto proposto seria uma mais valia para o cliente.

O projeto enunciado na PV consiste na construção de um tradutor digital de LGP orientado para hospitais que seja capaz de facilitar a comunicação entre indivíduos surdos e o resto da sociedade, bem como diminuir a ansiedade social que estes sentem em contextos públicos por não entenderem quem os rodeia (nomeadamente uma ida ao hospital) e aumentar a sua independência no que toca a direitos básicos, como a saúde.

## 3.4 Quality Function Deployment

*Quality Function Deployment* (QFD) é um conceito geral que fornece uma forma de traduzir os requisitos do cliente em requisitos técnicos apropriados para cada fase de desenvolvimento do produto [55]. Posto isto, esta técnica foi também aplicada ao presente projeto, sendo que o diagrama resultante pode ser observado na figura 3.6.

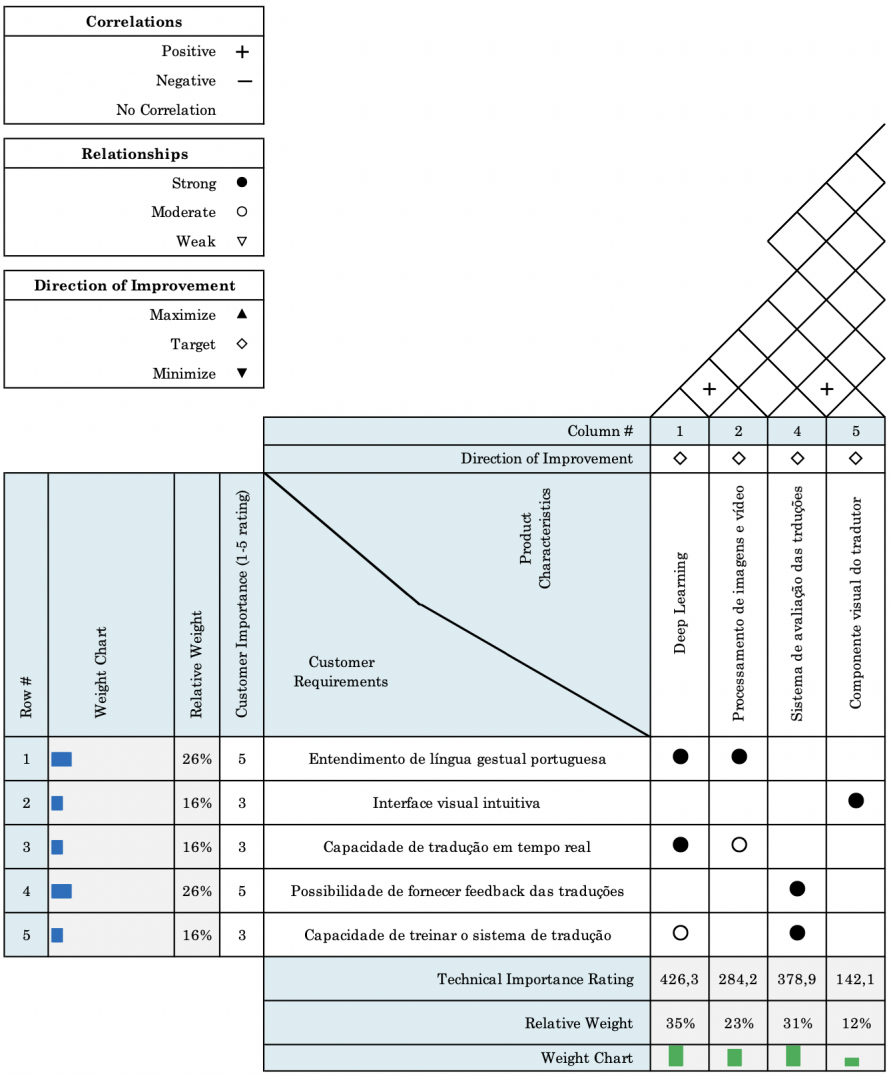


FIGURA 3.6: Quality Function Deployment

As necessidades do cliente foram recolhidas através de discussões com indivíduos que trabalham ativamente com indivíduos da comunidade surda e podem ser consultadas na parte esquerda do diagrama representado na figura 3.6.

Em contrapartida, as características técnicas encontram-se expostas no canto superior direito do mesmo diagrama.

Os símbolos utilizados possuem a sua legenda no canto superior esquerdo da figura 3.6.

A partir dos resultados obtidos, entende-se que os requisitos mais importantes para os utilizadores são a tradução efetiva da LGP e a possibilidade de fornecer *feedback* das traduções obtidas. Os restantes requisitos identificadas são ligeiramente menos importantes.

No que toca ao requisitos técnicos, a implementação de *deep learning* é o que se destaca, tendo correlação com vários requisitos do cliente, seguido pelo sistema de avaliação das traduções e também pelo processamento de imagens e vídeo.

# Capítulo 4

# Análise de Negócio

Este capítulo aborda a análise de domínio do projeto proposto. Inicialmente, realiza-se a engenharia de requisitos, na qual são identificados e explorados os requisitos funcionais e não funcionais levantados, sendo estes últimos representados com base no modelo FURPS+. De seguida, apresenta-se o modelo de domínio onde se encontram registados os conceitos de negócio.

## 4.1 Requisitos Funcionais

Os requisitos funcionais de um sistema definem o que este deve ser capaz de realizar, bem como se deve comportar em determinadas situações. Estes requisitos dependem, por exemplo, do tipo de software a ser desenvolvido e de quem utiliza o sistema. Os requisitos funcionais são, geralmente, descritos de forma abstrata, de forma a que sejam facilmente entendidos [56].

Este processo é de importância elevada para a qualidade do projeto, dado que existem várias áreas do desenvolvimento que são afetadas pelos resultados obtidos na Engenharia de Requisitos. Sendo assim, toda a documentação necessária a este método é elaborada e disponibilizada nesta secção no formato de *user stories*. Esta informação foi levantada de acordo a classificação FURPS+, um modelo utilizado para classificar a qualidade de *software*. O FURPS separa os requisitos por diferentes categorias, sendo que os requisitos funcionais prendem-se exclusivamente com a primeira destas, a "Funcionalidade"[57].

**US01** Como administrador quero autenticar-me no sistema;

**US02** Como administrador quero sair do sistema;

**US03** Como utilizador quero que o sistema realize a tradução de LGP para língua portuguesa escrita;

**US04** Como administrador pretendo treinar o sistema para que este seja mais preciso e capaz de detetar mais gestos de LGP;

**US05** Como utilizador quero preencher inquéritos de satisfação e usabilidade para demonstrar o meu agrado/desagrado com o sistema nestes tópicos.

## 4.2 Requisitos Não Funcionais

Os requisitos não funcionais constituem requisitos que não estão diretamente relacionados com os serviços fornecidos aos utilizadores de um dado sistema. Estes requisitos surgem pelas necessidades dos utilizadores, como por exemplo, restrições de orçamento, políticas organizacionais, necessidade de interoperabilidade com outros sistemas de *software* ou *hardware*, ou a partir de fatores externos, como regulamentos de segurança ou legislações de privacidade [56].

Nesta secção enunciam-se os requisitos não funcionais do sistema proposto a desenvolver, utilizando a classificação FURPS+, explorada na secção 4.1. O FURPS, no que toca aos requisitos não funcionais, separa-os por diferentes categorias como: Usabilidade; Confiabilidade; Desempenho e Suportabilidade. Mais tarde, o "+" foi acrescentado, anexando várias categorias distintas das previamente existentes, completando assim este modelo [57].

Para facilitar a compreensão do leitor, apresenta-se na tabela 4.1, os vários requisitos não funcionais identificados, indicando para cada um a categoria em que se enquadra no modelo FURPS+.

Tabela 4.1: Requisitos não funcionais

| Requisito | Categoria |
|---|---|
| RNF-01: Interface gráfica intuitiva e simples, onde deve ser possível interagir com o sistema através de gestos | Usabilidade |
| RNF-02: O tempo de resposta deve ser adequado para que a experiência de uma conversa real não se perca | Desempenho |
| RNF-03: O sistema deve ser compatível com dispositivos móveis e fixos | Suportabilidade |
| RNF-04: O sistema deve ser desenvolvido como uma PWA | +: Requisitos de implementação |
| RNF-05: O sistema deve funcionar com qualquer tipo de dispositivo, móvel ou fixo, desde que este contenha um componente de captura de imagem | +: Requisitos físicos |
| RNF-06: Utilizar boas práticas na construção do *design* arquitetural do projeto | +: Requisitos de *design* |

## 4.3 Modelo de Domínio

Para que o leitor consiga compreender o domínio do sistema, apresenta-se na figura 4.1 o modelo de domínio construído para o presente projeto.

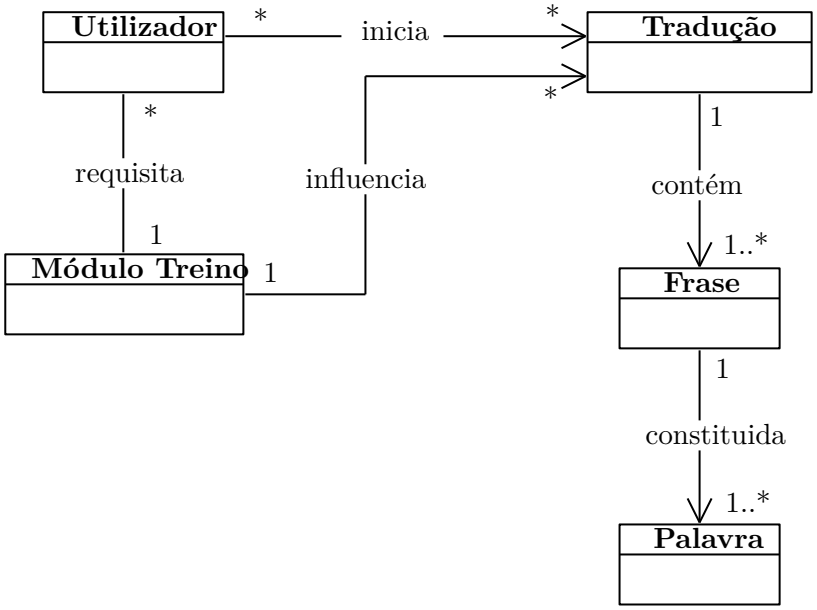


FIGURA 4.1: Modelo de Domínio

O diagrama ilustrado na figura 4.1 foi modelado com base na notação *Unified Modeling Language* (UML) e segundo a interpretação do autor Eric Evans. Este defende que o modelo de domínio é definido como a ideia que é suposto transmitir, sendo uma abstração rigorosamente organizada e selectiva do conhecimento dos peritos do domínio. Neste tipo de diagramas identificam-se classes concetuais, bem como objetos reais do domínio de interesse, que refletem conceitos de domínio e as interações que estes possuem entre si. Estes podem ou não transformar-se em elementos de *software* [44]. Os conceitos de domínio representados na figura 4.1 são explorados individualmente nos seguintes pontos.

**Utilizador** "Utilizador", tal como o nome indica, é um conceito que representa o utilizador do sistema. Um utilizador tem a capacidade de realizar traduções, sendo que o resultado desta ser-lhe-à apresentado imediatamente. Para além disto um utilizador pode também interagir com o módulo de treino de modo a melhorar o sistema..

**Tradução** O conceito "tradução" representa o processo de interpretar LGP e transformá-la em língua portuguesa escrita. Esta tradução é constituída por frases, que por sua vez são constituídas por palavras. O resultado desta operação é apresentado no ecrã ao utilizador.

**Módulo Treino** O "Módulo Treino" é um módulo responsável por treinar o sistema de tradução, de modo a "fortalecê-lo" e a dar-lhe mais precisão e confiança nas traduções efetuadas. Este módulo apenas pode ser acedido por um utilizador

com credenciais específicas, enquanto que o resto da aplicação não necessita de qualquer tipo de autenticação.

## 4.4 Sumário

Foram identificados cinco requisitos funcionais distintos, detalhados no formato de *user story*, apresentados no seguintes pontos:

- **US01** Como administrador quero autenticar-me no sistema;
- **US02** Como administrador quero sair do sistema;
- **US03** Como utilizador quero que o sistema realize a tradução de LGP para língua portuguesa escrita;
- **US04** Como administrador pretendo treinar o sistema para que este seja mais preciso e capaz de detetar mais gestos de LGP;
- **US05** Como utilizador quero preencher inquéritos de satisfação e usabilidade para demonstrar o meu agrado/desagrado com o sistema nestes tópicos.

Foram ainda definidos seis requisitos não funcionais, seguindo o modelo FURPS+. Estes podem ser visualizados de seguida:

- RNF-01: Interface gráfica intuitiva e simples, onde deve ser possível interagir com o sistema através de gestos;
- RNF-02: O tempo de resposta deve ser adequado para que a experiência de uma conversa real não se perca;
- RNF-03: O sistema deve ser compatível com dispositivos móveis e fixos;
- RNF-04: O sistema deve ser desenvolvido como uma PWA;
- RNF-05: O sistema deve funcionar com qualquer tipo de dispositivo, móvel ou fixo, desde que este contenha um componente de captura de imagem;
- RNF-06: Utilizar boas práticas na construção do design arquitetural do projeto.

Por último, foi construído o modelo de domínio para que fosse possível melhor compreender o domínio do sistema. Este é composto por cinco conceitos distintos: Utilizador; Módulo Treino; Tradução; Frase; Palavra.

# Capítulo 5

# Design

Nesta secção é analisado o *design* arquitetural desenvolvido para a solução do problema. Inicialmente, são detalhadas alternativas arquiteturais, prosseguindo-se para a escolha da mais pertinente. De seguida, são apresentados vários diagramas tendo em conta os modelos 4+1 e C4 cf. 2.11.2 e 2.11.3, respetivamente.

## 5.1 Arquitetura

Nesta secção apresentam-se duas alternativas de *design* que procuram solucionar o mesmo problema, embora de formas diferentes. Ambas as alternativas são apresentadas em alto nível, uma vez que se pretende considerar as soluções sem aprofundar os seus componentes.

### 5.1.1 Segregação de Responsabilidades

A primeira alternativa de *design* proposta baseia-se na segregação das responsabilidades no que toca à persistência dos dados. Nesta abordagem, são utilizados dois tipos de bases de dados, como é possível verificar na figura 5.1.

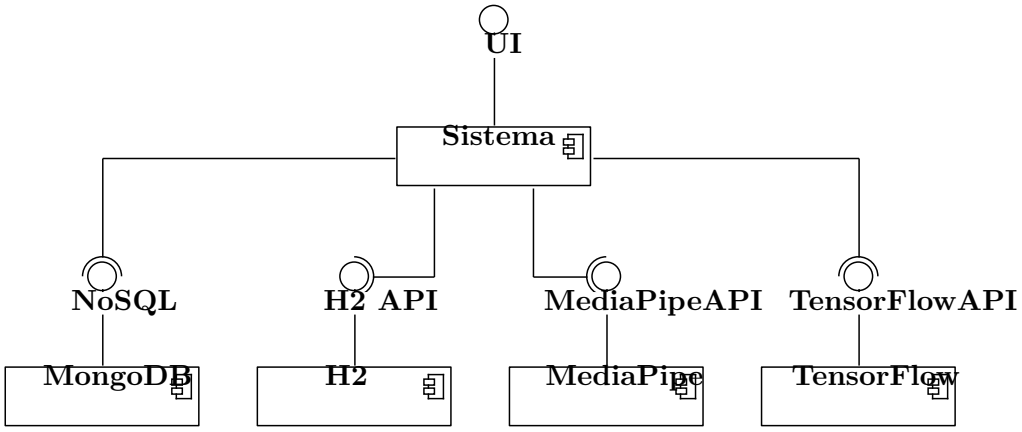


Figura 5.1: Vista lógica de nível 1 conforme os modelos 4+1 e C4 da alternativa 1 em UML

Na figura 5.1, apresenta-se o sistema, representado pelo componente "Sistema", e as suas ligações a sistemas externos. Neste cenário, o sistema oferece apenas uma *Application Program Interface* (API), a "UI". Esta representa a interface gráfica fornecida aos utilizadores da aplicação. Em relação aos sistemas externos, são utilizados 4:

**MediaPipe** *framework* utilizada para processar e identificar os gestos feitos pelo utilizador, tendo já sido analisado na secção 2.7.1 e selecionada na secção 3.2.5;

**TensorFlow** ferramenta utilizada para a construção de aplicações que façam uso de *deep learning*, explorada na secção 2.6.1.

**MongoDB** base de dados não relacional, já investigada na secção 2.10.1, utilizada neste caso para armazenar dados dos utilizadores e, eventualmente, dados que venham a ser necessários no futuro. Existirá apenas um componente no sistema dedicado a interagir com esta base de dados.

**H2** base de dados relacional, já estudada na secção 2.10.3. Neste caso será utilizada para armazenar os dados dos inquéritos de satisfação e utilização, sendo que apenas um componente do sistema será responsável por comunicar com ela.

As vantagens desta solução prendem-se essencialmente com uma maior facilidade na manutenção, uma vez que existe uma maior separação de responsabilidades entre os componentes do sistema, no sentido em que é mais fácil localizar o problema e alterações feitas num local não irão afetar os restantes. Permite também que haja uma maior ganho na escalabilidade, pelo uso de várias tecnologias e de bases de dados de tipos diferentes (relacional e não relacional). No entanto, todo o esforço é maior, uma vez que é uma solução mais complexa.

### 5.1.2 Centralização de API

A segunda alternativa de *design*, ao contrário da identificada na secção 5.1.1, foca-se na centralização das responsabilidades em relação às bases de dados. Para esta alternativa a vista lógica de nível 1 do sistema foi montada apenas com um tipo de bases de dados, como é possível visualizar na figura 5.2.

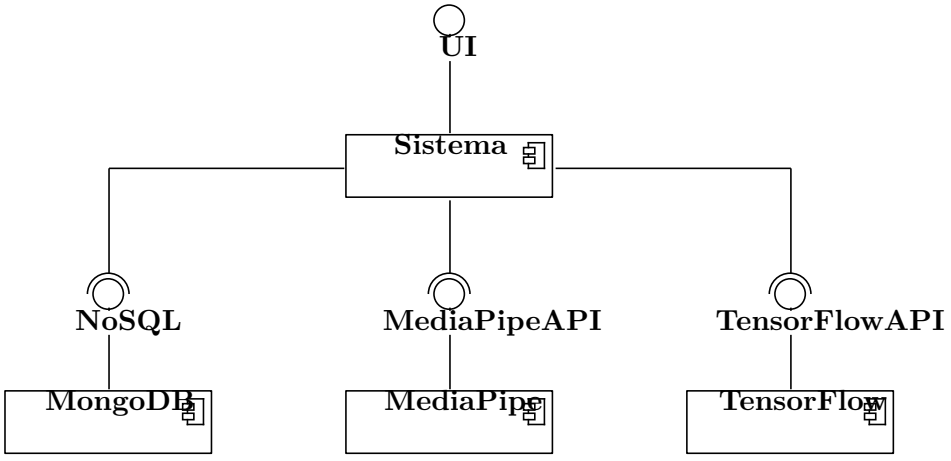


Figura 5.2: Vista lógica de nível 1 conforme os modelos 4+1 e C4 da alternativa 2 em UML

No diagrama exposto na figura 5.2, representa-se novamente o sistema e as suas ligações, sendo que não existem grandes diferenças para aquele exposto na 5.1. A única diferença notável é a utilização apenas de MongoDB no departamento da persistência, pelo que as responsabilidades de interação com a base de dados ficariam todas isoladas num só componente do sistema.

Deste modo todos os dados a serem persistidos ficariam armazenados na mesma base de dados, havendo uma gama de tecnologias utilizadas menor, diminuindo consequentemente a complexidade de implementação. No entanto, a centralização implica que todos os dados sejam persistidos na mesma base de dados, englobando entidades de domínio e de *analytics* (inquéritos). Esta junção implica uma dificuldade maior no que toca a manutenções futuras. Para além disto, pode haver ainda uma diminuição de *performance*, uma vez que uma base de dados poderá ficar sobrecarregada com todos os dados e o princípio de *Separation of Concerns* (SoC), que defende que o *software* deve ser decomposto de tal forma que diferentes preocupações do problema em questão sejam resolvidos em módulos distintos e mais simples [58], não é respeitado.

### 5.1.3 Avaliação e Escolha da Arquitetura

Ambas as alternativas apresentadas são válidas e aplicáveis para solucionar o problema em questão. No entanto, cada uma delas possui vantagens e desvantagens que conforme o objetivo as torna melhores ou piores.

A primeira alternativa visa a segregação das responsabilidades de persistência de dados, utilizando uma base de dados apenas para fins estatísticos (inquéritos de satisfação e usabilidade) e outra para armazenar todos os dados necessários da aplicação (como informação dos utilizadores por exemplo), respeitando o princípio de SoC. Assim, o sistema contém componentes separados para interagir com cada tipo de base de dados, diminuindo o acoplamento. Para além disto existem ainda vantagens no que toca à manutenção futura da aplicação. A localização do erro é mais fácil e a propagação deste fica contida no próprio módulo, sendo que alterações futuras também correm menos riscos. Deste modo a escalabilidade sofre também um aumento. A *performance* neste cenário sofre também uma melhoria, visto que cada base de dados fica menos sobrecarregada com informação, sendo mais rápido aceder aos dados pretendidos. Por outro lado, a segunda alternativa que tem como objetivo centralizar as responsabilidades de comunicação com a base de dados é mais simples no departamento da implementação, possibilitando um fluxo de trabalho mais direto e possui menos um ponto de falha e menos um serviço a manter.

Contemplando os pontos levantados, considera-se que a primeira alternativa é a melhor para o caso em questão, sendo que a capacidade de manutenção futura, permitindo uma gestão mais fácil, e a maior facilidade em expandir o projeto tornaram-se argumentos vencedores.

## 5.2 Nível 1

Nesta e nas seguintes secções é detalhado o *design* arquitetural da alternativa 1 (Segregação de Responsabilidades) seleccionada na secção 5.1.3, fazendo uso dos

modelos 4+1 e C4. A presente secção foca-se em apresentar ao leitor o desenho arquitetural de nível 1.

### 5.2.1 Vista Lógica

A vista lógica de nível 1 foi já apresentada na figura 5.1, tendo sido posteriormente analisada ao longo da secção 5.1.1, pelo que a mesma informação não será repetida.

### 5.2.2 Vista de Cenários

A vista de cenários de nível 1 corresponde ao diagrama de casos de uso, que pode ser visualizado na figura 5.3.

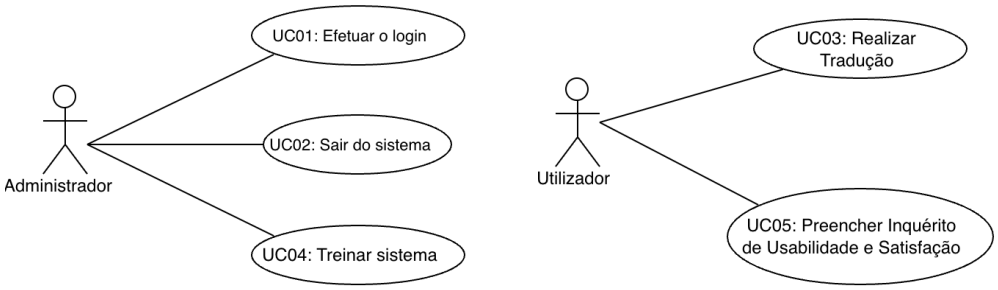


FIGURA 5.3: Vista de Cenários de Nível 1 conforme os modelos 4+1 e C4 em UML

### 5.2.3 Vista de Processos

De modo a demonstrar como os vários componentes interagem entre si, para o nível 1 do modelo C4 serão apresentadas as vistas de processos de todos os casos de uso. No entanto, para os níveis de granularidade mais fina não é necessário apresentar a vista de processos para cada um dos casos de uso existentes, pelo que foi selecionada a funcionalidade "Realizar Tradução". Este caso de uso aparenta ser abrangente e completo ao ponto de, ao longo deste capítulo, expor a informação relativa a esta vista.

**UC01: Efetuar o *Login***

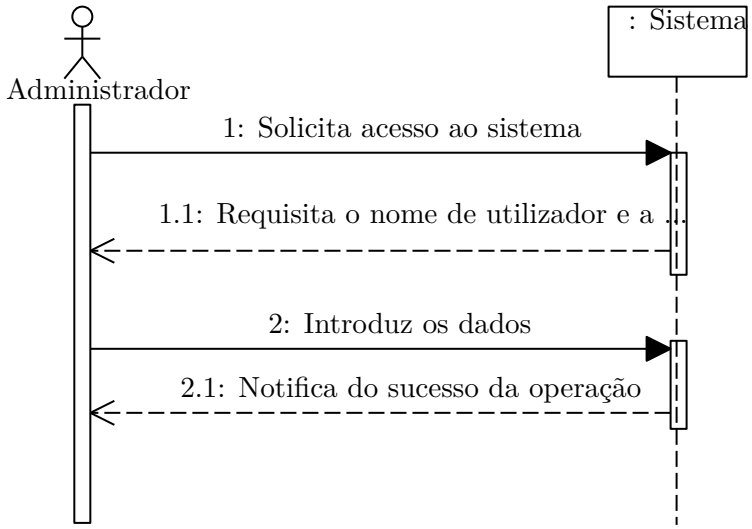


FIGURA 5.4: Vista de Processos do caso de uso 01 em UML

A figura 5.4 espelha o fluxo do primeiro caso de uso. O ator deste caso de uso é o administrador que tem como objetivo autenticar-se no sistema. Para tal, este requisita o seu nome de utilizador e respetiva palavra-chave. Depois do administrador introduzir os dados necessários, o sistema notifica-o do sucesso da operação.

**UC02: Sair do sistema**

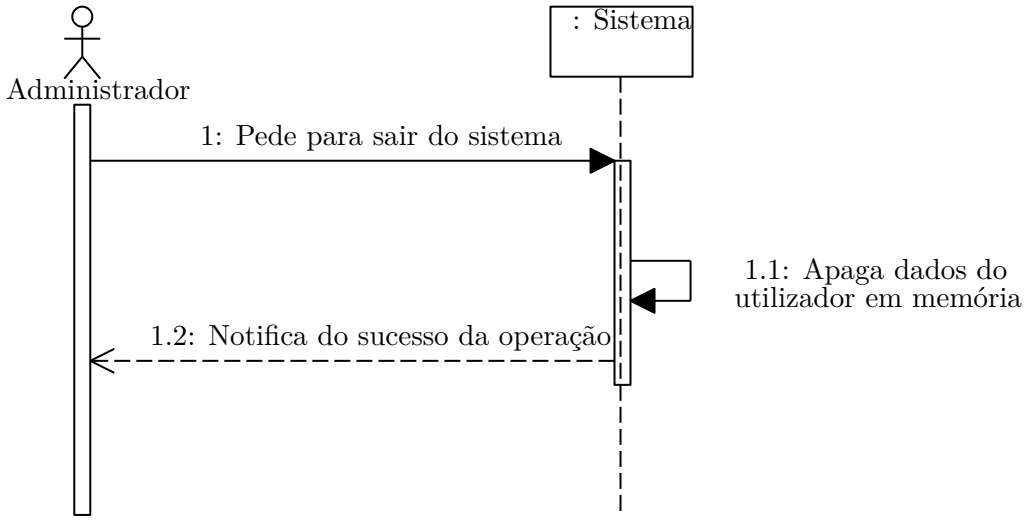


FIGURA 5.5: Vista de Processos do caso de uso 01 em UML

Na figura 5.5 é possível observar a interação entre o administrador e o sistema para que o primeiro possa realizar o *logout*. Para tal, o administrador apenas necessita de selecionar a opção para sair e o sistema trata de remover toda a informação que possua dele em memória local para evitar problemas de segurança. Por último, o sistema notifica o administrador do sucesso da operação.

**UC03: Realizar Tradução**

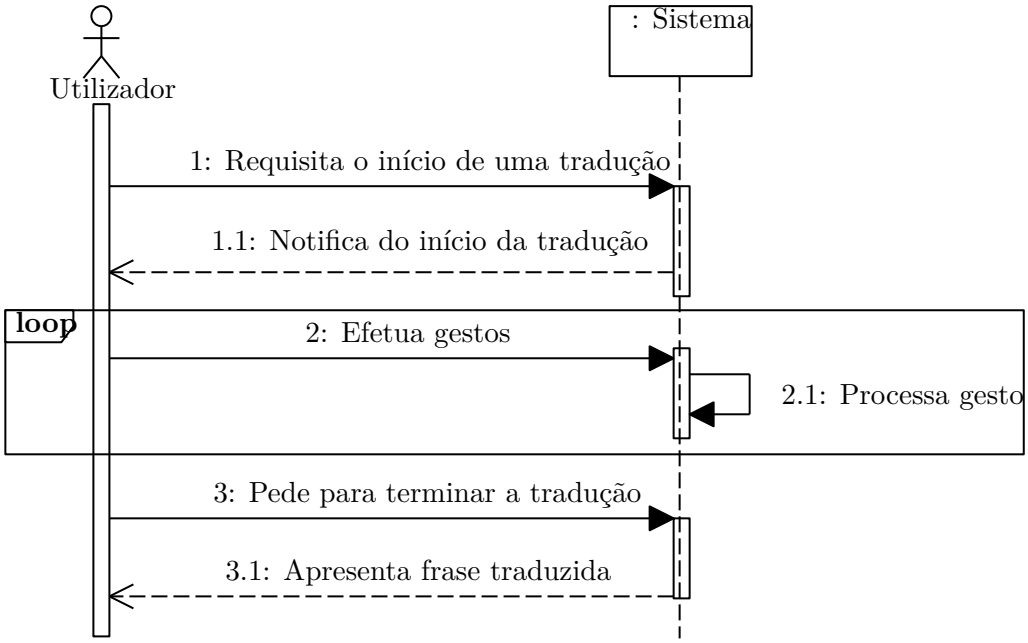


FIGURA 5.6: Vista de Processos do caso de uso 01 em UML

Na figura 5.6, verifica-se o fluxo da interação entre o utilizador e o sistema para realizar uma tradução. O utilizador faz esta solicitação carregando no botão de início. De seguida, o sistema liga a câmara do dispositivo e inicia a operação de tradução. O sistema notifica o utilizador que a tradução foi iniciada e que todos os gestos vão passar a ser identificados com o intuito de os traduzir. O utilizador efetua os gestos que desejar e quando tiver terminado, carrega no botão para desligar a câmara. O sistema processa os gestos e apresenta a tradução no ecrã.

**UC04: Treinar Sistema**

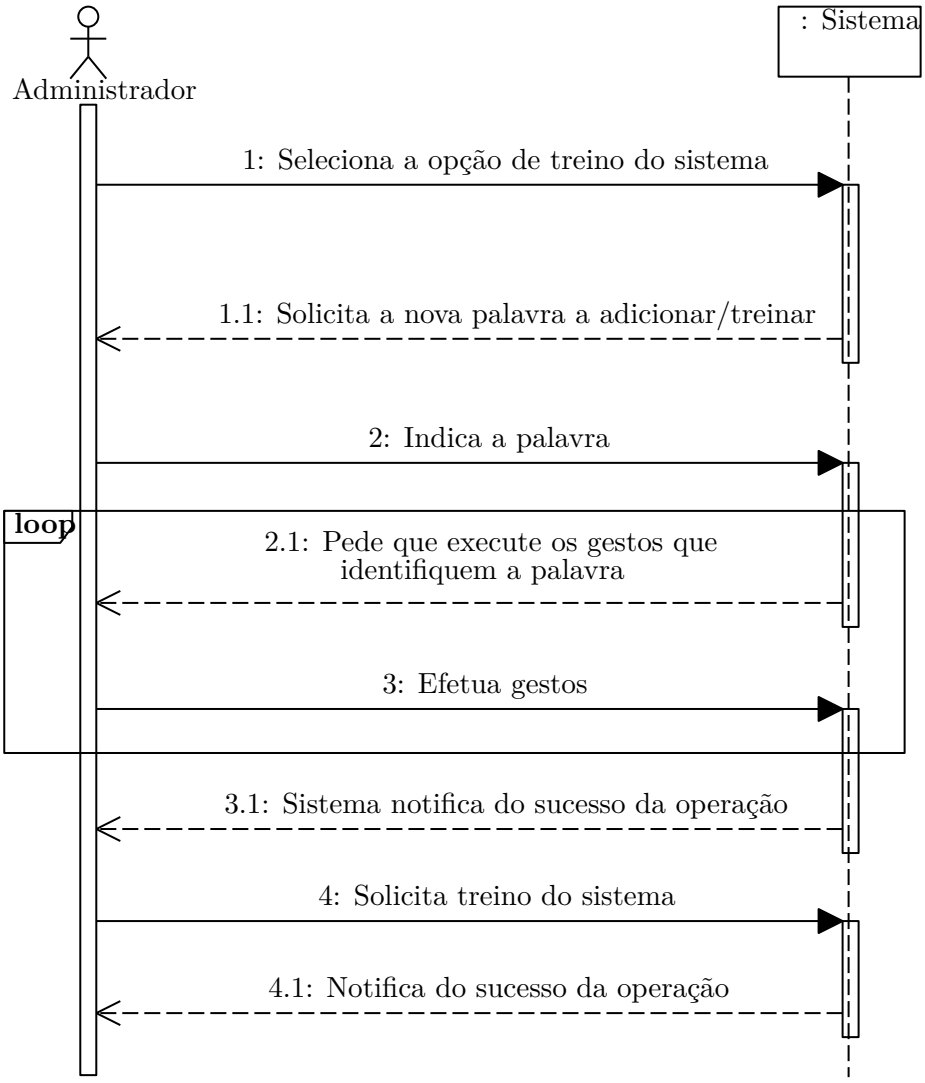


FIGURA 5.7: Vista de Processos do caso de uso 04 em UML

A figura 5.7 representa como o administrador pode treinar o sistema, para que este seja capaz de reconhecer novos gestos. A deteção de gestos pela parte do sistema constitui uma método de ML supervisionado, pelo que é necessário treinar o sistema previamente para que este consiga identificar os gestos feitos em LGP. Assim, o administrador indica a nova palavra a adicionar e realiza o gesto correspondente em LGP um certo número de vezes. O sistema notifica do sucesso da operação assim que o processo estiver terminado. Para terminar, o administrador solicita o treino do sistema a contar com o novo gesto adicionado. O sistema notifica do sucesso da operação aquando do término desta.

**UC05: Preencher Inquéritos de Usabilidade e Satisfação**

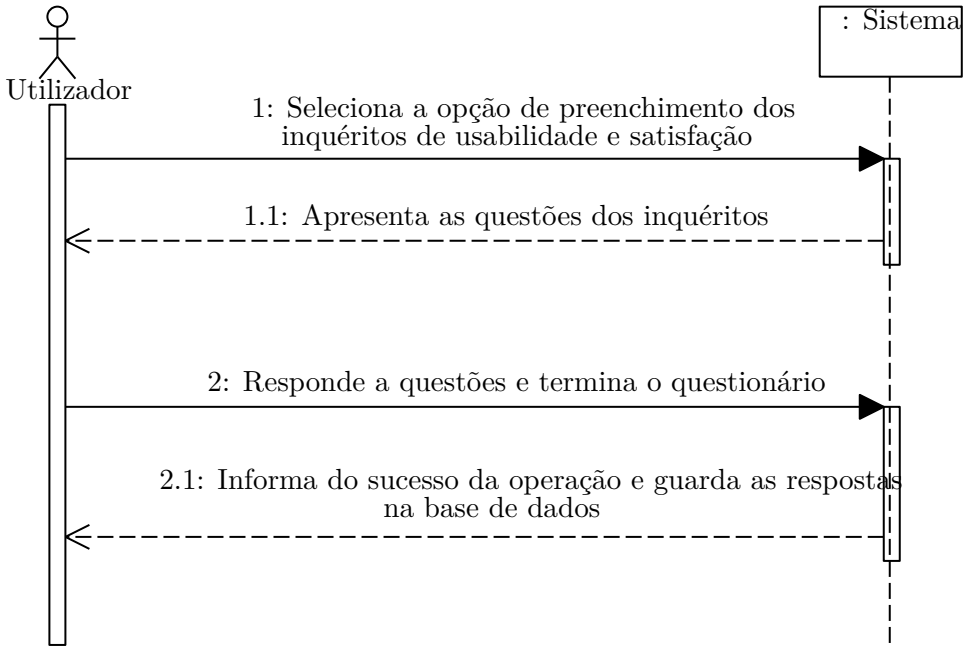


FIGURA 5.8: Vista de Processos do caso de uso 05 em UML

O último caso de uso, cujo fluxo se encontra espelhado na figura 5.8, encontra-se relacionado com o preenchimento de dois inquéritos, um ligado à usabilidade do sistema e o outro à satisfação que o utilizador sente em relação ao mesmo. Deste modo, o utilizador começa por selecionar a opção para preencher estes inquéritos e o sistema apresenta as questões correspondentes. Depois de todas as perguntas terem sido respondidas, o sistema notifica o utilizador do sucesso da operação.

## 5.3 Nível 2

Nesta secção apresenta-se ao leitor os principais componentes integrantes do sistema, que até agora era apenas representado como uma "caixa negra".

### 5.3.1 Vista Lógica

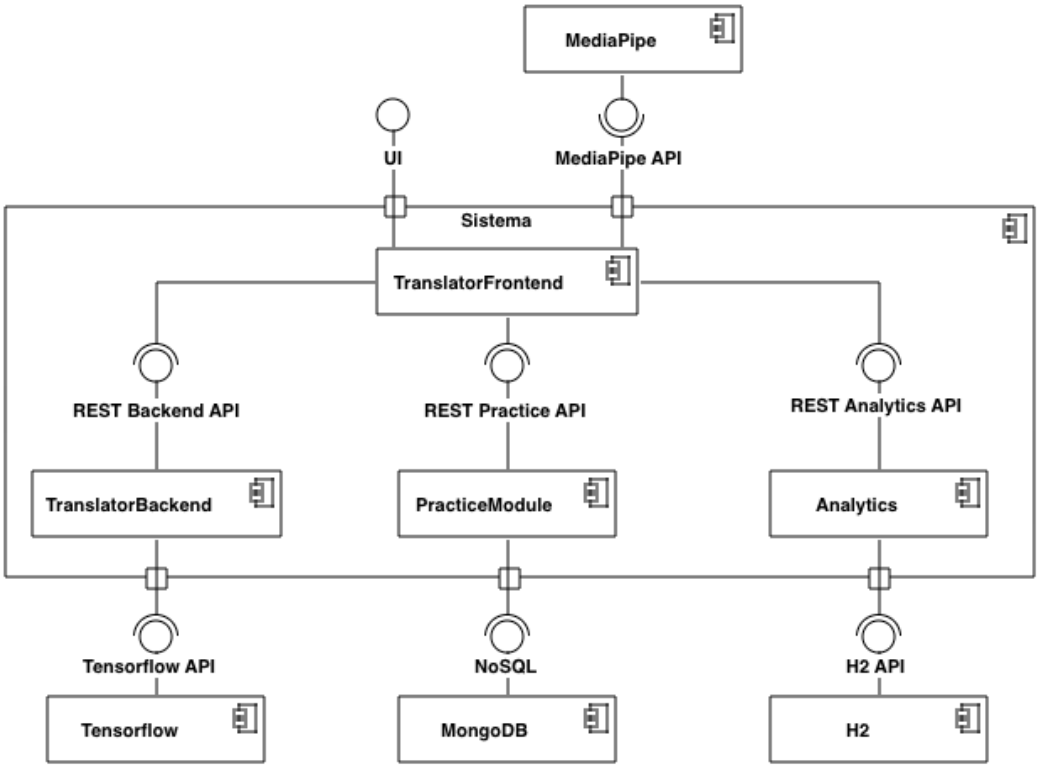


FIGURA 5.9: Vista lógica de nível 2

Analisando a figura 5.9 é possível identificar quatro componentes distintos que integram o sistema: "TranslatorFrontend"; "TranslatorBackend"; "PracticeModule" e "Analytics".

**TranslatorFrontend** responsável por construir a interface gráfica aos utilizadores e por comunicar com eles. Esta API consome a interface MediaPipe API, fornecida pela *framework* MediaPipe, que permite identificar mãos e caras de pessoas. Para além disto, o componente TranslatorFrontend consome também as várias APIs disponibilizadas pelos restantes módulos do sistema, de modo a dispor aos utilizadores todas as funcionalidades que estas interfaces oferecem;

**TranslatorBackend** responsável por realizar as traduções propriamente ditos, estando a ele associados os casos de uso "Iniciar Tradução", "Realizar Tradução" e "Terminar Tradução";

**PracticeModule** responsabiliza-se pela execução do caso de uso "Treinar sistema". Este componente é utilizado essencialmente para fortalecer o processo de tradução. Consome ainda a Mongo API de modo a interagir com a MongoDB, onde neste momento vai armazenar dados dos utilizadores que podem treinar o sistema;

**Analytics** responsável pelos casos de uso "Preencher inquérito de usabilidade" e "Preencher inquérito de satisfação". Esta aplicação trata as respostas a estes inquéritos e guarda-as na base de dados através da H2 API.

### 5.3.2 Vista de Processos

Para a demonstração desta vista, como já foi explicado , utiliza-se o caso de uso "Realizar Tradução".

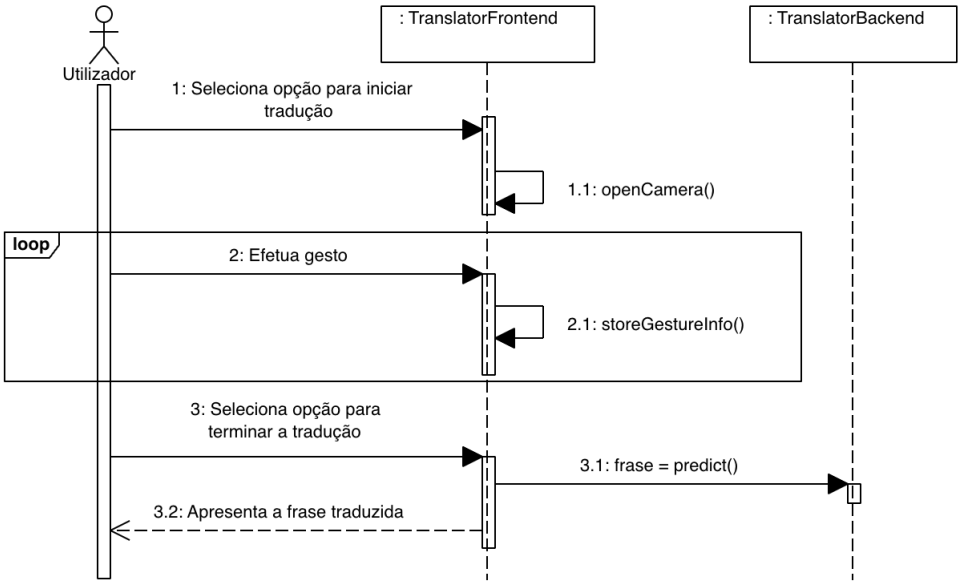


FIGURA 5.10: Vista de processos de nível 2 conforme os modelos 4+1 e C4 em UML

Analisando a figura 5.10, da esquerda para a direita, verifica-se que o caso de uso começa com o utilizador a selecionar a opção para iniciar a tradução. A aplicação TranslatorFrontend abre a câmara do dispositivo e armazena a informação relativa a todos os gestos que o utilizador efetue. Assim que este requisitar o término da tradução, a aplicação TranslatorFrontend envia um pedido REST para Translator-Backend, para que todos os gestos sejam traduzidos, construindo a frase final que é retornada e apresentada ao utilizador.

### 5.3.3 Vista de Implementação

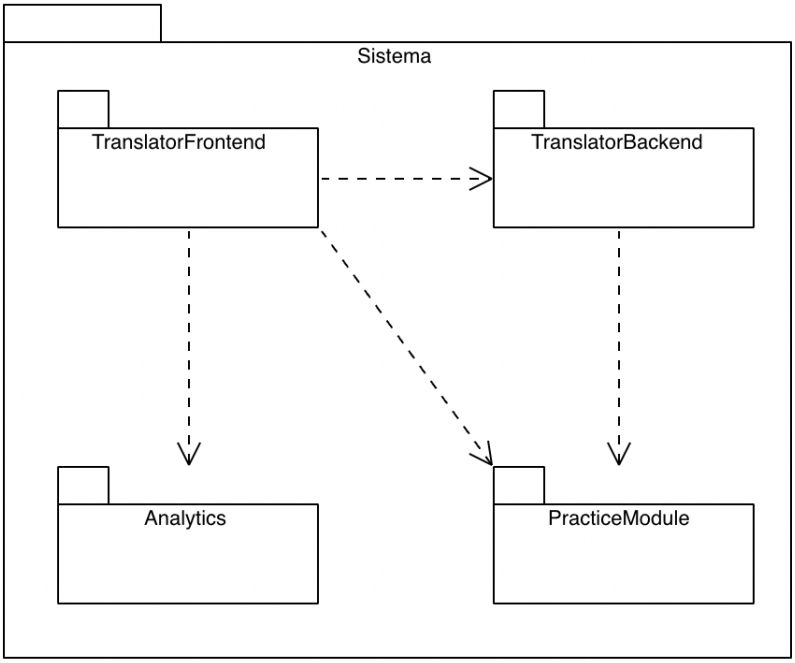


FIGURA 5.11: Vista de implementação de nível 2 conforme os modelos 4+1 e C4 em UML

A partir da figura 5.11, verifica-se a organização estática do software desenhado. Analisando este diagrama, conclui-se que a API TranslatorFrontend depende das APIs Analytics, PracticeModule e TranslatorBackend. Esta última, por sua vez, depende da PracticeModule, uma vez que os resultados das traduções vão depender do treino do sistema. Tanto Analytics como PracticeModule não apresentam nenhuma dependência pelas restantes aplicações apresentadas.

### 5.3.4 Vista Física

Para este caso em específico, não faz sentido apresentar a vista física do sistema ao leitor, uma vez que todas as aplicações vão ser utilizadas em modo local pelo facto de este não ser o foco principal do projeto. Assim, caso seja possível demonstrar que esta é uma solução viável, o próximo passo será fazer o *deploy* dos vários módulos.

## 5.4 Nível 3 - TranslatorBackend

Neste secção é abordado o nível 3 do *design* arquitetural construído para o *container* TranslatorBackend.

### 5.4.1 Vista Lógica

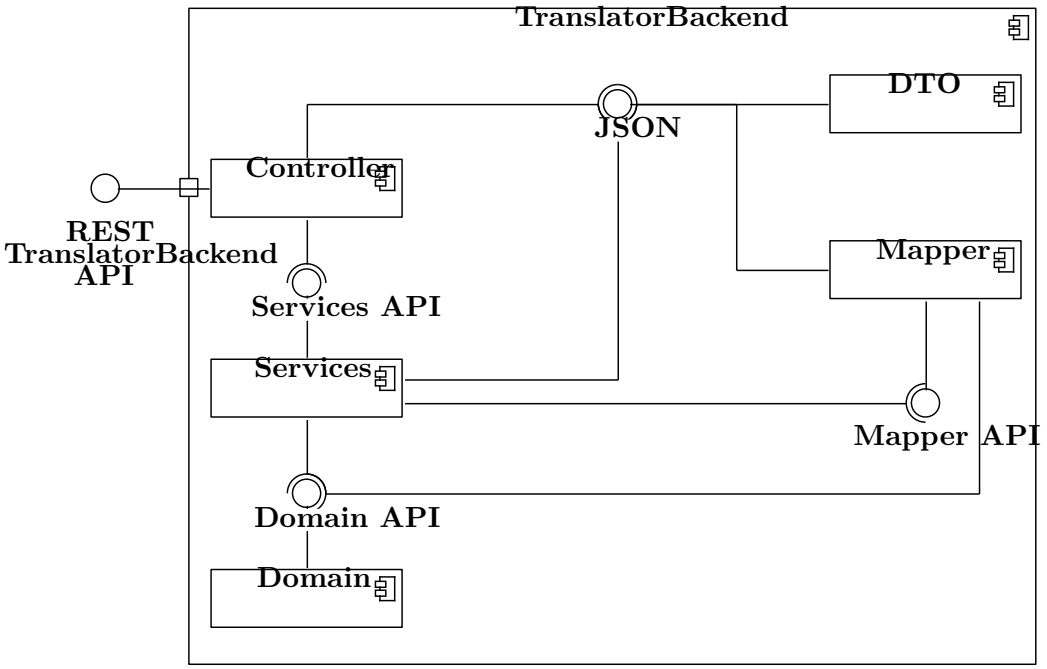


FIGURA 5.12: Vista lógica de nível 3 do *container* TranslatorBackend conforme os modelos 4+1 e C4 em UML

A partir da figura 5.12 é possível visualizar a composição do componente TranslatorBackend, que apresenta uma arquitetura por camadas.

**Controller** tem a responsabilidade de receber e controlar o fluxo dos pedidos feitos à aplicação. Consome a interface do componente DTO, quando recebe ou envia pedidos REST, e do componente Service;

**Service** camada que contém toda a lógica de tradução. É aqui que acontece a maioria do processamento. Estando os gestos traduzidos, esta camada vai construindo as frases e respetivas frases;

**Domain** camada que possui a lógica inerente aos objetos de domínio;

**DTO** camada utilizada para transferir informação entre componentes. Fornece uma interface que é consumida pelos componentes Controller e Mapper;

**Mapper** responsável por mapear os objetos. Deste modo, esta responsabilidade fica isolada num só componente.

## 5.5 Nível 3 - PracticeModule

Neste secção é abordado o nível 3 do *design* arquitetural construído para o módulo "PracticeModule".

### 5.5.1 Vista Lógica

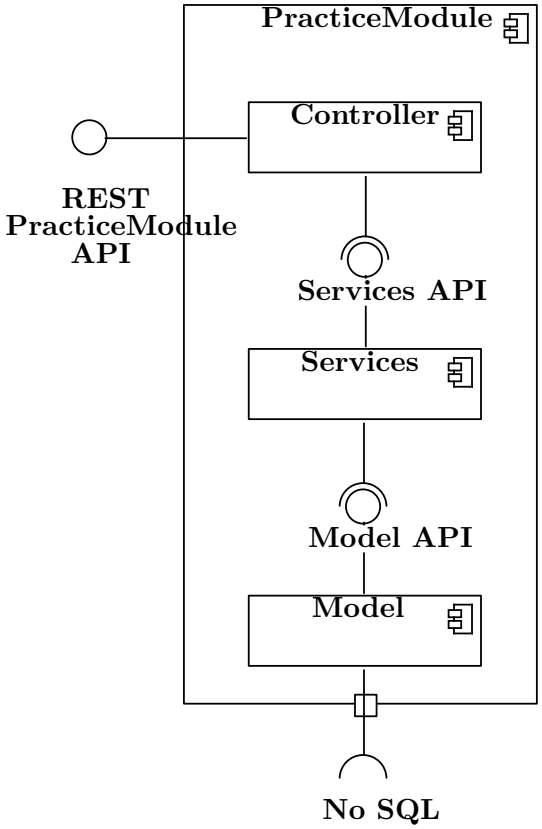


FIGURA 5.13: Vista lógica de nível 3 do *container* PracticeModule conforme os modelos 4+1 e C4 em UML

A figura 5.13 retrata a vista lógica do *container* Practice Module. A partir dela é possível entender que este também foi desenhado com uma arquitetura por camadas. No entanto, mais simples do que a apresentada na figura 5.12, uma vez que possui apenas três camadas.

**Controller** tem o intuito de receber e enviar pedidos REST, sendo que consome a interface proveniente de Services. Para além disto fornece a API REST PracticeModule API para o exterior;

**Services** trata do processamento dos dados, sendo aqui onde é feito realmente o treino do sistema de tradução;

**Model** contém informação do domínio e que interage com a base de dados (neste caso MongoDB).

## 5.6 Nível 3 - Analytics

Neste secção é abordado o nível 3 do *design* arquitetural construído para a aplicação Analytics.

### 5.6.1 Vista Lógica

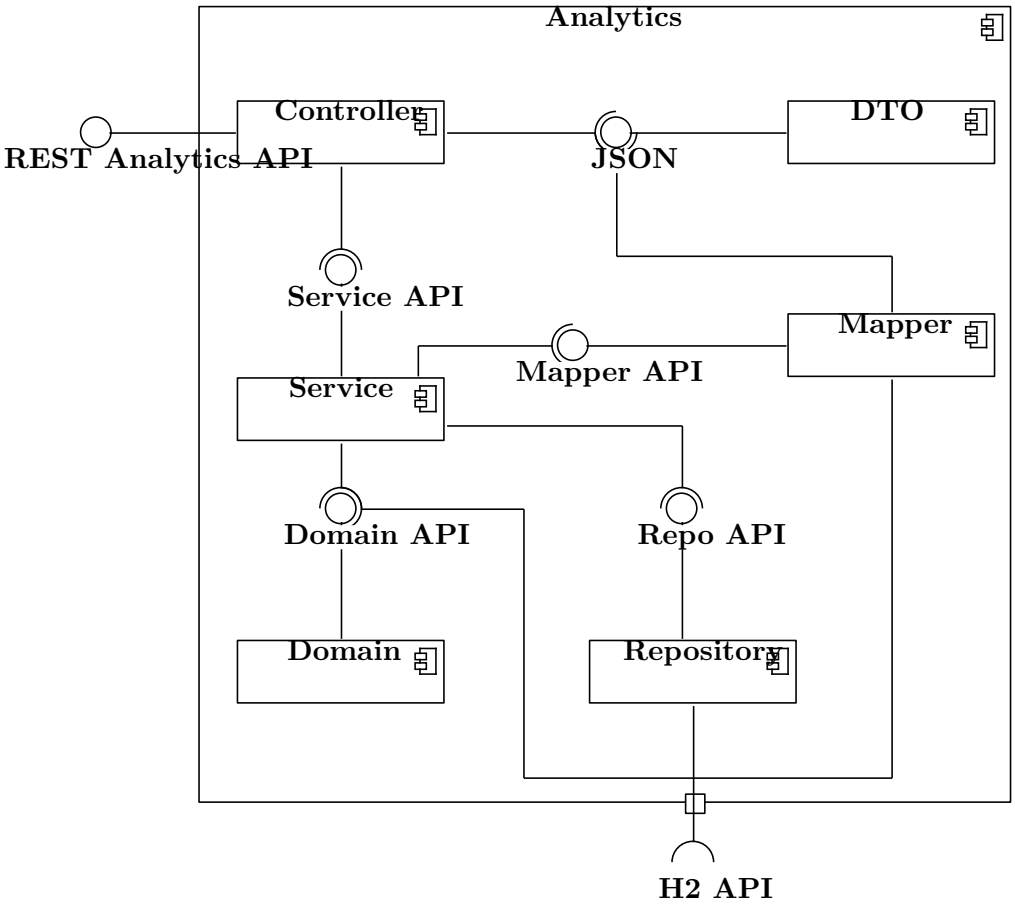


FIGURA 5.14: Vista lógica de nível 3 do *container* Analytics conforme os modelos 4+1 e C4 em UML

Na figura 5.14, observam-se os componentes que constituem a API Analytics, notando-se de imediato que apresenta uma arquitetura por camadas.

**Controller** componente que tem a responsabilidade de receber e controlar o fluxo dos pedidos feitos à aplicação. Para além disto, ainda fornece a interface REST Analytics API para o exterior e consome a interface do componente DTO, quando recebe ou envia pedidos REST, e do componente Service, quando utiliza esta camada para redirecionar os pedidos com a informação necessária;

**Service** componente que recebe a informação processada pelo Controller e trata a informação dos inquéritos. O componente Service consome três interfaces: a Domain API, quando cria objetos de domínio, a Repo API, quando precisa de comunicar com a base de dados e a Mapper API, quando é necessário mapear objetos, por exemplo um DTO para um objeto de domínio;

**DTO** é utilizado para transferir informação entre componentes e fornece uma interface que é consumida pelos componentes Controller e Mapper;

**Domain** componente que representa os objetos de domínio. A interface que este disponibiliza é consumida pelos componentes Service e Mapper. Neste caso o domínio prende-se com o inquérito;

**Mapper** responsável por mapear objetos, para que esta responsabilidade fique isolada num só componente;

**Repository** utilizado para interagir com a base de dados através da interface "H2 API", a qual este componente consome. Para além desta interface, a camada Repository ainda consome a Mapper API, para mapear objetos de persistência em objetos de domínio, sendo que fornece ainda interface Repo API, para isolar a responsabilidade de comunicar com a base de dados.

## 5.7 Nível 4

Para o nível quatro, visto que este é o nível que está mais próximo da implementação, apresentam-se apenas os diagramas de sequência dos casos de uso que sejam mais relevantes para a compreensão do projeto a desenvolver.

### 5.7.1 UC01 - Efetuar o *Login*

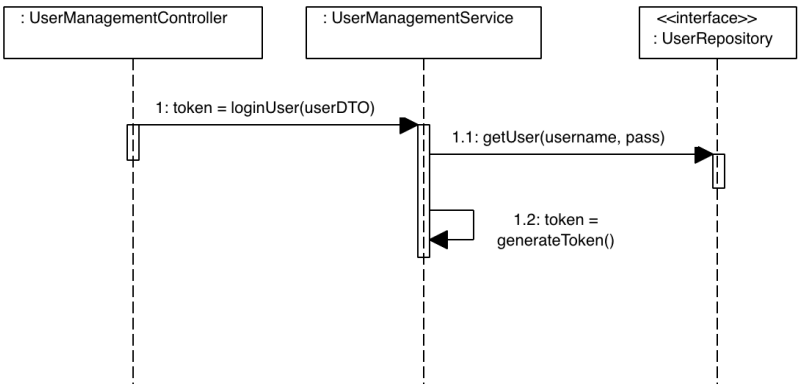


FIGURA 5.15: Diagrama de sequência do caso de uso 01 conforme os modelos 4+1 e C4 em UML

O diagrama ilustrado na figura5.15 representa o fluxo do caso de uso 01. É um caso de uso simples, a classe UserManagementController recebe um pedido REST com a informação do utilizador que está a tentar autenticar-se no sistema. O *controller* invoca o serviço UserManagementService que por sua vez comunica com a camada de persistência para descobrir se os dados presentes no objeto UserDTO retratam dados válidos de um administrador. Em caso positivo, o serviço gera um *token* de autorização que é retornado para o *frontend*.

No diagrama presente na figura 5.15, estão presentes os seguintes padrões:

**Controller** utilizado para controlar o fluxo de cada caso de uso;

**Information Expert** para que apenas cada classe contenha a sua própria informação;

**DTO** utilizado para transferir dados (por exemplo dos serviços para o *controller*);

### 5.7.2 UC02 - Sair do Sistema

O caso de uso "UC02 - Sair do Sistema" é o mais simples de todos, envolvendo apenas o componente TranslatorFrontend. Devido à sua simplicidade, um diagrama de sequência não acrescenta valor ao leitor no entendimento do projeto no seu global, uma vez que o SSD ilustrado na figura 5.5 é suficiente para retratar o funcionamento deste caso de uso, pelo que não é apresentado.

### 5.7.3 UC03 - Realizar Tradução

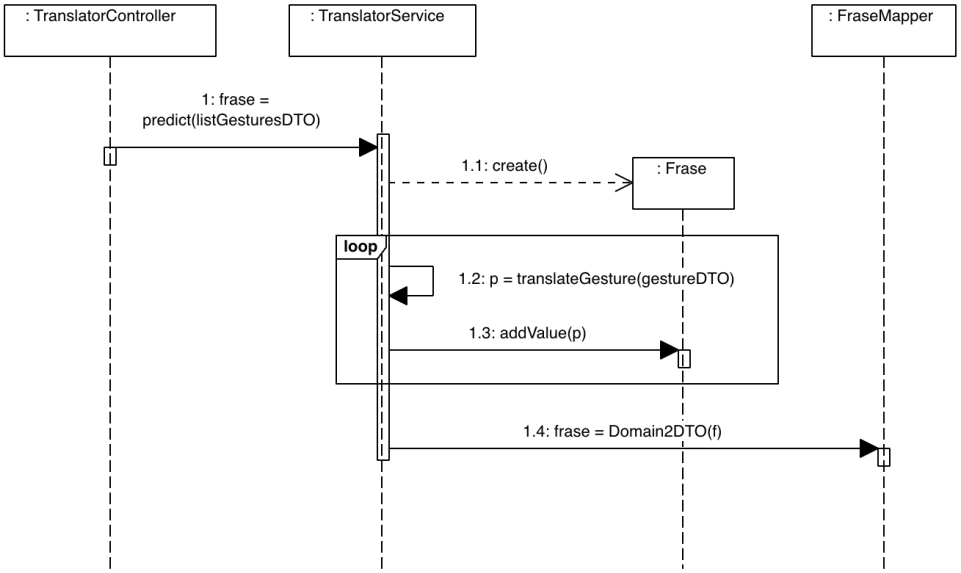


Figura 5.16: Diagrama de sequência do caso de uso 04 conforme os modelos 4+1 e C4 em UML

A figura 5.16 ilustra o diagrama de sequência representativo do caso de uso 3. Analisando este diagrama, o fluxo inicia-se com o *controller* TranslatorController a receber um pedido REST com uma lista de dados dos gestos realizados. Ao receber toda a informação em coleção permite que o sistema consiga modular palavras com continuidade, permitindo uma maior coerência nas construção frásica.

Assim, o *controller* encaminha esta informação para o serviço TranslatorServie que para cada gesto vai realizar a tradução e adicionar o resultado desta operação à frase a devolver no final. Por último, a frase é convertida para DTO com auxílio do Mapper e é retornada para o TranslatorController que remete a informação para o *frontend*.

No diagrama presente na figura 5.16, estão presentes os seguintes padrões:

**Controller** utilizado para controlar o fluxo de cada caso de uso;

**Information Expert** para que apenas cada classe contenha a sua própria informação;

**DTO** utilizado para transferir dados (por exemplo dos serviços para o *controller*);

**Mapper** utilizado para converter dados (por exemplo DTO para objetos de domínio).

### 5.7.4 UC04 - Treinar Sistema

A figura 5.17 representa o diagrama de sequência alusivo ao caso de uso em estudo.

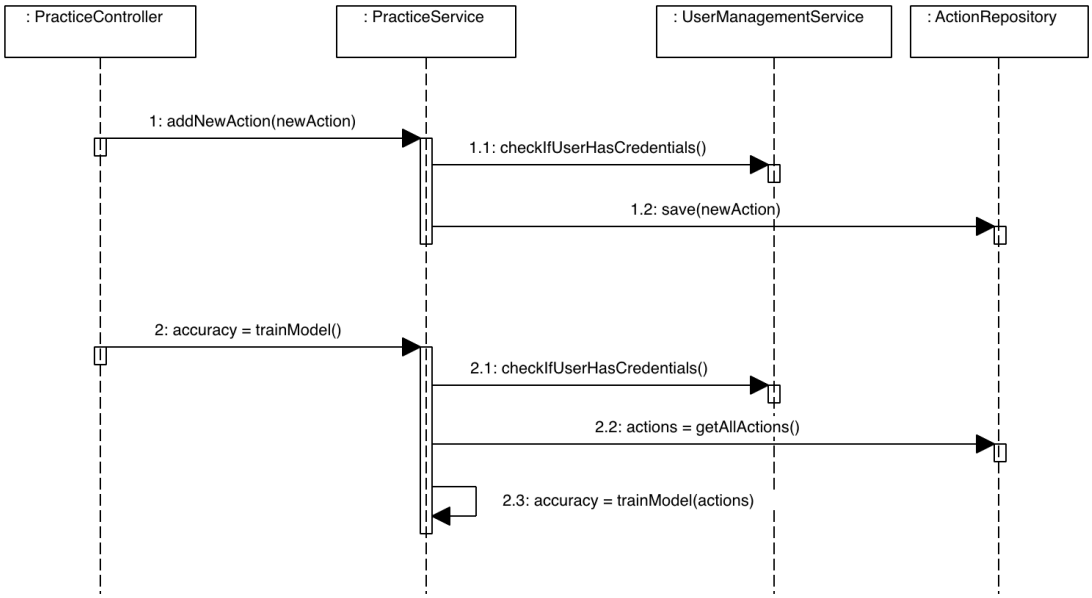


FIGURA 5.17: Diagrama de sequência do caso de uso 04 conforme os modelos 4+1 e C4 em UML

Observando o diagrama presente na figura 5.17, nota-se que este possui duas partes distintas. A primeira parte prende-se com a adição de novos gestos e respetivas traduções, enquanto que a segunda representa o treino propriamento dito do sistema.

Começando a análise pela primeira parte, a classe PracticeController recebe um pedido para adicionar um novo gesto ao reportório já existente. De seguida, invoca o serviço PracticeService que contém a lógica essencial para esta operação, enviando por parâmetro os dados do novo gesto e a respetiva tradução. O serviço, começa por verificar se o utilizador possui as credenciais necessárias para realizar esta operação, como se verifica na mensagem 1.1. Uma vez que este caso de uso afeta consideravelmente a precisão das traduções, e consequentemente a qualidade do sistema, é necessário restringir os indivíduos que podem treinar o sistema. Superada esta etapa, o serviço comunica com o repositório ActionRepository para adicionar uma nova entrada na tabela que armazena todas as traduções disponíveis. De seguida, processa os dados relativos aos gestos e atualiza o *dataset*.

A segunda parte deste caso de uso começa quando o administrador decide processar o modelo de IA. O *controller* recebe o pedido e invoca novamente o serviço PracticeService. Este serviço verifica novamente se o utilizador possui o cargo de administrador e em caso positivo avança para o treino. O primeiro passo é obter todas as traduções da base de dados, como se observa na mensagem 2.2. Este passo é necessário porque o sistema é treinado de raiz, ou seja, se fosse treinado apenas com o novo gesto os antigos seriam esquecidos. De seguida procede-se ao treino do sistema e consequente cálculo da precisão deste, que para terminar é retornada para o *frontend*.

No diagrama presente na figura 5.17, estão presentes os seguintes padrões:

**Controller** utilizado para controlar o fluxo de cada caso de uso;

**Information Expert** para que apenas cada classe contenha a sua própria informação;

**DTO** utilizado para transferir dados (por exemplo dos serviços para o *controller*);

**Mapper** utilizado para converter dados (por exemplo DTO para objetos de domínio).

### 5.7.5 UC05 - Preencher Inquérito de Usabilidade e de Satisfação

Na figura 5.18 é possível verificar o fluxo do caso de uso em análise.

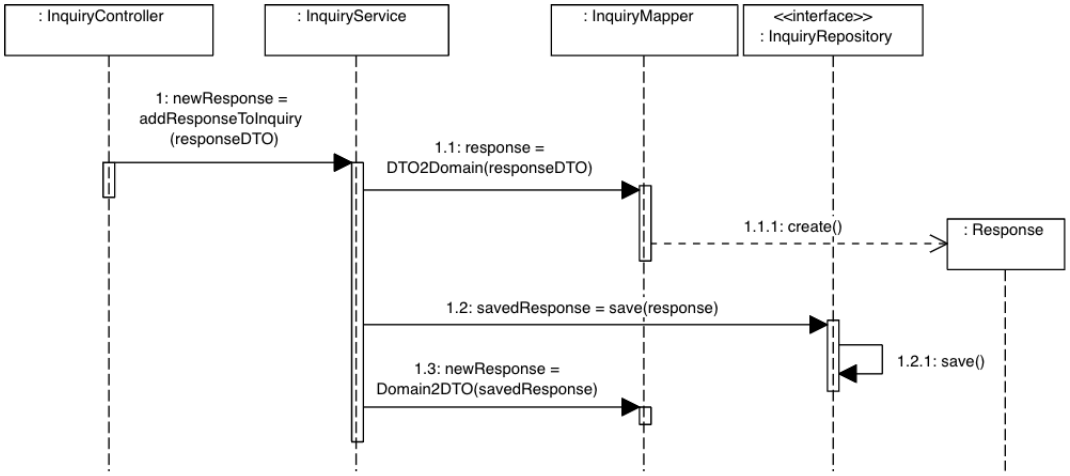


Figura 5.18: Diagrama de sequência do caso de uso 05 conforme os modelos 4+1 e C4 em UML

Analisando a figura 5.18, o caso de uso inicia-se na classe "InquiryController" a receber um pedido para armazenar as respostas aos inquéritos de usabilidade e de satisfação. O *controller* invoca o serviço InquiryService, que por sua vez recebe as informações da resposta no objeto "responseDTO".

Este objeto contém as várias perguntas dos questionários e as respostas atribuídas a cada uma delas.

O serviço utiliza o *mapper* para mapear o DTO num objeto de domínio, que é posteriormente armazenado na base de dados com o auxílio da interface "InquiryRepository". Esta interface pode ser implementada por várias classes na camada de persistência, de modo a que seja possível utilizar vários tipo de bases de dados. Por enquanto, apenas será utilizada uma implementação para H2.

Para terminar, o serviço utiliza novamente o *mapper* para converter o objeto de domínio retornado pela base de dados num DTO, para o enviar como resposta ao pedido original.

No diagrama presente na figura 5.18, estão presentes os seguintes padrões:

**Controller** utilizado para controlar o fluxo de cada caso de uso;

**Information Expert** para que apenas cada classe contenha a sua própria informação;

**DTO** utilizado para transferir dados (por exemplo dos serviços para o *controller*);

**Mapper** utilizado para converter dados (por exemplo DTO para objetos de domínio);

**Repository** utilizado para abstrair a camada de persistência e permitir o uso de vários

## 5.8 Sumário

O objetivo deste capítulo é apresentar ao utilizador o *design* arquitetural construído para o presente projeto.

A primeira etapa foi selecionar entre duas alternativas arquiteturais qual a mais adequada: Segregação de Responsabilidade ou Centralização de API. Avaliando as vantagens e desvantagens de cada uma, foi decidido optar pela primeira opção, devido à sua mais fácil gestão e capacidade de manutenção futura.

Posto isto, apresentam-se os vários diagramas de *design* seguindo os modelos 4+1 e C4, em UML. No nível dois entende-se que o sistema é composto por quatro componentes: TranslatorFrontend; TranslatorBackend; PracticeModule; Analytics.

O TranslatorFrontend tem como objetivo comunicar com o utilizador e fornecer a interface gráfica. Tal como o nome indica, constitui o *frontend* do sistema.

O TranslatorBackend é responsável por toda a lógica de tradução, efetuando todo o processamento necessário para as mesmas.

O componente PracticeModule responsabiliza-se pelo treino do sistema, sendo apenas utilizado por administradores. É essencial para fortalecer o processo de tradução.

Por último, Analytics trata de toda a gestão associada aos inquéritos.

# Capítulo 6

# Implementação

O presente capítulo expõe ao leitor uma descrição detalhada dos elementos relevantes para a implementação das várias *user stories*, previamente detalhadas na secção 4.1, sendo que algumas podem receber um foco maior do que outras consoante a sua relevância. A última secção deste capítulo ainda aborda a construção de uma PWA, o requisito não funcional considerado de maior valor.

## 6.1 User Story 01

Apesar da *user story 01* (como administrador quero autenticar-me no sistema) não ser considerada a mais relevante e essencial para o projeto, esta acabou por ser o ponto de partida. Esta decisão foi tomada, essencialmente, pela necessidade de delinear toda a interface gráfica que, conforme o requisito não funcional RNF-01, exposto na secção 4.2, indica que esta deve ser intuitiva e simples. Assim, a tarefa de traçar e definir toda a interface gráfica torna-se mais fácil após a construção da página inicial que, neste, caso é a página da autenticação. Assim, no seguimento desta lógica, a figura 6.1 demonstra a página de autenticação da aplicação.

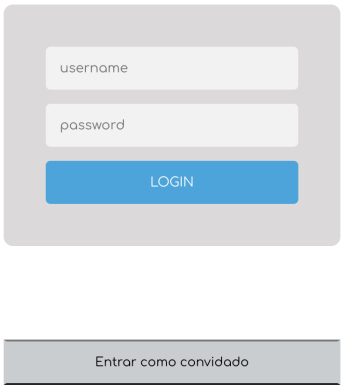


Figura 6.1: Página de autenticação

Na figura 6.1 é possível observar que esta página é bastante minimalista, contendo apenas um pequeno formulário onde se requisita o nome de utilizador e a *password* para efetuar a autenticação. Existe, também, um botão que permite aos utilizadores, que não são administradores, entrarem na aplicação e usufruírem das funcionalidades

a que têm direito (este último será abordado numa *user story* mais à frente). Assim que o administrador introduza os dados necessários e carregue no botão "Login", o componente "TranslatorFrontend" efetua um pedido REST para o componente "PracticeModule". Este pedido é recebido pelo *controller* que extrai a informação necessária e envia-a para o serviço "UserManagementService", que possui a lógica essencial para o desenvolvimento deste requisito e se encontra representada no excerto de código 6.1.

```
1  def login(user, password):
2      all_db_users = mongo.db.Users.find()
3      for db_user in all_db_users:
4          if user == db_user['name'] and password == db_user['password']:
5              payload = {
6                  'user': user,
7                  'exp': datetime.now() + timedelta(days=1)
8              }
9              encoded_token = jwt.encode(
10                 payload,
11                 "secret",
12                 algorithm='HS256'
13             )
14             return {'token': encoded_token, 'user': {'id': str(db_user['_id']),'name':
       db_user['name'], 'role': db_user['role']}}
15         else:
16             return {'token': 0}
```

Listing 6.1: Excerto de código do método "login"

Analisando o excerto de código 6.1, nota-se que o primeiro passo é recolher os vários administradores existentes na base de dados, como se verifica na linha 2. Estes dados são recolhidos com o auxílio da biblioteca "PyMongo", utilizada para intermediar a comunicação com MongoDB. De seguida, itera-se pelos vários utilizadores até se encontrar algum *match*. Caso exista uma combinação, gera-se um *token* de autenticação, como se pode observar na linha 9. Este *token* tem duração máxima de 1 dia e necessita de ser renovado após esse prazo. Terminado este processo, o *token* é retornado para o *controller* que se encarrega de o enviar para o componente "TranslatorFrontend". Este *token* vai ser depois utilizado em todos os pedidos com destino ao componente "PracticeComponente", para demonstrar que o emissor do pedido é um administrador autenticado que possui autoridade para efetuar as operações desejadas. No caso de não existir nenhum *match*, é retornado o valor 0, indicando que houve um erro.

É importante notar que as *passwords* neste momento não se encontram encriptadas, uma vez que a segurança não é o foco principal deste projeto. O processo de autenticação serve o propósito de demonstrar como as funcionalidades dos administradores e dos utilizadores comuns se encontram divididas e devidamente protegidas.

## 6.2 User Story 02

Este requisito (US02 - como administrador quero sair do sistema) é provavelmente o mais simples e não requer demasiadas explicações. Um administrador autenticado, que queira sair do sistema, apenas necessita de carregar no botão de *logout*, representado na figura 6.2, presente em todas as páginas da aplicação.

FIGURA 6.2: Botão de *logout*

Ao pressionar este botão, o sistema procede ao exercício de limpar o armazenamento local do cliente, de modo a que não existam vestígios do *token* de autenticação a ser utilizado.

## 6.3 User Story 03

Este requisito (US03 - como utilizador quero que o sistema realize a tradução de LGP para língua portuguesa escrita) é provavelmente o mais crucial para o projeto, visto que se foca no objetivo principal deste, a tradução de LGP para língua portuguesa. Além disto, esta *user story* é tão relevante como complexa, pelo que a sua análise será um pouco mais densa que as anteriores.

Esta funcionalidade tem início no componente "TranslatorFrontend" e não é restrita a administradores, pelo que não é necessária autenticação, sendo possível entrar no sistema como convidado, ao carregar no botão no fundo da figura 6.1. Assim, quer se passe pelo processo de autenticação ou não, a página de tradução, presente na figura 6.3, é sempre apresentada ao utilizador.

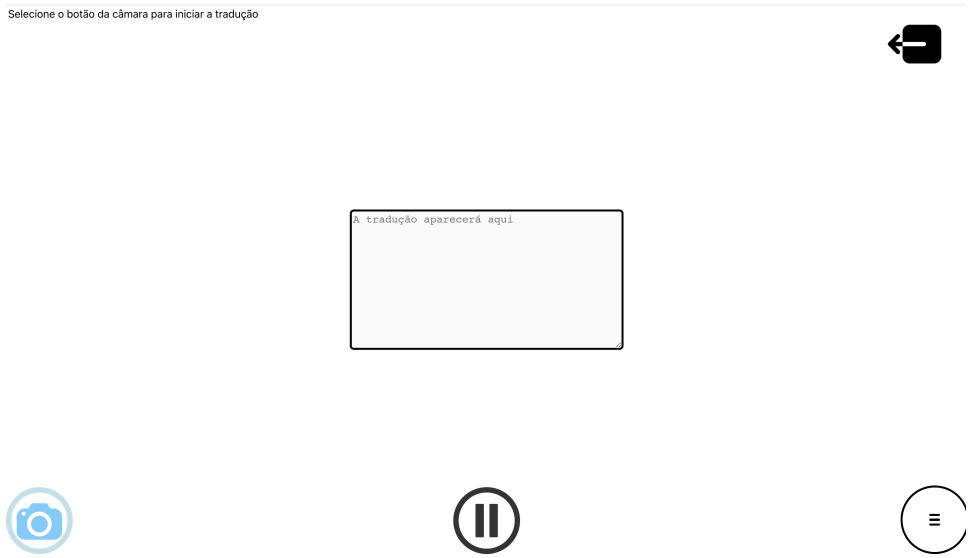


FIGURA 6.3: Página de tradução

Observando a figura 6.3 é possível notar que, apesar de simples, ainda existem alguns elementos nesta página. No canto superior direito posiciona-se o botão de *logout*, já abordado na secção 6.2. No centro da página permanece uma caixa de texto onde é apresentado o resultado das traduções efetuadas. Esta não permite receber texto, sendo apenas utilizada para o exibir. O rodapé da página é composto por três botões distintos. Começando esta análise da esquerda para a direita, o primeiro é

utilizado para ligar a câmara do dispositivo e captar os gestos realizados. O botão que se situa na parte central desliga a câmara e, de seguida, efetua um pedido REST para o componente "TranslatorBackend", de forma a traduzir os gestos efetuados anteriormente. Por último, o botão mais à direita funciona como um pequeno menu que, conforme as permissões do utilizador, disponibiliza as seguintes opções distintas:

- Treinar o Sistema - Esta opção apenas se encontra disponível para administradores e, quando selecionada, apresenta a página dedicada ao treino do sistema;
- Responder a Inquéritos - Esta opção encontra-se disponível para qualquer utilizador e, quando selecionada, apresenta a página dedicada aos inquéritos.

Graficamente, as opções deste menu encontram-se espelhadas na figura 6.4. A opção à esquerda permite treinar o sistema, enquanto que a superior possibilita o acesso aos inquéritos.

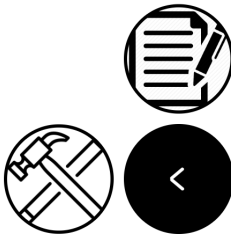

FIGURA 6.4: Opções disponíveis no menu

Mergulhando um pouco no aspeto técnico deste requisito, ao pressionar o botão que inicia a câmara, a função "useEffect", nativa do React e demonstrada no excerto de código 6.2, é invocada, sendo esta responsável pela inicialização da *framework* "Mediapipe Holistic".

```
1  useEffect(() => {
2          const config = {
3              locateFile: (file) => {
4                  return `https://cdn.jsdelivr.net/npm/@mediapipe/holistic@` +
5                      `${mpHolistic.VERSION}/${file}`;
6              }
7          };
8          const holistic = new mpHolistic.Holistic(config);
9          holistic.setOptions({
10             minDetectionConfidence: 0.5,
11             minTrackingConfidence: 0.5,
12             selfieMode: true,
13             enableFaceGeometry: true,
14             modelComplexity: 2,
15             smoothLandmarks: true,
16             enableSegmentation: false,
17             smoothSegmentation: true,
18         });
19         holistic.onResults(onResults);
20         if (typeof webcamRef.current !== "undefined" &&
21             webcamRef.current !== null &&
22             webcamRef.current.video != null) {
23
24             camera = new cam.Camera(webcamRef.current.video, {
25                 onFrame: async () => {
26                     await holistic.send({ image: webcamRef.current.video });
27                 },
28                 width: 640,
29                 height: 480
30             });
31             camera.start();
32         }
33     }, [openCamera]);
```

Listing 6.2: Excerto de código da função "useEffect"

Detalhando o excerto de código 6.2, a maioria das linhas de código nele presentes servem para instanciar a *framework* e definir o aspeto estético de alguns componentes, como se verifica, por exemplo, nas linhas 24 a 29. No entanto, a linha 19 é a mais pertinente destas. Esta linha de código regista a função que deve ser executada sempre que houverem resultados retornados pela parte do Mediapipe, sendo que, neste caso, a função a executar é a "onResults". Uma vez que esta função é extensa, alguns excertos dela foram divididos nos excertos de código 6.3 e 6.4.

```
1  async function onResults(results) {
2
3          // Remove landmarks we don't want to draw.
4          removeLandmarks(results);
5          const videoWidth = webcamRef.current.video.videoWidth;
6          const videoHeight = webcamRef.current.video.videoHeight;
7
8          // Set canvas width and height
9          canvasRef.current.width = videoWidth;
10         canvasRef.current.height = videoHeight;
11
12         // Draw the overlays.
13         const canvasElement = canvasRef.current;
14         ...
15
16         // Connect elbows to hands.
17         ...
18
19         try {
20             // Pose...
21             ...
22
23             // Hands...
24             drawingUtils.drawConnectors(
25                 canvasCtx, results.rightHandLandmarks, mpHolistic.HAND_CONNECTIONS,
26                 { color: 'white' });
27             drawingUtils.drawLandmarks(canvasCtx, results.rightHandLandmarks, {
28                 color: 'white',
29                 fillColor: 'rgb(0,217,231)',
30                 lineWidth: 2
31             });
32
33             drawingUtils.drawConnectors(
34                 canvasCtx, results.leftHandLandmarks, mpHolistic.HAND_CONNECTIONS,
35                 { color: 'white' });
36             drawingUtils.drawLandmarks(canvasCtx, results.leftHandLandmarks, {
37                 color: 'white',
38                 fillColor: 'rgb(255,138,0)',
39                 lineWidth: 2
40             });
41
42             // Face...
43             ...
44
45         } catch (ex) {
46             // do nothing
47         }
48
49
50     canvasCtx.restore();
51 }
```

Listing 6.3: Excerto de código do método "onResults"

O excerto de código 6.3 representa o início da função "onResults", sendo que as reticências ("...") são utilizadas para esconder código que não se considerou relevante apresentar ao leitor. Assim sendo, a primeira fase começa por invocar a função "removeLandmarks" na linha 4, utilizada para remover conectores e pontos desnecessários. Depois, definem-se as dimensões do objeto "canvasRef", de modo a apresentar o vídeo corretamente aos utilizadores. A partir da linha 18, dentro do bloco "try-catch", desenham-se os conectores das mãos (*Hands*), da cara (*Face*) e do corpo (*Pose*). O excerto de código 6.3 apenas demonstra como este processo é feito para as mãos, visto que os restantes, por serem semelhantes, não acrescentam valor para o leitor.

O excerto de código 6.4 contém a parte final da função "onResults".

```
1  async function onResults(results) {
2      ...
3      try {
4          ...
5          const filteredLeftHandData = results.rightHandLandmarks?.map(function (landmark) {
6              return {
7                  "x": landmark.x,
8                  "y": landmark.y,
9                  "z": landmark.z
10             };
11         });
12         const filteredRightHandData = ...
13         const filteredFaceData = ...
14
15         dataMatrix.push({
16             value: {
17                 "left_hand": filteredLeftHandData == null ? [] : filteredLeftHandData,
18                 "right_hand": filteredRightHandData == null ? [] : filteredRightHandData,
19                 "pose": results.poseLandmarks == null ? [] : results.poseLandmarks,
20                 "face": filteredFaceData == null ? [] : filteredFaceData
21             }
22         })
23
24         if (dataMatrix.length === 30) {
25             allWordsDataMatrix.push(dataMatrix);
26             dataMatrix = [];
27             await timer(2000);
28         }
29
30     } catch (ex) {
31         // do nothing
32     }
33     canvasCtx.restore();
34 }
```

Listing 6.4: Excerto final de código do método "onResults"

A linha 5 do excerto de código 6.4 marca a continuação da função em análise. O objetivo nesta fase é extrair os pontos (*keypoints*) das mãos, cara e corpo. Cada *keypoint* possui um conjunto de *landmarks*: "x", "y", "z" e "visibility". Esta última apenas possui valor para *keypoints* do corpo (*pose*), tendo sempre o valor *undefined* para os restantes. Por esta mesma razão, nas linhas 5, 12 e 13, realiza-se uma filtragem aos *keypoints* das mãos e cara, de forma a remover a *landmark* "visibility", que nunca contém valor, facilitando assim o trabalho de processamento do *backend*. A filtragem para a mão direita (linha 12) e para a cara (linha 13) encontra-se colapsada, uma vez que é realizada exatamente da mesma forma que a filtragem para a mão esquerda.

De seguida, com os vários *keypoints* recolhidos e filtrados, na linha 15, estes são adicionados à estrutura "dataMatrix". Na prática, este objeto possui, no presente momento, toda a informação de um *frame*[1] de um total de trinta, necessários para construir um vídeo de um gesto. O modelo de inteligência artificial foi construído para avaliar vídeos de trinta *frames*, visto que é um valor bom o suficiente para construir vídeos fluídos. Assim que esta estrutura possua o número desejado de *frames*, passa na condição da linha 24, significando que um vídeo relativo a um gesto está concluído.

Posto isto, os dados de todos os *frames* são adicionados ao objeto "allWordsMatrix", que armazena todos os vídeos realizados. Ou seja, em cada posição do *array*[2] "allWordsMatrix" está presente um vídeo relativo a um gesto. Na linha 26, o objeto "dataMatrix" é limpo, de forma a recolher mais um vídeo e, na linha 27, é feito um compasso de espera de dois segundos, onde a câmara fica congelada para que o utilizador se possa posicionar para executar o próximo gesto.

[1] Representa uma única imagem numa sequência de imagens, denominado de vídeo [59]
[2] Uma coleção de elementos semelhantes armazenados em locais de memória contíguos [60]

Do lado do componente "TranslatorFrontend" apenas falta realizar o pedido REST para o *backend* com a informação dos vários vídeos, sendo efetuado no momento em que a câmara é desligada pelo utilizador.

O pedido é recebido pelo componente "TranslatorBackend" no *controller* "TranslatorController" que extrai a informação necessária e a envia para o serviço "TranslatorService" invocando o método "predict", espelhado no excerto de código 6.5.

```
1  def predict(data_map):
2      actions = []
3      sequence = []
4      all_db_actions = mongo.db.Actions.find();
5      for db_action in all_db_actions:
6          actions.append(db_action['name'])
7      all_actions = np.array(actions)
8      final_sentence = '';
9      for dm in data_map:
10         sequence.clear()
11         for data in dm:
12             frame = data['value']
13             keypoints = extract_keypoints(frame['pose'], frame['face'], frame['right_hand'
       ], frame['left_hand'])
14             sequence.append(keypoints)
15             if len(sequence) == 30:
16                 model = build_model(all_actions)
17                 model.load_weights('action.h5')
18                 res = model.predict(np.expand_dims(sequence, axis=0))[0]
19                 final_sentence = final_sentence + ' ' + all_actions[np.argmax(res)];
20
21     return {'result': final_sentence.strip().replace('_', ' ')}
```

Listing 6.5: Excerto de código do método "predict"

Analisando o excerto de código 6.5, o primeiro passo é recolher da base de dados todas as ações existentes, como se verifica na linha 3, e para as quais o modelo de inteligência artificial se encontra treinado. Estas ações, à semelhança dos utilizadores, são também armazenadas com recurso a MongoDB.

Na linha 7 é instanciada a variável "final_sentence" que, no final do método, será retornada com a frase traduzida. Na linha 8 inicia-se um ciclo que itera pelos valores da variável "data_map", proveniente do *controller* e que contém a informação dos vários vídeos recolhidos no *frontend*. De seguida, na linha 10 itera-se por cada *frame* de cada vídeo e, na linha 11, extraem-se os vários *keypoints*, com recurso ao método "extract_keypoints", retratado no excerto de código 6.6.

```
1  def extract_keypoints(pose, face, right_hand, left_hand):
2      face_array = np.array([[res['x'], res['y'], res['z']] for res in face]).flatten() if
       face and len(
3          face) == 468 else np.zeros(
4          468 * 3)
5      pose_array = np.array(
6          [
7              [0, 0, 0, 0] if res is None else
8              [res['x'], res['y'], res['z'], res['visibility']]
9              for res in pose]).flatten() if pose and len(pose) == 33 else np.zeros(132)
10
11     lh = np.array([[res['x'], res['y'], res['z']] for res in left_hand]).flatten() if
       left_hand and len(
12         left_hand) == 21 else np.zeros(21 * 3)
13     rh = np.array([[res['x'], res['y'], res['z']] for res in right_hand]).flatten() if
       right_hand and len(
14         right_hand) == 21 else np.zeros(21 * 3)
15     return np.concatenate([pose_array, face_array, lh, rh])
```

Listing 6.6: Excerto de código do método "extract_keypoints"

Este método é essencial e de fácil compreensão. O seu objetivo é simplesmente extrair as coordenadas dos vários *keypoints* enviados pelo *frontend* e criar novos

*arrays*, onde as coordenadas nulas são convertidas para zero. No final, na linha 15, retorna-se um único *array*, contendo todos os restantes *arrays* numa única posição.

Regressando ao excerto de código 6.5, após a extração dos *keypoints* para uma estrutura de dados própria, estes são adicionados ao objeto "sequence" e, assim que possua trinta elementos, ou seja um vídeo completo, o próximo passo é carregar o modelo. Isto é executado nas linhas 16 e 17. Na linha 16 é utilizado o método "build_model", observado no excerto de código 6.7, de forma a construir e instanciar o modelo. Na linha 17 o método "load_weights", nativo da *framework* Tensorflow, carrega a informação do ficheiro "actions.h5" para o modelo. Este ficheiro é gerado quando o modelo é processado e treinado, procedimento este explorado na secção 6.5. Para terminar esta fase, na linha 18, o método "predict", também nativo do Tensorflow, é invocado para determinar qual é a tradução para o gesto realizado com base no modelo de inteligência artificial. De seguida, na linha 19, adiciona-se à frase a construir o novo resultado obtido.

```
1  def build_model(all_actions):
2      model = Sequential()
3      model.add(LSTM(64, return_sequences=True, activation='relu', input_shape=(30, 1662)))
4      model.add(LSTM(128, return_sequences=True, activation='relu'))
5      model.add(LSTM(64, return_sequences=False, activation='relu'))
6      model.add(Dense(128, activation='relu'))
7      model.add(Dense(64, activation='relu'))
8      model.add(Dense(all_actions.shape[0], activation='softmax'))
9      model.compile(optimizer='Adam', loss='categorical_crossentropy', metrics=['
       categorical_accuracy'])
10     return model
```

Listing 6.7: Excerto de código do método "build_model"

Para terminar a exploração desta *user story*, falta analisar o método "build_model". Ao observar o excerto de código 6.7, é possível verificar que o modelo utilizado é composto por três camadas *LSTM* e três camadas *Dense*. Para além disto, a informação mais relevante prende-se com as funções de *activation* utilizadas, bem como o *optimizer* selecionado. Foi decidido que o *optimizer* a utilizar seria o "Adam" e a função de *activation* "relu", com exceção da última camada, onde se utiliza "softmax", uma vez que o objetivo nesta é a obtenção de uma probabilidade para cada ação disponível, de modo a ser possível determinar qual a "vencedora". A seleção de "Adam" e "relu" baseou-se, essencialmente, num conjunto de testes feitos com várias opções, nos quais esta combinação se demonstrou a mais eficaz, obtendo uma precisão muito superior aos demais.

O presente modelo necessita de ser treinado, de modo a ser possível utilizar o mesmo, sendo analisado este processo na secção 6.5.

## 6.4 User Story 04

Apesar do processo de tradução já ter sido explorado ao longo da secção 6.3, é essencial treinar, primeiramente, o sistema para que seja capaz de realizar as traduções. Assim sendo, a presente secção foca-se no desenvolvimento do módulo "PracticeComponent", responsável por processar e treinar o modelo (US04 - como administrador quero treinar o sistema).

Iniciando novamente o fluxo de execução pelo *frontend*, esta operação é exclusiva a administradores, visto que é um procedimento sensível e crucial para o bom funcionamento de todo o sistema, pelo que apenas pode ser realizado por alguém com credenciais para tal. Deste modo, o administrador, após a sua autenticação no sistema, seleciona no menu, exposto na figura 6.4, a opção "Treinar o Sistema". Ao pressionar este botão, é apresentado um pequeno formulário, como o que se verifica na figura 6.5.

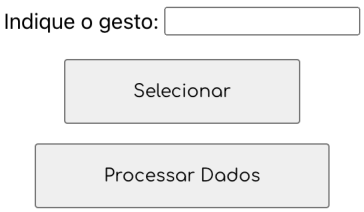


FIGURA 6.5: Formulário relativo ao treino do sistema

Explorando a figura 6.5, nota-se a presença de apenas um campo que necessita de dados, acompanhado pela *label* "Indique o gesto". Aqui, o administrador introduz o significado em língua portuguesa do novo gesto que o sistema irá suportar, como por exemplo, "Olá". Após o preenchimento desta informação, o administrador prossegue a pressionar o botão "Selecionar". Este botão irá presentear uma página muito semelhante à retratada na figura 6.3, no entanto, sem a caixa de texto central e sem o botão para desligar a câmara. Neste momento, apesar do sistema saber qual o significado do novo gesto, este ainda não conhece o gesto em si. Posto isto, nesta fase, o administrador grava sessenta vídeos ilustrativos do novo gesto.

O número sessenta surgiu após a execução de testes com quantidades distintas de vídeos. Concluiu-se, assim, que, a partir deste valor, foi possível obter previsões com precisão elevada e sem despender grandes quantidades de tempo a construir o *dataset*.

De modo a proceder à gravação dos vídeos, o administrador seleciona o botão azul com a câmara que, tal como na página de tradução, inicia a câmara do dispositivo. A gravação é acompanhada pela informação do número do vídeo atual, de modo a ser possível conhecer a quantidade em falta. Mais uma vez, cada vídeo é composto por trinta *frames* e, ao fim de cada vídeo, é feita uma pausa de 2 segundos, devidamente assinalada por uma mensagem, de forma ao administrador se reposicionar para efetuar o novo gesto. A figura 6.6 ilustra uma destas pausas. Nesta figura, é possível verificar que o vídeo a ser gravado era o quarto de sessenta no total e a mensagem a verde "A Colecionar" é indicativa de que o sistema se encontra em pausa.

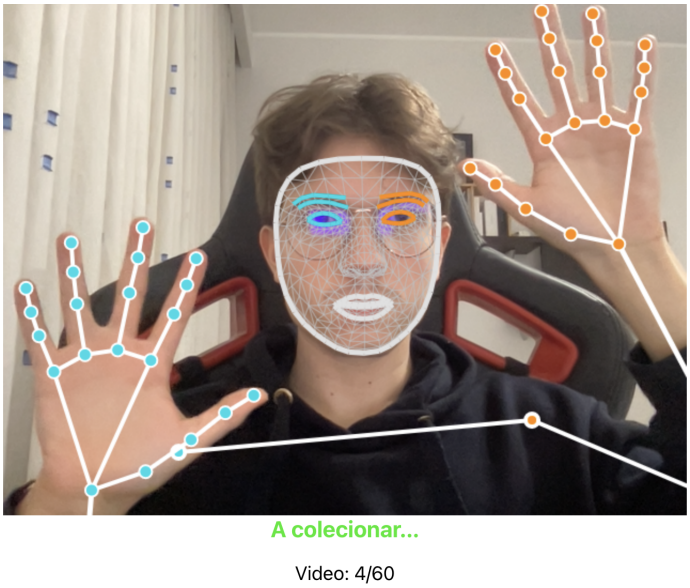


FIGURA 6.6: Procedimento de treino do sistema

Assim que os sessenta vídeos estejam concluídos, a câmara desliga-se automaticamente e a aplicação regressa ao formulário presenta na figura 6.5. Para além disto, é realizado um pedido REST ao componente "PracticeComponent", com o objetivo de adicionar à base de dados de gestos uma nova entrada e armazenar a informação relativa aos vídeos gravados. O corpo do pedido REST é constituído pelo significado dos novos gestos, pelo número de vídeos gravados (que é enviado caso no futuro se deseje alterar este valor), pelo nome do administrador autenticado no sistema, pelo *token* de segurança e por um *array* que, em cada posição, armazena o número do vídeo e a lista de *keypoints* de cada *frame*. No total, existirão sessenta conjuntos de *keypoints* para cada vídeo. No *backend*, o *controller* envia esta informação para o serviço correspondente, representado no excerto de código 6.8.

```
def add_gesture(logged_user, token, action, nr_videos, data_map):
    if validate(token, logged_user):
        create_folder(action, nr_videos)
        # insert new Action in DB
        mongo.db.Actions.insert_one({'name': action})
        video = 0
        indexes = []
        while video < nr_videos:
            for data in data_map:
                if data['key'] == video:
                    frame = data['value']
                    results = extract_keypoints(frame['pose'], frame['face'], frame['
right_hand'], frame['left_hand'])
                    indexes.append(data['key'])
                    frame_index = len(indexes) - 1
                    npy_path = os.path.join(DATA_PATH, action, str(video), str(frame_index
))
                    np.save(npy_path, results)
            video += 1
            indexes.clear()
        return {'result': 'New action added!'}
    else:
        return {'result': 'Invalid token'}
```

LISTING 6.8: Excerto de código do método "add_gesture"

O método "add_gesture" 6.8 começa por validar o *token*, invocando o método "validate", presente na linha 2, de uma classe utilitária. Este método descodifica o *token* e verifica a validade dos dados, bem como a correspondência dos mesmos com o utilizador autenticado no sistema.

De seguida, a execução prossegue e, na linha 3, utiliza-se um novo método, denominado "create_folder". Este é responsável pela criação de uma nova pasta num caminho específico para o novo gesto. Esta pasta armazena toda a informação recolhida dos vídeos gravados. Assim, caso o novo gesto signifique "Olá", uma nova pasta "Olá" será criada e, no interior desta, serão geradas sessenta novas pastas, uma para cada vídeo. Cada pasta, numerada de 0 a 59, armazena trinta ficheiros, cada um com informação relativa a um *frame*. Todas estas pastas serão utilizadas para efetivamente treinar o modelo de IA, abordado, com mais detalhe, mais adiante da presente secção.

Continuando a análise do método "add_gesture", após a criação das pastas, na linha 5, adiciona-se o novo gesto à base de dados não relacional MongoDB. De seguida, itera-se pela informação dos vídeos ao longo das linhas 8 a 17. Nestes ciclos, para cada vídeo, são extraídos os *keypoints*, com o método "extract_keypoints", já explorado na secção 6.3. Na linha 15 adicionam-se os ficheiros com a informação dos *frames* às pastas que foram mencionadas no parágrafo anterior. A variável "DATA_PATH" contém o caminho para onde estas novas pastas e ficheiros devem ser criadas. Por último, na linha 19 retorna-se uma mensagem de sucesso, ou de insucesso, na linha 21, caso o *token* recebido não seja válido.

O administrador pode repetir este processo o número de vezes que quiser, adicionando vários gestos ao sistema. Porém, o modelo de IA não se encontra, ainda, pronto para efetuar as traduções destes gestos. Primeiramente, é necessário realizar o processamento dos dados e, para tal, o administrador necessita de pressionar o botão "Processar Dados" retratado na figura 6.5. Este botão leva a que um novo pedido REST seja feito ao componente "PracticeComponente", de modo a treinar o sistema, tendo em conta todos os gestos existentes na base de dados. O pedido é simples, contendo apenas o *token* e o nome de utilizador, e tendo como único objetivo notificar o *backend* de que pode iniciar o treino do modelo. O *controller* recebe o pedido e, após retirar a informação necessária, envia-a para a camada dos serviços, onde a lógica essencial deste processo se encontra, nomeadamente no método "train_system", demonstrado no excerto de código 6.9.

```
1  def train_system(token, logged_user):
2      if validate(token, logged_user):
3          actions = []
4          all_db_actions = mongo.db.Actions.find();
5          for db_action in all_db_actions:
6              actions.append(db_action['name'])
7          all_actions = np.array(actions)
8          all_videos, labels = [], []
9          label_map = {label: num for num, label in enumerate(all_actions)}
10         for action in all_actions:
11             for video in range(no_video):
12                 single_video = []
13                 for frame_num in range(video_length):
14                     # load frame
15                     frame = np.load(os.path.join(DATA_PATH, action, str(video), "{}.npy".
       format(frame_num)))
16                     # build video
17                     single_video.append(frame)
18                 all_videos.append(single_video)
19                 # add label to labels array
20                 labels.append(label_map[action])
21         x = np.array(all_videos)
22         y = keras.utils.np_utils.to_categorical(labels).astype(int)
23         x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.25)
24         model = train_lstm_neural_network(all_actions, x_train, y_train)
25         accuracy = evaluate_model(model, x_test, y_test)
26         return {'result': accuracy}
27     else:
28         return {'result': 'Invalid token'}
```

Listing 6.9: Excerto de código do método "train_system"

Este método, à semelhança do "add_gesture", inicia-se com a validação do *token*, visto ser uma funcionalidade restrita a administradores. De seguida, na linha 3, comunica com a base de dados, de modo a recolher todos os gestos a serem suportados pelo modelo de IA. Estes gestos são todos armazenados num *numpy array* [3] na linha 7. Posteriormente, na linha 9, o dicionário "label_map" é instanciado, onde as chaves são as traduções dos gestos retiradas da base de dados e os respetivos valores são números inteiros únicos dentro do dicionário. Ou seja, o dicionário fica com valores num formato semelhante ao seguinte: "'Olá': 0, 'Sim': 1, 'Não': 2". O passo seguinte é preparar e estruturar a informação para que o modelo a possa utilizar. Este exercício começa por carregar todas as *frames* de todos os vídeos gravados, para os armazenar no *array* "all_videos". Simultaneamente, ao preencher esta lista, também se constrói o *array* "labels", que armazena o identificador de cada gesto presente no dicionário "label_map". Os *arrays* "all_videos" e "label_map" constituem, respetivamente, as dimensões X e Y do modelo de IA. Nas linhas 21 e 22, as variáveis representativas destas dimensões são criadas com base nestas duas listas, de modo a treinar o modelo, realizado na linha 24 com o método "train_lstm_neural_network". No entanto, antes disso, com recurso à função "train_test_split", fornecida pela biblioteca "sklearn", a informação de treino e a informação de teste são separadas, como se verifica na linha 23. Foi definido que 25% dos dados seriam utilizados para testes, enquanto que o restante seria utilizado para treino, daí as variáveis "x_train", "x_test", "y_train" e "y_test". As variáveis que terminam com o sufixo "train" possuem os dados de treino, enquanto que as que terminam com o sufixo "test" contêm os de teste.

[3] *Arrays* fornecidos pela biblioteca Numpy, sendo mais rápidos, compactos e necessitando de menos memória do que outras estruturas nativas do Python, sendo otimizados para computação científica e numérica [61].

```
1 def train_lstm_neural_network(actions, x_train, y_train):
2     log_dir = os.path.join('Logs')
3     tb_callback = TensorBoard(log_dir=log_dir)
4     model = build_model(actions)
5     model.fit(x_train, y_train, epochs=100, callbacks=[tb_callback], batch_size=32)
6     model.summary()
7     model.save("action.h5")
8     return model
```

Listing 6.10: Excerto de código do método "train_lstm_neural_network"

O método "train_lstm_neural_network" encontra-se representado ao longo do excerto de código 6.10. Este método recebe por parâmetro a lista "actions", contendo o conjunto de todas as traduções de gestos suportadas pelo sistema, bem como o conjunto de dados destinados a treino ("x_train" e "y_train"). No que concerne ao fluxo de execução do método, as duas primeiras linhas destinam-se essencialmente a armazenar ficheiros de *log* para serem utilizados pela "Tensorboard", ferramenta do Tensorflow explorada na secção 2.6.1. A linha 3 não apresenta nada de novo. O modelo de IA é construído com base no método "build_model" já representado na figura 6.7 e analisado na secção 6.3. Na linha 5 é realizado o treino propriamente dito com a função "fit", que recebe por parâmetro os dados relativos ao treino ("x_train" e "y_train"), o número de *epochs*, a função *callback* e o *batch size*. O número de *epochs* representa o número de vezes que se percorre o *dataset* inteiro ao treinar o sistema. Neste caso, foi determinado que cem seria uma quantidade suficiente para gerar resultados com precisão elevada. A função *callback* indicada foi a "tb_callback", definida na linha 3, de modo a tornar possível a verificação da evolução do treino do modelo na "Tensorboard". A figura 6.7 demonstra um gráfico presente na "Tensorboard", representativo da evolução da precisão do modelo.

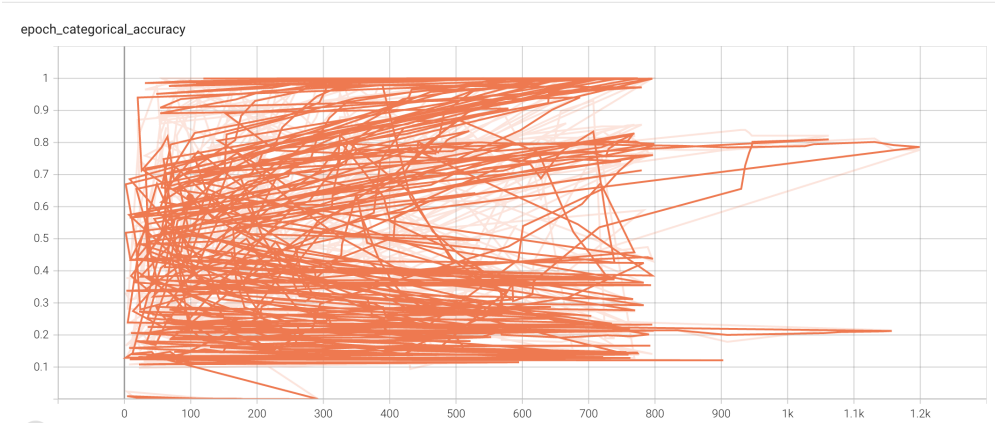


Figura 6.7: Evolução gráfica da precisão do modelo de IA

O eixo "X" representa o número de *epochs* e o eixo "Y" espelha a precisão do modelo, que varia entre 0 e 1 (ou 0% a 100%). Neste gráfico, é possível notar as várias experiências realizadas com valores distintos, verificando-se que com cem *epochs* é possível alcançar uma precisão à volta dos 90%. Para além da precisão alcançada, é importante considerar o tempo necessário para treinar o modelo. Assim, quanto maior for o número de *epochs*, maior será o tempo necessário para realizar esta

operação. Deste modo, considera-se ter alcançado um bom equilíbrio entre tempo e qualidade do modelo.

Ao treinar o modelo, para além da métrica da precisão, também se avalia a perda. O objetivo é diminuir a perda, através do aumento da precisão. A evolução desta métrica encontra-se detalhada no gráfico presente na figura 6.8, onde o eixo "X" representa o número de *epochs* e o eixo "Y" detalha a perda do modelo. Verifica-se, uma vez mais, que para cem *epochs*, a perda atinge um valor de quase 0%, o que é o desejado.

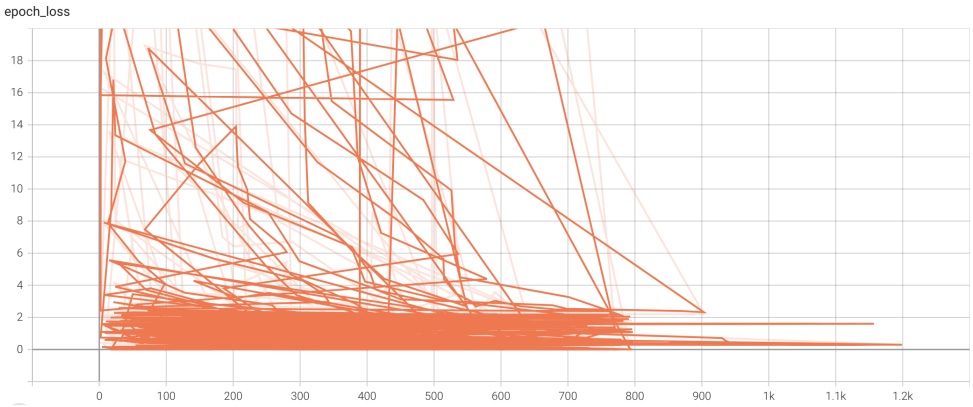


FIGURA 6.8: Evolução gráfica da perde de *epochs* do modelo de IA

Por último, o *batch size* representa o número de amostras do *dataset* a ser processadas por cada iteração a este. O valor deste parâmetro deve ser maior ou igual a um e menor ou igual ao valor total de amostras no *dataset*. O valor trinta e dois foi determinado após vários testes, permitindo um processo mais rápido e com resultados positivos.

Regressando ao excerto de código 6.10, a linha 6 apenas é utilizada para imprimir um pequeno sumário ilustrativo do treino do modelo, presente na figura 6.9. Aqui, é possível observar as várias camadas constituintes do modelo, bem como a quantidade de parâmetros que passa por cada uma delas. No fim, verifica-se que, no total, foram utilizados à volta de seiscentos mil parâmetros, o que no mundo de *deep learning* é um valor consideravelmente baixo para a precisão de cerca de 90% alcançada.

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
lstm (LSTM)                  (None, 30, 64)            442112
_________________________________________________________________
lstm_1 (LSTM)                (None, 30, 128)           98816
_________________________________________________________________
lstm_2 (LSTM)                (None, 64)                49408
_________________________________________________________________
dense (Dense)                (None, 128)               8320
_________________________________________________________________
dense_1 (Dense)              (None, 64)                8256
_________________________________________________________________
dense_2 (Dense)              (None, 7)                 455
=================================================================
Total params: 607,367
Trainable params: 607,367
Non-trainable params: 0
```

Figura 6.9: Sumário do treino do modelo de IA

Para terminar, na linha 7, preserva-se o modelo e, na linha 8, é retornado de volta para o método "train_system". Neste método apenas falta invocar a função "evaluate_model", que tem como objetivo calcular a precisão com base nos dados de teste. Este valor é retornado na linha 26 para o *controller*, que por sua vez envia este valor de volta para o *frontend*. Aqui, conforme o valor obtido, apresenta-se uma mensagem ao administrador. No caso do valor da precisão ser inferior a 80%, sugere-se que se volte a treinar o sistema. Por outro lado, indica-se que o modelo encontra-se pronto para realizar traduções com uma precisão elevada.

## 6.5 User Story 05

O último requisito funcional (US05 - como utilizador quero preencher inquéritos de satisfação e usabilidade para demonstrar o meu agrado/desagrado com o sistema nestes tópicos) não é menos importante que os anteriores, uma vez que é o único método para determinar a satisfação dos utilizadores perante o sistema e o seu respetivo nível de usabilidade. Estes índices são de elevada relevância, visto que o público-alvo possui défices auditivos, sendo necessário realizar um esforço extra para o projeto se adequar às suas necessidades. Deste modo, é possível determinar a eficácia e pertinência da abordagem selecionada, de forma a ser melhorada.

Esta *user story* encontra-se disponível para qualquer utilizador, pelo que, após entrar no sistema, basta selecionar a opção "Responder a Inquéritos" presente no menu exposto na figura 6.4. Ao pressionar esta opção, a página relativa aos inquéritos é disponibilizada. Esta página, tal como se verifica na figura 6.10, possui um vídeo no centro como forma de introduzir o inquérito e o seu modo de funcionamento. Ao carregar no botão "Iniciar" as perguntas são apresentadas no ecrã, bem como as opções de resposta disponíveis.

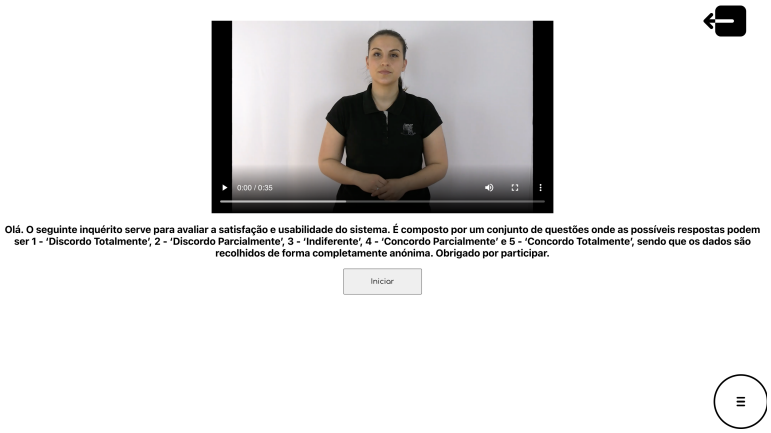


FIGURA 6.10: Página relativa aos inquéritos

De modo a que os utilizadores sejam capazes de responder aos inquéritos, sem qualquer constrangimento, as questões encontram-se todas elaboradas em LGP, em formato de vídeo, sendo acompanhados pela tradução em língua portuguesa escrita. Todos os vídeos foram realizados com a colaboração da Associação de Tradutores e Intérpretes de Língua Gestual Portuguesa (ATILGP). Para responder às questões, o utilizador necessita de escolher um valor da escala disponível, tal como espelha a figura 6.11.


FIGURA 6.11: Escala de respostas possíveis a questões do inquérito

Assim que todas as questões estejam respondidas, o utilizador termina o inquérito e é realizado um pedido REST para o componente "Analytics". O *body* deste pedido é constituído pelo objeto "InquiryDTO". Este, respeitando as boas práticas de um DTO, é muito simples, uma vez que possui apenas uma lista, onde, em cada posição, se encontra um objeto com o identificador das questões (armazenadas numa base

de dados H2) e a respetiva resposta. Do lado do *backend*, o *controller* "InquiryController", seguindo a norma, recebe o pedido e envia a informação necessária para o serviço "InquiryService", presente no excerto de código 6.11.

```
1  public InquiryDTO createInquiry(InquiryDTO inquiryDTO) {
2          try{
3              List<AnswerQuestionDTO> listOfAnswerQuestionDTO =  inquiryDTO.
       getListOfAnswerQuestionDTO();
4              List<AnswerQuestion> listOfAnswerQuestion =  new ArrayList<>();
5              for (AnswerQuestionDTO aq: listOfAnswerQuestionDTO) {
6                  Optional<Question> question = questionRepository.findById(aq.getQuestionId
       ());
7                  if(question.isPresent()) {
8                      listOfAnswerQuestion.add(AnswerQuestionMapper.DTO2Domain(question.get
       (), aq.getAnswerDTO()));
9                  }
10             }
11             Inquiry inquiry = new Inquiry(listOfAnswerQuestion);
12             inquiryRepository.save(inquiry);
13             return InquiryMapper.Domain2DTO(inquiry);
14         } catch(IOException e) {
15             inquiryDTO.setId(0L);
16             return inquiryDTO;
17         }
18     }
```

Listing 6.11: Excerto de código do método "createInquiry" do serviço "InquiryService"

Na linha 1 do método "createInquiry" extrai-se a lista de perguntas e respostas do DTO, para que, na linha 5, se itere por ela com um ciclo *for*. Neste ciclo, para cada par questão/resposta, comunica-se com a camada de persistência através do objeto "questionRepository", instanciado por *dependency injection*, de modo a carregar a questão correspondente ao identificador. Nesta camada, é aplicado o padrão *repository*, de maneira a tornar a lógica de comunicação com a base de dados restrita na camada de persistência. De seguida, na linha 7, averigua-se se a base de dados retorna alguma questão ou não. Em caso positivo, na linha 8, com recurso ao *mapper* "AnswerQuestionMapper", mapeia-se o DTO para o respetivo objeto de domínio e adiciona-se à lista de pares questão/resposta constituinte do inquérito a ser armazenada na base de dados. No fim do ciclo, o inquérito é criado, na linha 11, com a lista de questões/respostas construída ao longo do *loop* anterior. Na linha 12 este objeto é preservado na base de dados. Para concluir, na linha 13, o *mapper* transforma o objeto de domínio "inquiry" num DTO e retorna-o para o *controller* que, por sua vez, o retorna para o *frontend*, terminando, assim, o fluxo deste requisito funcional.

## 6.6 Construção de uma PWA

Tal como já foi explorado na secção 2.8, o objetivo é construir uma *Progressive Web App* e, para tal, é necessário seguir alguns passos. Em primeiro lugar, é fundamental determinar a forma de avaliar esta métrica, tendo-se optado pelo "Lighthouse", *plugin* do Google Chrome. Este constitui uma ferramenta utilizada para otimizar a execução e qualidade de aplicações *web* [62]. "Lighthouse" permite gerar relatórios de várias categorias, entre elas "Progressive Web Apps", a mais relevante para o projeto em questão. Deste modo, para que seja possível obter o certificado de PWA do "Lighthouse", é fundamental cumprir as seguintes etapas:

- Construir um *manifest* - ficheiro JSON utilizado para especificar detalhes da aplicação como, por exemplo, a sua descrição;
- Definir *service workers* - permite ao *browser* garantir que a aplicação funciona em modo *offline*.

Concluídos estes passos, é possível gerar o relatório do "Lightouse" e obter o certificado de PWA, apresentado na figura 6.12.

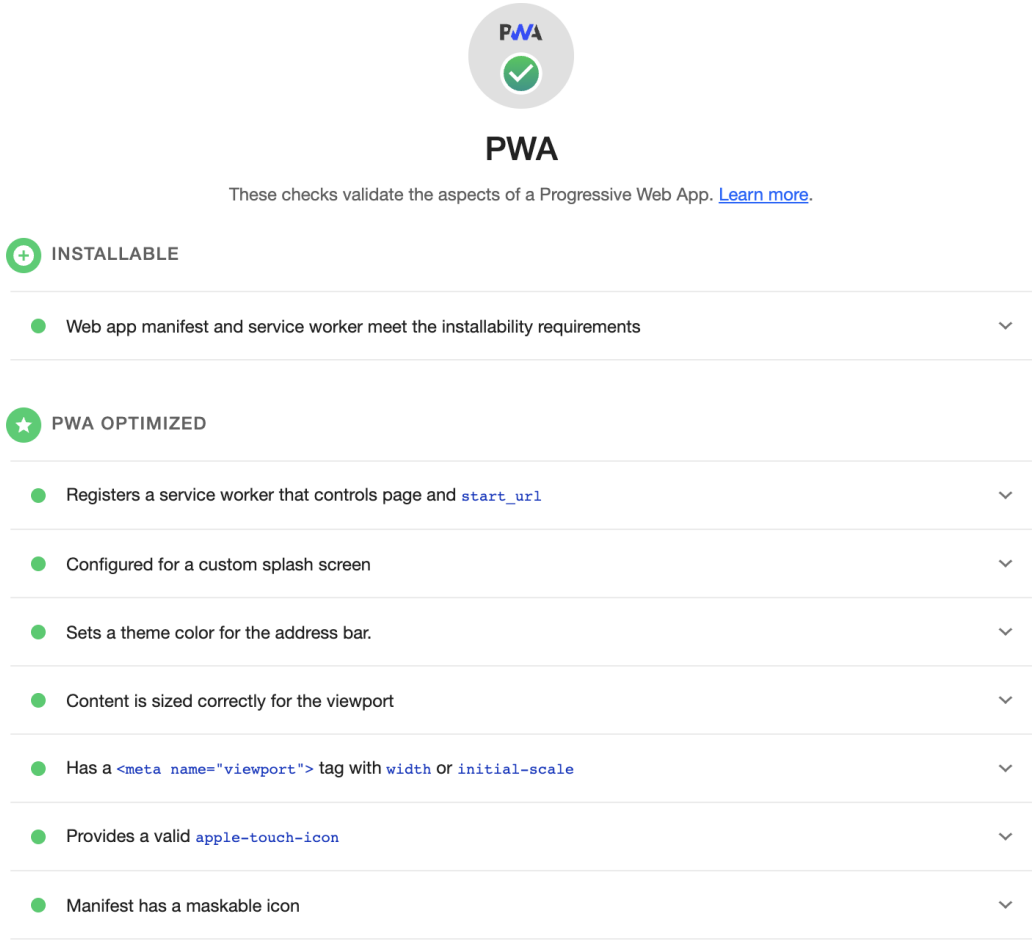


FIGURA 6.12: Certificado PWA

Assim que a aplicação se enquadrar nos requisitos de uma PWA, ao abri-la no *browser*, surge a opção de a descarregar para o dispositivo. A figura 6.13 representa o ícone da aplicação no ambiente de trabalho de um computador.


FIGURA 6.13: Ícone da aplicação

A aplicação descarregada possui um funcionamento e aparência iguais à aplicação *web*. Deste modo, é possível aproveitar todas as vantagens proporcionadas pela PWA.

## 6.7 Sumário

Esta capítulo demonstra o processo de implementação das várias *user stories*. As duas primeiras foram simples de desenvolver, sem terem oferecido problemas adicionais aos inicialmente planeados. Os desafios surgiram na implementação da terceira *user story*, associada ao processo de tradução em si.

O processo de tradução inicia-se com o utilizador a ligar a câmara do dispositivo e a efetuar os restos. O sistema assume uma pausa de dois segundos entre gestos para que o utilizador se possa reposicionar para realizar o gesto seguinte. Todos os gestos são gravados em vídeos de trinta *frames*, enviados e processados no módulo TranslatorBackend. Aqui, o sistema extrai os dados dos vários vídeos e utiliza o modelo de IA construído para realizar as traduções, traduzindo a frase para língua portuguesa.

De modo a que o modelo de IA consiga efetuar as traduções, é, inicialmente, essencial treiná-lo, o que diz respeito à quarta *user story*. Esta é limitada a administradores, uma vez que não é qualquer utilizador que deve ter o poder de influenciar o funcionamento do sistema. Para treinar o modelo, o administrador indica a tradução dos novos gestos e grava um conjunto de sessenta vídeos (cada um com trinta *frames*). O PracticeModule encarrega-se de treinar o modelo de IA, com auxílio da *framework* Tensorflow. Todas as variáveis envolvidas neste processo foram determinadas com base no teste e avaliação de resultados obtidos. No momento da escrita do presente relatório, o modelo atinge uma percentagem de precisão que ronda os 90%.

Por último, a quinta *user story* foi mais simples de desenvolver, comparativamente às duas anteriores. O utilizador seleciona a opção para responder aos inquéritos, sendo-lhe apresentadas as questões em formato de vídeo, com a colaboração de uma intérprete, que realiza todas as perguntas em LGP. As questões são antecedidas de uma pequena introdução explicativa de todo o processo. Terminado o inquérito, as respostas são recolhidas e enviadas para o componente Analytics, que se encarrega do resto do processamento.

# Capítulo 7

# Experimentação e Avaliação

O presente capítulo começa por retratar os identificadores de avaliação selecionados para avaliar o projeto desenvolvido. De seguida, apresentam-se as hipóteses construídas, bem como as metodologias de avaliação. Por último, efetua-se a associação entre os indicadores e os métodos de avaliação identificados.

## 7.1 Identificadores de Avaliação

Os identificadores de avaliação pretendem ajudar a avaliar o projeto construído no âmbito do presente relatório. Assim, os identificadores eleitos foram os seguintes:

- Cumprimentos de Requisitos Funcionais e Não Funcionais - este identificador permite identificar se os requisitos detalhados nas secções 4.1 e 4.2 foram cumpridos;
- Usabilidade e Satisfação - a usabilidade e a satisfação são cruciais para o projeto em questão, uma vez que este é para ser utilizado por indivíduos com incapacidades auditivas. A norma ISO 9241-11 indica que a usabilidade de um sistema define se este pode ser utilizado com eficácia, eficiência e também satisfação [63]. Assim, estes dois identificadores encontram-se relacionados.

## 7.2 Definição de Hipóteses

Nesta secção, são definidas as hipóteses que possuem o objetivo de avaliar os identificadores explorados na secção 7.1.

Deste modo, foram instituídas as seguintes hipóteses:

- Total cumprimentos dos requisitos funcionais e não funcionais;
- Grau de usabilidade da aplicação com valor igual ou superior a 80%;
- Grau de satisfação dos utilizadores com valor igual ou superior a 80%.

A primeira hipótese relaciona-se com o primeiro identificador de avaliação, “Cumprimento de Requisitos Funcionais e Não Funcionais”. Assim, esta hipótese visa entender se os requisitos funcionais e não funcionais, detalhados nas secções 4.1 e 4.2, respetivamente, foram completamente implementados, uma vez que são todos de elevada importância.

A segunda hipótese encontra-se associada ao indicador do grau de usabilidade da aplicação, devendo ser igual ou superior a 80%. A escolha deste valor baseou-se no sistema *Customer Satisfaction Score* (CSAT). O CSAT é um indicador de desempenho utilizado para medir a satisfação e outras métricas, com base no *input* dado pelos utilizadores. Neste caso, será aplicado para determinar o grau de usabilidade [64]. Segundo o CSAT, os resultados que alcancem o valor de 80%, ou superior, indicam que os utilizadores consideraram o sistema fácil de utilizar [64].

Tendo isto em conta, foi determinado que o resultado de 80% para o grau de usabilidade seria o valor a cumprir.

No que toca à última hipótese, esta baseia-se no grau de satisfação do utilizador. De modo a manter a consistência, recorreu-se, também, ao sistema CSAT para realizar a medição da satisfação, pelo que o valor pretendido deve ser, também, igual ou superior a 80%.

## 7.3 Metodologia de Avaliação

Esta secção serve para apresentar ao leitor as várias metodologias de avaliação a serem utilizadas.

### 7.3.1 Testes de *Software*

Existem vários tipos de testes utilizados para comprovar a qualidade do *software* desenvolvido. No entanto, uma vez que o essencial é demonstrar que os requisitos foram cumpridos, são essencialmente executados testes funcionais.

Os testes funcionais são definidos de acordo com os requisitos funcionais do *software*. Considera-se que não existe conhecimento sobre o interior do programa, estando o foco concentrado nos *outputs* dos vários casos de uso. Estes testes refletem a ótica do utilizador [65] e são especialmente vantajosos para encontrar erros de comportamento ou desempenho, bem como falhas na interface gráfica [65]. Para realizar estes testes, são desenvolvidos guiões com os vários passos que um utilizador normalmente tomaria, bem como o resultado esperado de cada um deles.

### 7.3.2 Inquéritos de Satisfação

No sentido de analisar a satisfação do utilizador, foi criado um questionário com o objetivo de inquirir sobre este tema. Decidiu-se utilizar esta abordagem, uma vez que o nível de satisfação é considerado um indicador subjetivo.

O inquérito segue uma estrutura com perguntas simples e concretas, de modo a não incluir ambiguidade como uma variável na equação, existindo, apenas, respostas fechadas com as seguintes cinco opções:

- Discordo totalmente;
- Discordo parcialmente;
- Indiferente;
- Concordo parcialmente;

- Concordo totalmente.

Este conjunto de possíveis respostas encontra-se de acordo com a escala de "Likert" [66]. Esta escala foi selecionada de forma a ser possível oferecer uma resposta intermédia ao utilizador, no caso deste não se sentir particularmente direcionado para o lado positivo ou negativo do espetro. Assim, garante-se uma forma confiável de medir o grau de satisfação.

Tendo em conta o público-alvo do projeto, este inquérito é apresentado na forma de vídeo, integrado na solução construída, onde as questões são realizadas em LGP, para além do típico formato em texto.

### 7.3.3 Inquéritos de Usabilidade

No sentido de avaliar a usabilidade do sistema, foi elaborado um questionário que demonstra as dificuldades que os utilizadores apresentam ao utilizar a aplicação construída. Este segue as mesmas linhas que o inquérito de satisfação, baseando-se na escala de "Likert", sendo as várias questões exibidas em LGP.

As perguntas que constituem este inquérito foram inspiradas no questionário *System Usability Scale* de John Brooke, que possui o objetivo de medir o grau de satisfação de produtos, serviços, etc. Este inquérito é vantajoso, devido à facilidade em apresentar o mesmo aos utilizadores, à possibilidade de o utilizar em amostras de tamanho reduzido e, acima de tudo, à sua validade. [67].

### 7.3.4 Testes de Hipótese

Os testes de hipótese correspondem a procedimentos estatísticos que se baseiam na análise de uma amostra a partir da teoria da probabilidade, procurando avaliar os parâmetros desconhecidos de uma população [68]. Assim, definem-se duas hipóteses:

- Hipótese nula ($H_0$) - corresponde à hipótese que representa a ausência do efeito que se deseja verificar [68];
- Hipótese alternativa ($H_1$) - corresponde à hipótese que o investigador deseja verificar [68].

Para que seja possível determinar a rejeição da hipótese nula, utiliza-se o *p-value* (valor de prova). Este representa a probabilidade de se obter uma estatística do teste que seja igual ou mais extrema do que a estatística encontrada na amostra observada, sob a suposição de que $H_0$ é verdadeira. Caso *p-value* $<= \alpha$ (nível de significância do teste), rejeita-se $H_0$. Caso contrário, não é possível fazê-lo [68].

Existem dois tipos de testes para representar os testes de hipóteses: testes paramétricos e testes não paramétricos. Os testes paramétricos pressupõem a normalidade dos dados, uma variância homogénea e com intervalos contínuos e iguais. Por outro lado, os não paramétricos não necessitam de tantos requisitos, podendo ser aplicados mesmo que nenhum dos pressupostos anteriores se verifique [69].

Por fim, os testes paramétricos acabam por apresentar vantagens em relação aos não paramétricos, uma vez que estes últimos não são tão fidedignos, sendo menos capazes de rejeitar a hipótese nula [69].

## 7.4 Associação de Métodos de Avaliação

A tabela 7.1 demonstra a associação entre os identificadores definidos e os métodos de avaliação delineados.

Tabela 7.1: Associação entre indicadores e métodos de avaliação

| Indicador | Método de Avaliação |
|---|---|
| Cumprimento dos Requisitos | Testes de *Software* |
| Usabilidade | Inquérito de Usabilidade e Testes de Hipótese |
| Satisfação | Inquérito de Satisfação e Testes de Hipótese |

## 7.5 Avaliação de Resultados

Ao longo desta secção apresenta-se ao leitor a avaliação dos resultados obtidos perante as hipóteses definidas na secção 7.2. Assim, são explorados os resultados advindos de cada método de avaliação realizado.

### 7.5.1 Testes de Software

Nesta secção apresentam-se e exploram-se os testes funcionais desenvolvidos e executados com a finalidade de determinar eventuais problemas no fluxo de execução de cada *user story*. Assim, foi construído um teste para cada *user story*, cinco no total. Para cada um destes testes segue-se um procedimento e um conjunto de critérios que necessitam de ser validados para o sucesso do teste. Caso algum falhe, então considera-se que o teste falha na sua globalidade.

Não se considera pertinente para o leitor apresentar todos os testes funcionais nesta secção, uma vez que algumas *user stories* acabam por ser mais relevantes que outras. Deste modo, apenas se exploram aqui os testes para três *user stories*, enquanto que os restantes podem ser visualizados no Anexo A.

O primeiro teste funcional a ser analisado é relativo à *user story* "Como utilizador quero que o sistema realize a tradução de LGP para língua portuguesa escrita" e encontra-se representado na tabela 7.2.

Tabela 7.2: Teste Funcional para a US03

| | |
|---|---|
| ***User Story*** | US03 - Como utilizador quero que o sistema realize a tradução de LGP para língua portuguesa escrita |
| **Procedimento** | O utilizador entra no sistema como convidado. O sistema apresenta a página de tradução. O utilizador seleciona o botão para ligar a câmara do dispositivo. A câmara é iniciada e começa a detetar as várias partes do corpo do utilizador. O utilizador realiza gestos de LGP e desliga a câmara. A câmara é desligada e a mensagem traduzida aparece no centro do ecrã. |
| **Critérios** | 1. O utilizador entra no sistema como convidado e o sistema apresenta a página de tradução.<br>2. O utilizador não consegue visualizar a opção de "Treinar o Sistema" no menu.<br>3. Ao carregar no botão para ligar a câmara, esta inicia.<br>4. Com a câmara ligada, o sistema demonstra que deteta as várias partes do corpo do utilizador.<br>5. Ao carregar no botão de pausa, a câmara desliga-se.<br>6. Assim que a câmara se desliga, a mensagem traduzida é apresentada no centro do ecrã. |
| **Resultados** | 1. Passou<br>2. Passou<br>3. Passou<br>4. Passou<br>5. Passou<br>6. Passou |
| **Resultado Final** | Passou |

O teste realizado para a *user story* "Como administrador pretendo treinar o sistema para que este seja mais preciso e capaz de detetar mais gestos de LGP" encontra-se exposto na tabela 7.3.

Tabela 7.3: Teste Funcional para a US04

| | |
|---|---|
| ***User Story*** | US04 - Como administrador pretendo treinar o sistema para que este seja mais preciso e capaz de detetar mais gestos de LGP |
| **Procedimento** | O administrador autentica-se no sistema e seleciona a opção "Treinar o Sistema" no menu. O sistema requisita a introdução do significado do novo gesto. O administrador indica o novo gesto e pressiona o botão "Selecionar". De seguida, o administrador pressiona o botão para ligar a câmara e realiza os gestos. Enquanto o administrador efetua os gestos, o sistema fornece *feedback* sobre os vídeos. Assim que os sessenta vídeos tiverem sido gravados, o sistema desliga a câmara e regressa à página anterior. O administrador pressiona o botão "Processar Dados". O sistema indica que a operação está a decorrer. Assim que a operação terminar, o sistema indica ao administrador a precisão do sistema, em percentagem, bem como uma sugestão consoante o valor. |
| **Critérios** | 1. Ao selecionar a opção "Treinar o Sistema" no menu, o sistema requisita a tradução do novo gesto.<br>2. Pressionar o botão "Selecionar" sem a tradução do gesto não faz nada.<br>3. Depois de indicar o gesto e pressionar o botão "Selecionar" o sistema direciona o administrador para a página de treino.<br>4. Na página de treino, depois de pressionar o botão da câmara, esta é iniciada.<br>5. O sistema fornece *feedback* sobre os vídeos.<br>6. Quando todos os vídeos tiverem sido gravados, o sistema desliga a câmara automaticamente.<br>7. Depois de desligar a câmara, o sistema regressa à página anterior.<br>8. Ao pressionar o botão "Processar Dados", o sistema fornece *feedback* sobre a operação.<br>9. Quando a operação termina, o sistema indica se o sistema deve indicar a precisão do modelo de IA, bem como uma sugestão. |
| **Resultados** | 1. Passou<br>2. Passou<br>3. Passou<br>4. Passou<br>5. Passou<br>6. Passou<br>7. Passou<br>8. Passou<br>9. Passou |
| **Resultado Final** | Passou |

Por último, o teste para a *user story* "Como utilizador quero preencher inquéritos de satisfação e usabilidade para demonstrar o meu agrado/desagrado com o sistema nestes tópicos" está representado na tabela 7.4.

Tabela 7.4: Teste Funcional para a US05

| | |
|---|---|
| ***User Story*** | US05 - Como utilizador quero preencher inquéritos de satisfação e usabilidade para demonstrar o meu agrado/desagrado com o sistema nestes tópicos |
| **Procedimento** | O utilizador entra no sistema como convidado e seleciona a opção "Responder a Inquéritos" no menu. De seguida, o sistema apresenta um vídeo introdutório ao inquérito. Para cada vídeo é possível iniciá-lo, fazer pausa e aumentar o seu tamanho. Cada vídeo é também acompanhado da sua tradução para língua portuguesa. Ao pressionar o botão 'Ìniciar" as questões do inquérito são apresentadas e é possível responder a cada uma delas numa escala de 1 a 5 estrelas. Quando o inquérito é terminado, o sistema regressa à página de tradução. |
| **Critérios** | 1. O sistema apresenta o vídeo introdutório ao selecionar a opção "Responder a Inquéritos".<br>2. É possível iniciar, parar e aumentar o tamanho de cada vídeo.<br>3. Todos os vídeos encontram-se acompanhados pela sua tradução em língua portuguesa.<br>4. É possível responder ás questões numa escala de 1 a 5 estrelas.<br>5. Depois de responder a todas as questões, o sistema regressa à página de tradução. |
| **Resultados** | 1. Passou<br>2. Passou<br>3. Passou<br>4. Passou<br>5. Passou |
| **Resultado Final** | Passou |

### 7.5.2 Inquérito de Satisfação

Os inquéritos de satisfação e de usabilidade são visualizados pelo sistema como apenas um único inquérito global, pelo que os utilizadores respondem de uma só vez

a todas as questões. Ainda assim, para que a análise dos graus de satisfação e de usabilidade seja mais fácil de analisar, foi decidido que no presente capítulo seriam considerados dois inquéritos distintos. Uma vez que o inquérito global é constituído por cinco questões de satisfação e cinco de usabilidade, as primeiras são exploradas nesta secção, enquanto que as restantes apenas são estudadas na secção 7.5.3.

No total, o inquérito conta com cinco respostas de indivíduos distintos, constituindo uma amostra reduzida. No entanto, apesar disto, considera-se relevante a avaliação destes dados, uma vez que, apesar de reduzida, a amostra demonstra o *feedback* de utilizadores surdos.

Tal mencionado anteriormente, o inquérito de satisfação é composto por cinco questões que são analisadas individualmente, com base na hipótese relativa à satisfação dos utilizadores, definida na secção 7.2. Cada questão é avaliada a partir de testes de hipótese, assumindo um grau de confiança de 95% e um nível de significância de 0.05.

Os testes de hipótese foram todos realizados no ambiente de desenvolvimento RStudio, utilizando a linguagem de programação R. As respostas aos inquéritos formam uma amostra independente, quantitativa e com uma dimensão igual ao número de inquiridos (cinco). Antes de aplicar qualquer teste de hipóteses, é fundamental verificar a normalidade dos dados. Posto isto, a análise de cada questão apresenta-se de seguida.

**Questão 1** - A aplicação deteta os gestos efetuados?

Como já foi explicado, o primeiro passo é determinar se os dados da amostra seguem uma distribuição normal. Para este efeito, utiliza-se o teste de normalidade *Shapiro-Wilk*, criado por Samuel Shapiro e Martin Wilk, [70]. As duas hipóteses definidas são:

- ($H_0$) - Os valores da amostra seguem uma distribuição normal.
- ($H_1$) - Os valores da amostra não seguem uma distribuição normal

O excerto de código 7.1 retrata o teste de *Shapiro-Wilk* empregue para os dados da questão 1.

```
1 > q1 <- c(4,4,5,5,5)
2
3 > shapiro.test(q1)
4
5   Shapiro-Wilk normality test
6
7 data:  q1
8 W = 0.68403, p-value = 0.00647
```

Listing 7.1: Excerto de código do teste de normalidade *Shapiro-Wilk* aplicado à questão 1 do Inquérito de Satisfação

A primeira linha do excerto de código 7.1 define o vetor que possui todas as respostas à questão em análise. De seguida, na linha 3, é invocado o teste de *Shapiro-Wilk* para que seja possível obter o valor de *p-value*. Nesta questão, o *p-value* obteve um valor de 0.00647, sendo inferior ao nível de significância definido (0.05), rejeitando,

assim, a hipótese nula ($H_0$). Deste modo, é possível afirmar que a amostra não segue uma distribuição normal.

Tendo em conta o resultado obtido, é necessário aplicar, de seguida, um teste não paramétrico, nomeadamente o teste de *Wilcoxon* [71].

Como foi detalhado na secção 7.2, a hipótese definida em redor da satisfação prevê a obtenção de um grau de satisfação igual ou superior a 80%. Posto isto, é necessário converter a escala utilizada nos inquéritos (1 - 5). Assim, a satisfação deve ser superior a 3.99, uma vez que quatro equivale aos 80%. Definidos estes pontos, é possível avançar para o teste de *Wilcoxon*, onde as hipóteses definidas para este são as seguintes:

- ($H_0$) - A média das respostas obtidas é inferior ou igual a 3.99.
- ($H_1$) - A média das respostas obtidas é superior a 3.99.

O excerto de código 7.2 demonstra como o teste de *Wilcoxon* foi aplicado à questão em análise.

```
> wilcox.test(q1,mu=3.99,alternative="greater")

  Wilcoxon signed rank test with continuity correction

data:  q1
V = 15, p-value = 0.02667
alternative hypothesis: true location is greater than 3.99
```

Listing 7.2: Excerto de código do teste de *WilcoxonQ1* aplicado à questão 1 do Inquérito de Satisfação

De acordo com o resultado obtido no excerto de código 7.2, verifica-se que o valor de *p-value* é 0.02667, sendo inferior ao nível de significância, pelo que é permitido rejeitar a hipótese nula. Concluindo, é possível aferir, com um grau de confiança de 95%, que a concordância dos inquiridos na questão 1 é igual ou superior a 80%.

**Questão 2** - Todos as funcionalidades da aplicação funcionam da forma correta?

O vetor de respostas obtido para a questão 2 foi "q2 <- c(4,5,4,4,5)". Como foi explicado anteriormente, a primeira etapa é determinar se a amostra segue uma distribuição normal, utilizando o teste *Shapiro-Wilk*, retratado no excerto de código 7.3.

```
> q2 <- c(4,5,4,4,5)

> shapiro.test(q2)

  Shapiro-Wilk normality test

data:  q2
W = 0.68403, p-value = 0.00647
```

Listing 7.3: Excerto de código do teste de *Shapiro-Wilk* aplicado à questão 2 do Inquérito de Satisfação

Aplicando o teste de *Shapiro-Wilk* notou-se que não se verifica uma distribuição normal na amostra. Assim, é necessário aplicar o teste de *Wilcoxon*, com as mesmas hipóteses que foram definidas para a primeira questão.

```
> wilcox.test(q2,mu=3.99,alternative="greater")

  Wilcoxon signed rank test with continuity correction

data:  q2
V = 15, p-value = 0.02667
alternative hypothesis: true location is greater than 3.99
```

Listing 7.4: Excerto de código do teste de *Wilcoxon* aplicado à questão 2 do Inquérito de Satisfação

Como se verifica no excerto de código 7.4, o valor de *p-value* é 0.02667, sendo inferior ao nível de significância. Desta forma, rejeita-se a hipótese nula e confirma-se, com um grau de confiança de 95%, que a concordância dos inquiridos nesta questão é superior ou igual a 80%.

**Questão 3** - A aplicação facilita o processo de comunicação?

Nesta questão, todos os inquiridos forneceram a mesma resposta, pelo que o vetor obtido foi "q3 <- c(5,5,5,5,5)". Com base nestes dados, é possível concluir que esta amostra constitui uma distribuição discreta, com o valor de probabilidade 1. Uma vez que o objetivo é obter um nível de satisfação igual ou superior a 80% e todas as respostas equivalem ao nível máximo, é possível concluir que a satisfação dos inquiridos, relativamente a esta questão, atingiu a marca desejada.

**Questão 4** - Tendo em conta o publico alvo do sistema desenvolvido, a forma de como os inquéritos são apresentados (em formato de vídeo) é fundamental à realização dos mesmos?

A questão quatro recebeu exatamente as mesmas respostas da questão três, obtendo, também, os mesmos resultados. Desta maneira, é possível aferir que esta questão atingiu, também, um grau de satisfação igual ou superior a 80%.

**Questão 5** - Considera que o sistema traduz corretamente os seus gestos?

As respostas à última questão foram as mais díspares, obtendo-se o vetor de respostas: "q5 <- c(4,4,3,4,5)". Neste cenário, é necessário realizar o teste de *Shapiro-Wilk* novamente, como se ilustra no excerto de código 7.5.

```
> q5 <- c(4,4,3,4,5)

> shapiro.test(q5)

  Shapiro-Wilk normality test

data:  q5
W = 0.88349, p-value = 0.3254
```

Listing 7.5: Excerto de código do teste de *Shapiro-Wilk* aplicado à questão 5 do Inquérito de Satisfação

Observando os resultados obtidos no excerto de código 7.5, verifica-se que o valor do *p-value* é 0.3254, que por sua vez é superior ao valor de significância, não sendo possível rejeitar a hipótese nula.

Uma vez que a amostra atual segue uma distribuição normal, não é necessário aplicar o teste de *Wilcoxon*. Neste caso, é empregue o *t-test* [72]. As hipóteses definidas para este teste são as seguintes:

- ($H_0$) - A média das respostas obtidas é inferior ou igual a 3.99.
- ($H_1$) - A média das respostas obtidas é superior a 3.99.

O *t-test* foi aplicado tal como demonstrado no excerto de código 7.6.

```
> t.test(q5, mu = 3.99, alternative = "greater")

  One Sample t-test

data:  q5
t = 0.031623, df = 4, p-value = 0.4881
alternative hypothesis: true mean is greater than 3.99
95 percent confidence interval:
 3.325851      Inf
sample estimates:
mean of x
        4
```

Listing 7.6: Excerto de código do *t-test* aplicado à questão 5 do Inquérito de Satisfação

Como é possível comprovar com o excerto de código 7.6, o *p-value* obtido foi 0.4881, não sendo permitido rejeitar a hipótese nula. Deste modo, a questão cinco do inquérito de satisfação não possui um grau de satisfação superior ou igual a 80%.

### 7.5.3 Inquérito de Usabilidade

Tal como foi mencionado na secção 7.5.2, ambos os inquéritos foram respondidos pelos mesmos utilizadores, uma vez que o sistema os trata como um único inquérito global. Desta forma, considera-se que a amostra para o inquérito de usabilidade é, também, reduzida. No entanto, é relevante proceder à análise dos resultados obtidos.

O inquérito de usabilidade é composto por cinco questões de resposta fechada. Todas as respostas foram avaliadas e estudadas recorrendo ao RStudio, à semelhança do que foi feito para o inquérito de satisfação. Assume-se novamente um nível de significância de 0.05 e um grau de confiança de 95%.

A análise destas questões segue as mesmas etapas do estudo dos resultados do inquérito de satisfação, onde o primeiro passo é utilizar o teste de *Shapiro-Wilk* para determinar se a amostra segue uma distribuição normal. De seguida, conforme o resultado anterior, aplica-se um novo teste para verificar o grau de usabilidade obtido.

**Questão 1** - Considera que não precisaria de suporte técnico para conseguir utilizar o sistema?

O excerto de código 7.7 apresenta o teste de *Shapiro-Wilk* realizado para os dados da questão 1.

```
> q1 <- c(2,2,4,4,3)
>
> shapiro.test(q1)

  Shapiro-Wilk normality test

data:  q1
W = 0.82083, p-value = 0.1185
```

Listing 7.7: Excerto de código do teste de *Shapiro-Wilkd* aplicado à questão 1 do Inquérito de Usabilidade

A partir do excerto de código 7.7 nota-se que a amostra segue uma distribuição normal, uma vez que o valor de *p-value* é superior a 0.05. Deste modo, é necessário aplicar o *t-test*, conforme detalhado no excerto de código 7.8.

```
> t.test(q1, mu = 3.99, alternative = "greater")

  One Sample t-test

data:  q1
t = -2.2137, df = 4, p-value = 0.9544
alternative hypothesis: true mean is greater than 3.99
95 percent confidence interval:
 2.046609      Inf
sample estimates:
mean of x
        3
```

Listing 7.8: Excerto de código do *t-test* aplicado à questão 1 do Inquérito de Usabilidade

As hipóteses definidas são as mesmas para todas as seguintes questões e encontram-se detalhadas nos seguintes pontos:

- ($H_0$) - A média das respostas obtidas é inferior a 4.
- ($H_1$) - A médias das respostas obtidas é igual ou superior a 4.

Analisando o excerto de código 7.8, conclui-se que o valor obtido para o *p-value* foi 0.9544, que é superior ao nível de significância, não sendo possível rejeitar $H_0$. Assim, é possível concluir, com um grau de confiança de 95%, que a questão um do inquérito de usabilidade não obtém uma concordância igual ou superior a 80%.

**Questão 2** - Considera que não seria necessária uma formação para utilizar o sistema?

Na questão dois todos os inquiridos indicaram a mesma resposta, pelo que o vetor de respostas obtido foi "q2 <- c(2,2,2,2,2)". Tendo em conta estes dados, é possível determinar que esta amostra constitui uma distribuição discreta, com o valor de probabilidade 1. Visto que todas as respostas correspondem a um valor muito baixo, conclui-se que o grau de usabilidade é inferior a 80%.

**Questão 3** - Considera a experiência da interação com o sistema apelativa e imersiva?

O excerto de código 7.9 ilustra o teste executado para determinar a normalidade da distribuição da amostra das respostas recolhidas para a questão três.

```
> q3 <- c(5,4,5,5,4)

> shapiro.test(q3)

  Shapiro-Wilk normality test

data:  q3
W = 0.68403, p-value = 0.00647
```

Listing 7.9: Excerto de código do teste de *Shapiro-Wilkd* aplicado à questão 3 do Inquérito de Usabilidade

Após a utilização do teste de *Shapiro-Wilk*, é possível concluir que a amostra não segue uma distribuição normal, visto que o valor de *p-value* é inferior ao nível de significância. Desta forma, aplica-se o teste de *Wilcoxon*, empregando-se as mesmas hipóteses definidas para a questão um do inquérito de usabilidade.

```
> wilcox.test(q3,mu=3.99,alternative="greater")

  Wilcoxon signed rank test with continuity correction

data:  q3
V = 15, p-value = 0.02667
alternative hypothesis: true location is greater than 3.99
```

Listing 7.10: Excerto de código do teste de *Wilcoxon* aplicado à questão 3 do Inquérito de Usabilidade

O resultado deste teste indica que é possível rejeitar a hipótese nula, dado que o *p-value* é inferior a 0.05 (nível de significância). Conclui-se, deste modo, que a concordância dos utilizadores nesta questão é superior ou igual a 80%, com um grau de 95% de confiança.

**Questão 4** - Considera simples o processo de realizar uma tradução?

A questão quatro, tal como a questão dois, obteve as mesmas respostas pela parte de todos os inquiridos. No entanto, ao contrário do que se observou na segunda, esta questão obteve o valor máximo de pontos possível. Posto isto, é possível afirmar que o grau de usabilidade nesta questão é superior ou igual a 80%.

**Questão 5** - Considera simples o processo de preenchimento do inquérito?

O excerto de código 7.11 apresenta o teste executado para determinar a normalidade da distribuição da amostra relacionada com a presente questão.

```
> q5 <- c(4,5,5,5,5)

> shapiro.test(q5)

  Shapiro-Wilk normality test

data:  q5
W = 0.55218, p-value = 0.000131
```

Listing 7.11: Excerto de código do teste de *Shapiro-Wilkd* aplicado à questão 5 do Inquérito de Usabilidade

O teste demonstrou que a amostra não segue uma distribuição normal, visto que o valor de *p-value* é inferior ao nível de significância determinado. Assim, é necessário aplicar o teste de *Wilcoxon*, demonstrado no excerto de código **??**.

```
> wilcox.test(q5,mu=3.99,alternative="greater")

  Wilcoxon signed rank test with continuity correction

data:  q5
V = 15, p-value = 0.02386
alternative hypothesis: true location is greater than 3.99
```

Listing 7.12: Excerto de código do teste de *Wilcoxon* aplicado à questão 5 do Inquérito de Usabilidade

Explorando o excerto de código 7.12, verifica-se que o valor de *p-value* é 0.02386, sendo inferior ao nível de significância. É, então, possível afirmar, com um grau de confiança de 95%, que a concordância dos inquiridos nesta questão é superior a 80%.

### 7.5.4 Avaliação Global

Para terminar esta secção, é necessário avaliar o cumprimento das hipóteses definidas na secção 7.2, com base nos resultados recolhidos.

A primeira hipótese expressa o total cumprimento dos requisitos funcionais e não funcionais. Após a execução de todos os testes de *software* realizados, comprova-se que estes foram todos bem sucedidos. Esta informação, aliada à exposta no capítulo 6, é suficiente para comprovar a primeira hipótese.

A segunda hipótese encontra-se relacionada com o grau de usabilidade, calculado com recurso a um inquérito de usabilidade. Esta hipóteses visa a obtenção de um grau de usabilidade igual ou superior a 80%. As respostas ao inquérito, analisadas na secção 7.5.3, foram positivas, tendo alcançado um grau superior ou igual a 80% em três das cinco questões. Deste modo, como a maioria alcançou este resultado, ainda que por pouco, considera-se que a segunda hipótese foi, também, comprovada.

Por último, a terceira hipótese objetiva a obtenção de um grau de satisfação superior ou igual a 80%. Para avaliar esta métrica, foi utilizado um inquérito de satisfação. As respostas a este inquérito, estudadas na secção 7.5.2, foram bastante esclarecedoras, visto que quatro das cinco alcançaram um grau de satisfação superior ou igual a 80%.

Seguindo a mesma lógica da hipótese anterior, pode-se concluir que esta hipótese foi comprovada.

## 7.6 Sumário

Foram definidos dois identificadores de avaliação para classificar o projeto desenvolvido: o cumprimento dos requisitos funcionais e não funcionais; a usabilidade e a satisfação.

Para avaliar os identificadores eleitos foram constituídas três hipóteses distintas:

- Total cumprimento dos requisitos funcionais e não funcionais;
- Grau de usabilidade da aplicação com valor igual ou superior a 80%;
- Grau de satisfação dos utilizadores com valor igual ou superior a 80%.

Assim, para comprovar ou rejeitar estas hipóteses, foram realizados testes de *software* e, também, inquéritos de usabilidade e de satisfação. Os testes servem o propósito de comprovar o cumprimento dos requisitos funcionais e não funcionais, sendo que as respostas aos inquéritos, aplicadas em testes de hipóteses, possibilitam a avaliação dos graus de satisfação e de usabilidade.

No que toca aos resultados obtidos, todos os testes realizados passaram com sucesso, pelo que a primeira hipótese foi comprovada. Em relação à segunda hipótese, três das cinco questões do inquérito obtiveram um grau de satisfação superior ou igual a 80%. Uma vez que a maioria obteve esta marca, assume-se que esta hipótese também foi comprovada. Por último, no que toca ao grau de usabilidade, quatro das cinco perguntas do inquérito de usabilidade, alcançaram os valores pretendidos. Seguindo um raciocínio idêntico, afirma-se que a terceira hipótese foi comprovada. Deste modo, todas as hipóteses delineadas foram comprovadas.

# Capítulo 8

# Conclusão

Este capítulo retrata a conclusão final da presente dissertação. Primeiramente, analisam-se os objetivos alcançados. De seguida, delineiam-se as limitações existentes e, posteriormente, explora-se o trabalho futuro a desenvolver. Para terminar, apresenta-se a apreciação final do projeto.

## 8.1 Objetivos Alcançados

Neste projeto desenvolveu-se uma aplicação capaz de traduzir língua gestual portuguesa para língua portuguesa escrita, no contexto de saúde. Para que tal fosse possível, foi substancial seguir certos passos.

Inicialmente, foi identificado o problema existente na sociedade, sendo que, apesar de todos os avanços tecnológicos, a comunidade surda sente-se alienada de tal, ocorrendo o fenómeno denominado de *digital divide*. Foram, também, detalhados os objetivos para esta dissertação, servindo o propósito de fio condutor.

Para a construção desta solução, ou de qualquer outra, é necessária uma investigação prévia no que toca às soluções já existentes e às tecnologias que devem ser utilizadas. Esta pesquisa encontra-se pormenorizada no capítulo 2, "Estado de Arte". Através desta investigação, foi possível entender que, apesar de existir um número considerável de soluções, poucas são comerciáveis e muitas delas requerem equipamentos próprios, como luvas especiais, por exemplo.

Determinadas as tecnologias a utilizar, considerou-se pertinente realizar uma análise de valor do tema proposto. Este assunto foi abordado no capítulo 3, começando por aprofundar e depois identificar e analisar a oportunidade. Os resultados obtidos a partir do estudo, em conjunto com a proposta de valor, indicaram que o projeto proposto neste documento apresenta valor para o seu público alvo.

Terminando estas etapas, foi desenvolvida a análise de domínio e o *design* arquitetural da aplicação, bem como explicitadas as formas de a avaliar. Esta avaliação baseia-se não só em testes de *software*, mas também em questionários de usabilidade e de satisfação.

Com o objetivo de analisar a conclusão dos objetivos, procedeu-se à avaliação da solução, seguindo uma metodologia baseada em testes de *software*, testes de hipóteses e inquéritos de satisfação e usabilidade. Na presente secção, pretende-se retirar

conclusões sobre o estado final dos objetivos definidos na secção 1.3. Assim, os três objetivos definidos foram os seguintes:

- Desenvolver uma aplicação de fácil acesso, sendo possível utilizá-la tanto no computador como em telemóveis;
- A aplicação deve detetar gestos e traduzi-los corretamente para texto;
- Realizar traduções sem necessidade de equipamento especial, apenas a câmara do dispositivo tecnológico a ser utilizado.

No que toca ao primeiro objetivo traçado, como se verifica se secção 6.6, é possível utilizar a aplicação construída em qualquer dispositivo, desde que possua uma câmara. Este objetivo ainda contempla a acessibilidade, que em parte se deve inerentemente à disponibilidade em vários dispositivos, mas também à sua usabilidade. Esta última parte foi analisada na secção 7.5.3, com recurso às respostas fornecidas ao inquérito de usabilidade. Este inquérito é composto por cinco questões, sendo que três delas superaram o grau de usabilidade estabelecido de 80%. Posto isto, como a maioria das questões do inquérito alcançaram esta marca, aliado à possibilidade de utilizar a aplicação tanto em dispositivos móveis como em computadores, considera-se que o primeiro objetivo foi alcançado.

O sucesso do segundo objetivo apenas pode ser traçado com base na satisfação dos utilizadores, pelo que se faz uso das respostas ao inquérito de satisfação. Os resultados deste inquérito, demonstrados na secção 7.5.2, foram bastante positivos, dado que em cinco questões, quatro delas obtiveram um grau de satisfação igual ou superior ao estabelecido de 80%. Deste modo, conclui-se que os utilizadores se sentiram bastante satisfeitos a utilizar o sistema. Por outro lado, a questão que alcançou piores resultados foi a última, associada à precisão das traduções. Com base nesta informação, considera-se que o projeto evolui numa boa direção, apesar de se encontrar ainda num estado prematuro, não se considerando que este objetivo tenha sido completamente alcançado.

Por último, o terceiro objetivo pretende que as traduções sejam realizadas apenas com a câmara do dispositivo, o que é a realidade do presente projeto. Como foi explicado ao longo do capítulo 6, a deteção dos gestos efetuados é realizada com a *framework* MediaPipe, estudada na secção 2.7.1. Esta *framework* não necessita de qualquer tipo de equipamento extra e, a junção desta com a restante implementação, resultou num tradutor (ainda que um pouco longe da perfeição) capaz de cumprir este último objetivo.

## 8.2 Limitações

No global, a implementação do projeto foi bem sucedida, o que se comprova pela avaliação realizada e pelos objetivos alcançados. Ainda assim, identificaram-se algumas limitações, que podem vir a ser trabalhadas no futuro.

Duas das limitações prendem-se com o processo de tradução. A primeira é relativa à construção frásica. A verdade é que o sistema é capaz de realizar traduções e com bastante precisão e eficácia. Porém, estas traduções não constituem frases

gramaticalmente corretas, necessitando de conjugar verbos e adicionar pontuação consoante o caso. Este trabalho foi iniciado, no entanto, sem frutos, devido às restrições de tempo. Ainda no processo de tradução, a seguinte limitação é a espera de dois segundos necessária entre gestos. Num cenário ideal, todos os gestos deveriam ser efetuados de forma fluída, de modo a simular a comunicação entre indivíduos o melhor possível.

Para além do que já foi mencionado, o tamanho reduzido do *dataset* foi uma outra limitação. No momento de escrita deste documento, o sistema é capaz de realizar as traduções para os gestos: "Olá"; "Tudo bem"; "Ter"; "Dor"; "Cabeça"; "Dificuldade" e "Ver". Apesar dos dados recolhidos serem suficientes para demonstrar a qualidade do sistema desenvolvido, um *dataset* maior tornaria-o mais completo. Ainda assim, o facto da interface gráfica facilitar o aumento dos dados acaba por atenuar um pouco este ponto.

O número reduzido de respostas obtidas nos inquéritos de satisfação e usabilidade representa a última limitação detetada. Uma amostra maior forneceria resultados mais coesos e, acima de tudo, mais fiáveis.

## 8.3 Trabalho Futuro

O presente projeto, no âmbito de uma prova de conceito, alcançou imensos resultados positivos. No entanto, da forma que se encontra no presente, apresenta ainda inúmeras possibilidades de evolução, sendo as mais relevantes as seguintes:

- **Adicionar um modelo de correção gramatical** - como foi mencionado na secção 8.2, a inexistência de um modelo capaz de construir frases gramaticalmente corretas constitui uma limitação para o projeto. Deste modo, faz sentido que a adição deste modelo seja o principal ponto a trabalhar no futuro;
- **Traduzir língua portuguesa para LGP** - traduzir língua portuguesa para LGP nunca foi o objetivo deste projeto. Ainda assim, visto que o oposto já foi alcançado, possibilitar a tradução no sentido inverso iria aumentar ainda mais a integração da comunidade surda com o resto da sociedade;
- **Criar contas para utilizadores comuns** - neste momento, apenas os administradores se podem autenticar no sistema, porém, ao estender esta operação, uma infinidade de novas possibilidades surgem. Os utilizadores poderiam, por exemplo, gravar certas traduções e identificá-las como desejado para as utilizarem novamente no futuro, sem a necessidade de realizarem de novo os gestos;
- **Realizar e traduzir chamadas em vídeo** - os utilizadores poderiam realizar chamadas de vídeo com outras pessoas e os gestos efetuados seriam traduzidos em tempo real durante a conversa. O projeto foi desenhado de raiz para poder ser utilizado em dispositivos móveis e, esta nova funcionalidade, iria aumentar a participação dos indivíduos surdos na comunicação com os demais.

## 8.4 Apreciação Final

Terminando este projeto posso finalmente dizer que concluí cinco longos (e curtos) anos no ISEP. Ao longo do mestrado tive a confirmação de que enquanto quisermos, existe sempre algo novo para aprendermos, especialmente no mundo da informática. Mundo este que se encontra presente em todos os pilares da nossa comunidade e que permite ajudar e mudar por completo a vida de imensas pessoas. Por isso mesmo é que escolhi este projeto para a minha dissertação. Ao longo destes cinco anos, aprendi imenso e senti que finalmente podia construir algo que pudesse realmente ter impacto nas vidas dos outros e não hesitei.

Este projeto permitiu-me evoluir bastante, mas não apenas a nível técnico. É verdade que a dissertação possibilitou-me o contacto com novas tecnologias, mas também consegui aprender qual a realidade da comunidade surda, incluindo as dificuldades pelas quais os seus membros têm de passar diariamente. Acima de tudo, fiquei admirado e inspirado com a força de vontade destes indivíduos em superar os obstáculos presentes na sua vida. Para além de tudo isto, tive ainda a oportunidade de aprender um pouco de uma nova língua, a língua gestual portuguesa.

Por fim, sinto-me inteiramente satisfeito e também com orgulho no projeto desenvolvido. Pode ainda existir aspetos a melhorar, contudo, sinto que este projeto era bastante desafiante e consegui superar os maiores obstáculos que este me colocou. O sentimento de saber que estou a contribuir para melhorar a vida de outros é a maior recompensa que poderia querer.

# Bibliografia


[1] Alexa Kuenburg, Paul Fellinger e Johannes Fellinger. «Health Care Access Among Deaf People». Em: *Journal of Deaf Studies and Deaf Education* (2016), pp. 1–10. DOI: 10.1093/deafed/env042. URL: https://academic.oup.com/jdsde/article/21/1/1/2404217.

[2] Jan A G M Van Dijk. «Digital Divide: Impact of Access». Em: (2017). DOI: 10.1002/9781118783764.wbieme0043.

[3] *Deaf people face unique challenges as pandemic drags on - The Verge*. URL: https://www.theverge.com/22254591/deaf-communication-tech-access-coronavirus-isolation.

[4] Maria Fernanda Neves Silveira de Souza et al. «Principais dificuldades e obstáculos enfrentados pela comunidade surda no acesso à saúde: uma revisão integrativa de literatura». Em: *Revista CEFAC* 19 (3 jun. de 2017), pp. 395–405. ISSN: 1516-1846. DOI: 10.1590/1982-0216201719317116. URL: http://www.scielo.br/j/rcefac/a/Lr7dq73TcmLt3GSsxv3H75J/?lang=pt.

[5] Sign Community. *Introduction to Sign Language*. 2013. URL: https://www.signcommunity.org.uk/. (accessed: 30.12.2021).

[6] *GAF ::. Sítio Oficial do Gabinete de Atendimento à Família*. Nov. de 2020. URL: https://www.gaf.pt/pt/noticias/sabia-que-a-lingua-gestual-portuguesa-lgp-e-uma-das-3-linguas-oficiais-em-portugal. (accessed: 30.12.2021).

[7] Isavel Sofia Correia. *Morfologia Derivacional em Língua Gestual Portuguesa*. 2014. URL: https://www.porsinal.pt/index.php?ps=artigos&idt=artc&cat=9&idart=422. (accessed: 30.12.2021).

[8] Rafaela Teixeira. «Ação de Formação em Língua Gestual Portuguesa-Nível A1». Em: (2015).

[9] Maria Augusta Conde Amaral, Amândio Coutinho e Maria Raquel Delgado Martins. *Para Uma Gramática da Língua Gestual Portuguesa*. Caminho, 1994.

[10] *VIRTUALSIGN*. URL: http://193.136.60.223/virtualsign/pt/index.php#close. (accessed: 30.12.2021).

[11] Paula Escudeiro et al. «Real Time Bidirectional Translator of Portuguese Sign Language». Em: ().

[12] *Hand Talk App | Sign Language Translator in the palm of your hand*. URL: https://www.handtalk.me/en/app. (accessed: 30.12.2021).

[13] *We translate sign language. Automatically | SignAll Chat*. URL: https://www.signall.us/chat#learn-more. (accessed: 30.12.2021).

[14] Andrew Entwistle. «WHAT IS ARTIFICIAL INTELLIGENCE?» Em: *Engineering materials and design* 32.3 (1988). ISSN: 00138045. DOI: 10.7551/mitpress/12518.003.0004. URL: http://www-formal.stanford.edu/jmc/.

[15] A M Turing. «Computing machinery and intelligence». Em: *Machine Intelligence: Perspectives on the Computational Model*. Vol. 49. 2012, pp. 1–28. ISBN: 9781136525049. DOI: 10.1525/9780520318267-013.

[16] Forsyth Ponce et al. «Artificial Intelligence A Modern Approach Fourth Edition». Em: (2021). URL: https://lccn.loc.gov/2019047498.

[17] «What are the differences between a CV and a Resume ?» Em: (). URL: https://www.usgs.gov/faqs/what-are-differences-between-data-dataset-and-database.

[18] *What is Artificial Intelligence (AI)? | IBM*. URL: https://www.ibm.com/cloud/learn/what-is-artificial-intelligence (acedido em 26/01/2022).

[19] Batta Mahesh. «Machine Learning Algorithms-A Review Self Flowing Generator View project Machine Learning Algorithms». Em: *International Journal of Science and Research* (2018). ISSN: 2319-7064. DOI: 10.21275/ART20203995. URL: https://www.researchgate.net/publication/344717762.

[20] Thang Le Duc et al. «Machine learning methods for reliable resource provisioning in edge-cloud computing: A survey». Em: *ACM Computing Surveys* 52 (5 set. de 2019). ISSN: 15577341. DOI: 10.1145/3341145.

[21] IBM Cloud Education. *What are Neural Networks? | IBM*. 2020. URL: https://www.ibm.com/cloud/learn/neural-networks#toc-types-of-n-YgdI1-Kt%20https://www.ibm.com/cloud/learn/neural-networks.

[22] Sashikanth Arepalli. «Deep Learning». Em: (2016).

[23] IBM. *What are Convolutional Neural Networks? | IBM*. Out. de 2020. URL: https://www.ibm.com/cloud/learn/convolutional-neural-networks.

[24] *The overall architecture of the Convolutional Neural Network CNN includes an input.png (850×355)*. URL: https://www.researchgate.net/publication/331540139/figure/fig4/AS:733273504354306@1551837435967/The-overall-architecture-of-the-Convolutional-Neural-Network-CNN-includes-an-input.png.

[25] IBM. *What are Recurrent Neural Networks? | IBM*. Set. de 2020. URL: https://www.ibm.com/cloud/learn/recurrent-neural-networks#toc-what-are-r-btVB33l5.

[26] *TensorFlow*. URL: https://www.tensorflow.org/ (acedido em 27/01/2022).

[27] *Monitorar com TensorBoard - AI tools for Visual Studio | Microsoft Docs*. URL: https://docs.microsoft.com/pt-br/visualstudio/ai/monitor-tensorboard?view=vs-2017 (acedido em 27/01/2022).

[28] Qiang Ji. «Computer vision applications». Em: *Probabilistic Graphical Models for Computer Vision*. Academic Press, jan. de 2020, pp. 191–297. ISBN: 978-0-12-803467-5. DOI: 10.1016/b978-0-12-803467-5.00010-1.

[29] *ABOUT | OpenCV*. URL: https://opencv.org/about/%20http://opencv.org/about.html (acedido em 27/01/2022).

[30] Tiago Acioli. *Entenda o que é Merchandising*. 2016. URL: https://gaea.com.br/entenda-o-que-e-framework/%20http://manualdatecnologia.com/anuncios-mt/entenda-o-que-e-firmware/%20https://medium.com/publicitariossc/entenda-o-que-%7B%5C'%7Be%7D%7D-merchandising-86a93153428c (acedido em 27/01/2022).

[31] Google. *Home - mediapipe*. 2021. URL: https://google.github.io/mediapipe/ (acedido em 28/01/2022).

[32] *fingerpose: Documentation | Openbase*. URL: https://openbase.com/js/fingerpose/documentation (acedido em 28/01/2022).

[33] Sang Shin, Java Technology Architect e Sun Microsystems. «Disclaimer Acknowledgments». Em: *Agenda* (2006).

[34] TechTarget. *What is browser?* URL: https://whatis.techtarget.com/definition/browser (acedido em 28/01/2022).

[35] KC Karnes. *What are Progressive Web Apps and Are They Worth It? | CleverTap*. 2020. URL: https://www.outsystems.com/blog/posts/progressive-web-apps-pwa/%20https://web.dev/what-are-pwas/%20https://clevertap.com/blog/progressive-web-apps/ (acedido em 29/01/2022).

[36] *Single-page application vs. multiple-page application | by Neoteric | Medium*. URL: https://medium.com/@NeotericEU/single-page-application-vs-multiple-page-application-2591588efe58 (acedido em 28/01/2022).

[37] React. *React A JAVASCRIPT LIBRARY FOR BUILDING USER INTERFACES*. 2016. URL: https://reactjs.org/ (acedido em 29/01/2022).

[38] Microsoft. *.NET introduction and overview | Microsoft Docs*. 2020. URL: https://docs.microsoft.com/en-us/dotnet/core/introduction (acedido em 29/01/2022).

[39] *What is Flask Python - Python Tutorial*. 2010. URL: https://pythonbasics.org/what-is-flask-python/.

[40] *Spring Framework*. URL: https://spring.io/projects/spring-framework.

[41] *What Is MongoDB? | MongoDB*. URL: https://www.mongodb.com/what-is-mongodb.

[42] *What is SQL Server*. URL: https://www.sqlservertutorial.net/getting-started/what-is-sql-server/.

[43] *H2 Database Engine*. URL: https://www.h2database.com/html/main.html.

[44] Erick Evans. *Domain-Driven Design: Tackling Complexity in the Heart of Software*. Vol. 7873. 2003, pp. -529.

[45] Philippe Kruchten. «Architectural Blueprints-The "4+1"View Model of Software Architecture». Em: *IEEE Software* 12 (6 1995), pp. 42–50.

[46] Robert Blumen Salesforce e Sven Johann. *Editor: SOFTWARE ENGINEERING Software Architecture for Developers*. URL: www.se-radio.net..

[47] *O modelo C4 de documentação para Arquitetura de Software*. URL: https://www.infoq.com/br/articles/C4-architecture-model/.

[48] Paul Belliveau et al. *The PDMA ToolBook for New Product Development*. 2002.

[49] Peter A. Koen, Heidi M.J. Bertels e Elko Kleinschmidt. «Managing the front end of innovation-part I: Results from a three-year study». Em: *Research Technology Management* 57 (2 mar. de 2014), pp. 34–43. ISSN: 19300166. DOI: 10.5437/08956308X5702145.

[50] Peter A Koen. *Understanding the Front End: A Common Language and Structured Picture*. 2004. URL: www.frontendinnovation.com.

[51] Michael Baker e Susan Hart. «The Marketing Book, Sixth Edition». Em: (6 2008).

[52] *Google Trends*. URL: https://trends.google.pt/trends/?geo=PT.

[53] Thomas Saaty. *The Analytical Hierarchy process, planning, priority, Resource Allocation*. 1980, p. 287. ISBN: 0070543712.

[54] Paul Belleflamme e Nicolas Neysen. «DP A multisided value proposition canvas for digital platforms Paul Belleflamme and Nicolas Neysen 2 0 2 0 / 2 8». Em: (2020), pp. -20. URL: https://uclouvain.be/en/research-institutes/.

[55] *PII: S0377-2217(02)00178-9 | Elsevier Enhanced Reader*. URL: https://reader.elsevier.com/reader/sd/pii/S0377221702001789?token=EE37584086AD302851BC02A3EFF3BDB354E33263B2EFB1A7EAC13B71E4613294ED457F827ECF6E247DCD8A9918350863&originRegion=eu-west-1&originCreation=20220203235212.

[56] Ian Sommerville et al. «SOFTWARE ENGINEERING Ninth Edition». Em: (2011).

[57] *What does furps mean - Definition of furps - Word finder*. URL: https://findwords.info/term/furps.

[58] Bart De et al. «On the importance of the separation-of-concerns principle in secure software engineering». Em: (2002).

[59] *What is Frame. Frame in the world of animated video*. URL: https://darvideo.tv/dictionary/frame/.

[60] *What is an Array? Types of Array | Great Learning*. URL: https://www.mygreatlearning.com/blog/what-is-an-array-learn-more-in-one-read/.

[61] *NumPy: the absolute basics for beginners — NumPy v1.24.dev0 Manual*. URL: https://numpy.org/devdocs/user/absolute_beginners.html.

[62] *Auditar apps da Web com o Lighthouse | Tools for Web Developers | Google Developers*. URL: https://developers.google.com/web/tools/lighthouse/.

[63] *ISO 9241-11:2018(en), Ergonomics of human-system interaction — Part 11: Usability: Definitions and concepts*. URL: https://www.iso.org/obp/ui/#iso:std:iso:9241:-11:ed-2:v1:en.

[64] *What Customer Satisfaction Score (CSAT) IS and How It Can Help Understand Your Customers*. URL: https://www.cmswire.com/customer-experience/what-is-customer-satisfaction-score-csat/.

[65] *Testes Funcionais de Software*. URL: https://www.devmedia.com.br/testes-funcionais-de-software/23565#3.

[66] Ankur Joshi et al. «Article no.BJAST.2015.157 Opinion Article Joshi et al». Em: BJAST (4 2015), p. 157. ISSN: 2231-0843. DOI: 10.9734/BJAST/2015/14975.

[67] Assistant Secretary for Public Affairs. «System Usability Scale (SUS)». Em: (set. de 2013).

[68] *Teste de Hipóteses - Pro Educacional*. URL: https://proeducacional.com/ead/curso-cga-modulo-i/capitulos/capitulo-4/aulas/teste-de-hipoteses/.

[69] *Diferenças entre os testes paramétricos e os não paramétricos - Análise Estatística .PT*. URL: https://analise-estatistica.pt/2012/10/diferencas-entre-os-testes-parametricos-e-os-nao-parametricos.html.

[70] S. S. Shapiro e M. B. Wilk. «An Analysis of Variance Test for Normality (Complete Samples)». Em: *Biometrika* 52 (3/4 dez. de 1965), p. 591. ISSN: 00063444. DOI: 10.2307/2333709.

[71] Frank Wilcoxon. «Individual Comparisons by Ranking Methods». Em: *Biometrics Bulletin* 1 (6 dez. de 1945), p. 80. ISSN: 00994987. DOI: 10.2307/3001968.

[72] Amanda Ross e Victor L. Willson. «One-Sample T-Test». Em: *Basic and Advanced Statistical Tests* (2017), pp. 9–12. DOI: 10.1007/978-94-6351-086-8_2. URL: https://link.springer.com/chapter/10.1007/978-94-6351-086-8_2.

# Apêndice A

# Testes Funcionais

Neste Anexo, apresentam-se os restantes testes funcionais realizados. Assim, apresentam-se os testes funcionais nas tabelas A.1 e A.2.

Tabela A.1: Teste Funcional para a US01

| | |
|---|---|
| ***User Story*** | US01 - Como Administrador quero autenticar-me no sistema |
| **Procedimento** | O administrador inicia a aplicação. O sistema apresenta o formulário de *login*. O administrador indica o seu nome de utilizador bem como a respetiva palavra-chave e pressiona o botão "Login". |
| **Critérios** | 1. O sistema apresenta o formulário de autenticação ao iniciar a aplicação.<br>2. O sistema permite ao administrador introduzir o nome de utilizador e a palavra-chave.<br>3. Caso o administrador introduza dados incorretos, o sistema indica que os dados são inválidos.<br>4. Caso o administrador introduza os dados corretos, o sistema autentica o administrador e apresenta a página de tradução.<br>5. O administrador possui acesso à opção "Treinar o Sistema". |
| **Resultados** | 1. Passou<br>2. Passou<br>3. Passou<br>4. Passou<br>5. Passou |
| **Resultado Final** | Passou |

Tabela A.2: Teste Funcional para a US01

| | |
|---|---|
| ***User Story*** | US02 - Como Administrador quero sair do sistema |
| **Procedimento** | Depois do administrador se encontrar autenticado no sistema, pressiona o botão de *logout*. O sistema redireciona o utilizador para a página de autenticação. |
| **Critérios** | 1. O sistema apresenta o botão de *logout* em qualquer página depois de o utilizador se encontrar autenticado.<br>2. Ao pressionar o botão, o sistema redireciona o utilizador para a página de autenticação. |
| **Resultados** | 1. Passou<br>2. Passou |
| **Resultado Final** | Passou |