INSTITUTO SUPERIOR DE ENGENHARIA DO PORTO
MESTRADO EM ENGENHARIA INFORMÁTICA




Projeto AT-Home - Sistema de apoio a cuidadores

ROBERTO JORGE ROBALO SILVA
Outubro de 2018

POLITÉCNICO
DO PORTO

Instituto Superior de Engenharia do Porto

Projeto AT-Home

Sistema de Apoio a Cuidadores

Roberto Jorge Robalo Silva

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

Orientador:
Constantino Martins
Luiz Faria

Porto, Outubro de 2018

Aos meus Pais, Irmão, Família e Amigos

Resumo

Promover a atividade de exercício físico e ter uma alimentação equilibrada são pontos chave para ter uma vida equilibrada e poder evitar alguns problemas de saúde. Com o passar da idade vão aparecendo determinados problemas de saúde, normalmente tratados por profissionais especializados. Por outro lado, alguns dos problemas podem ser resolvidos sem a intervenção de profissionais de saúde, mas nem sempre as pessoas que necessitam auxílio conseguem os resolver, ou por dificuldades físicas ou mentais.

Na sociedade atual existem diversas Instituições Sociais que apoiam esse tipo de pessoas necessitadas. A missão dessas Instituições é o prestação de serviço e cuidados de saúde. Os cuidadores são aqueles que se encarregam de o fazer. Nessa ótica, os modos de funcionamento e tratamento devem ser controlados. Como nem todas as Instituições têm recursos suficientes para tal, uma solução informática que auxilie será uma mais-valia para a resolução desta problemática. É daí que surge o projeto AT-Home.

Nesta dissertação é proposta uma solução informática ao projeto AT-Home, sendo avaliadas diversas soluções existentes no mercado, comparando as vantagens e desvantagens de cada uma e promover a resolução do problema através de uma solução que vá de encontro às necessidades requeridas.

Palavras-chave: Instituições Sociais, Cuidados de Saúde, Pessoas Necessitadas, Cuidadores, Projeto AT-Home

Abstract

Promote physical activity and have a balanced diet are key points to have a stable life and can avoid some health problems. With increasing age, certain health problems will appear, usually treated by some specialized professionals. On the other hand, some of the problems can be solved without health professional's interventions, but not always the people who need assistance have the capacity to address them due to physical or mental difficulties.

In today's society there are many social institutions that support this kind of need. The mission of these institutions is the provision of health care and services. Caregivers are the ones who take care of it. From this point of view, the operation modes and treatment should be controlled. As not all institutions have sufficient resources to do so, a software solution that helps will be an asset to the resolution of this problem. That's where the AT-Home project arises.

In this dissertation a software solution is proposed to the AT-Home project, evaluating several existing solutions in the market, comparing the advantages and disadvantages of each one and promoting the resolution of the problem through a solution that meets the required needs.

Keywords: Social Institutions, Health care, Needy people, Caregivers, AT-Home Project

Índice

1 Introdução 16

1.1 Enquadramento e Motivação 16

1.2 Objetivo 17

1.3 Contribuição 19

1.4 Organização da Dissertação 19

2 Contexto e Estado de Arte 21

2.1 Contexto 21

2.2 Problema 23

2.3 Estado de Arte em Soluções/Abordagens Existentes 24

2.3.1 Penbase – Domi Connect 24

2.3.2 Greatcall - Healthcare 25

2.3.3 CareZone 26

2.3.4 AARP Caregiving 26

2.4 Avaliação de Critérios das Soluções/Abordagens Existentes 27

2.5 Estado de Arte em Tecnologia Relevante 28

2.6 Análise de Valor 31

2.6.1 Modelo NCD 32

2.6.2 Valor, Valor Para o Cliente e Valor Percecionado 34

2.6.3 Proposta de Valor 35

2.6.4 Modelo de Negócio CANVAS 36

3 Análise e Design 39

3.1 Visão Geral 40

3.2 Partes Interessadas 40

3.3 Actores 40

3.4 Requisitos 41

3.4.1 Requisitos Não Funcionais 41

3.4.2 Requisitos Funcionais 42

3.5 Vista de Cenário - Casos de Uso (UML) 44

3.6 Modelo de Domínio 47

3.7 Modelo Relacional 48

3.8 Vista Lógica (UML) 49

3.9 Vista Física (UML) 51

3.10 Vista de Processo (UML) 56

3.11 Vista de Implementação (UML) 57

4 Implementação 59

4.1 Metodologia 59

4.2 Tecnologias utilizadas 60
4.2.1 PHP 60
4.2.2 XAMPP 61
4.2.3 Laravel 61
4.2.4 Git e Bitbucket 64

4.3 Serviços implementados 64
4.3.1 Registo e acesso a ocorrências 65
4.3.2 Registo e acesso a intervenções e tipificação das mesmas 69
4.3.3 Registo e acesso à informação básica do paciente 72
4.3.4 Registo e acesso a tratamentos de saúde e tipificação das mesmas 76

4.4 Aplicação web *backoffice* 78

4.5 Padrões de *software* 84

5 Avaliação 92

5.1 Testes 92
5.1.1 Testes Unitários 92
5.1.2 Testes de Integração 99
5.1.3 Testes de Sistema 101
5.1.4 Testes de Aceitação 102
5.1.5 Testes de Usabilidade 103

5.2 Processo de Avaliação 103
5.2.1 Metodologia da Avaliação 104

6 Conclusões 107

6.1 Resultados do projeto 107

6.2 Objetivos realizados 108

6.3 Limitações e trabalho futuro 109

6.4 Espírito crítico 109

7 Referências 112

Anexo 1 – Modelo Relacional 115

Anexo 2 – Inquéritos de Satisfação 116

Lista de Figuras

Figura 1 – Modelo New Concept Development (NCD) - Fonte: (ResearchGate, 2018) 32
Figura 2 - Diagrama de Casos de Uso (UML) 45
Figura 3 - Modelo de Domínio (UML) 48
Figura 4 - Vista Lógica (UML) 50
Figura 5 - Vista Lógica (UML) - alternativa 1 50
Figura 6 - Vista Lógica (UML) - alternativa 2 51
Figura 7 - Vista Física (UML) 52
Figura 8 - Vista Física (UML) - alternativa 1 53
Figura 9 - Vista Física (UML) - alternativa 2 54
Figura 10 - Vista Física (UML) - alternativa 3 55
Figura 11 - Diagrama de Atividades - Registo de Tratamentos 57
Figura 12 - Estrutura de diretórios Laravel 62
Figura 13 - Diagrama de sequência acesso a ocorrências 65
Figura 14 - Diagrama de sequência criação de ocorrência 68
Figura 15 - Diagrama de sequência acesso a intervenções 70
Figura 16 - Diagrama de sequência criação de intervenção 71
Figura 17 - Diagrama de sequência acesso a informação básica do paciente 73
Figura 18 - Diagrama de sequência acesso a tratamentos de saúde 76
Figura 19 - Diagrama de sequência de registo de tratamento de saúde 77
Figura 20 - Menu principal da aplicação *backoffice* 79
Figura 21 - Menu de pacientes 79
Figura 22 - Configuração de tarefas para cuidadores 80
Figura 23 - Configuração de roupas 81
Figura 24 - Menu de tipificações 81
Figura 25 - Configuração de utilizadores 82
Figura 26 - Menu *dashboards* 83
Figura 27 - Permissões de menus 83
Figura 28 - Exemplo do padrão *Creator* 84
Figura 29 - Exemplo do padrão *Information Expert* 85
Figura 30 - Exemplo do padrão *Low Coupling* 86
Figura 31 - Exemplo do padrão *Controller* 87
Figura 32 - Exemplo do padrão *High Cohesion* 88
Figura 33 - Exemplo 1 do Padrão Polymorphism 89
Figura 34 - Exemplo 2 do padrão Polymorphism 90
Figura 35 - Exemplo do padrão *Pure Fabrication* 91
Figura 36 - Configuração base de dados de teste 93
Figura 37 - Associação base de dados de teste 93
Figura 38 - Configuração de caso de teste 94
Figura 39 - Exemplo de teste unitário de tipos de utilizador 94
Figura 40 - Classe *seeder* do tipo de utilizador 95

Figura 41 - Teste unitário tipo de utilizador 96
Figura 42 - Teste unitário à chamada registo de utilizador 97
Figura 43 - Teste unitário à chamada login 98
Figura 44 - Validação de testes unitários de login e registo de utilizador 99
Figura 45 – Dados de teste de integração 100
Figura 46 - Configuração de teste de integração 100
Figura 47 - Resultados de teste de integração ao login 101
Figura 48 - Teste de sistema 102

Lista de Tabelas

Tabela 1 - Avaliação dos principais critérios entre diferentes soluções 27
Tabela 2 - Modelo Canvas projeto AT-Home 38
Tabela 3 - Vantagens e desvantagens de base de dados não relacionais (Leavitt, 2010) 55
Tabela 4 - Testes de aceitação da aplicação móvel 103
Tabela 5 – Classificação das perguntas e respetiva escala para critérios de usabilidade..... 104
Tabela 6 – Classificação das perguntas e respetiva escala para critérios de fiabilidade 105
Tabela 7 – Classificação das perguntas e respetiva escala para critérios de adaptabilidade .. 106

Lista de Acrónimos

AHP	Processo hierárquico analítico *(Analytic Hierarchy Process*)
API	Interface de programação de aplicação (*Application Programming Interface)*
CRUD	Operações criação, leitura, atualização e eliminação *(Create, Read, Update, Delete*)
DPO	Agente de proteção de dados (*Data Protection Officer)*
HTTP	Protocolo de transferência de hipertexto (*Hypertext Transfer Protocol)*
IOT	Internet das coisas (*Internet Of Things)*
INEM	Instituto nacional de emergência médica
LARAVEL	Ferramenta de desenvolvimento para web, baseada em PHP
NCD	Modelo de conceção de desenvolvimento (*New Concept Development Model*)
PHP	Linguagem de programação para web (*Hypertext Preprocessor)*
QR Code	Código de barras com informação codificada (*Quick Response Code)*
REST	Transferência de estado representacional (*Representational State Transfer)*
RGPD	Regulamento geral sobre a proteção de dados
SGBD/DBMS	Sistema de gestão de base de dados (*DataBase Management System)*
UI	Interface com utilizador (*User Interface)*
UML	Linguagem de modelo unificada (*Unified Modeling Language*)

1 Introdução

Este capítulo tem como finalidade introduzir o tema do projeto AT-Home, como este surgiu e quais as suas motivações. Também é descrito os objetivos e contributos do trabalho a realizar e, por fim, a organização desta dissertação.

1.1 Enquadramento e Motivação

O problema de várias instituições sociais de apoio ao domicílio é a falta de ferramentas tecnológicas, que possibilitam o planeamento e registo dos seus funcionários (cuidadores). Muitas das vezes por esquecimento, outras das vezes por engano, os cuidadores das Instituições sabem quais os tipos de cuidados a realizar às pessoas necessitadas, nos seus domicílios, mas depois não registam os cuidados prestados. Normalmente, este tipo de instituições são organizações sem fins lucrativos, pelo que a perda de informação registada não é efetivamente monetária, mas sim à credibilidade das mesmas. É neste contexto que o projeto AT-Home se insere.

A motivação deste projeto é ajudar pessoas com necessidades especiais, para que estas possam ser tratadas de melhor forma possível. A possibilidade de um sistema de recomendação que proteja as pessoas consoante determinados critérios de saúde como, por exemplo, inconsciência ou a febre alta após longas horas pudesse contactar o INEM de uma emergência, foi uma ideia discutida. No entanto, após várias reuniões com o coordenador do projeto, considerou-se que as reais necessidades das instituições são o apoio na realização do trabalho dos cuidadores e, ao mesmo tempo, as próprias instituições monitorizarem as pessoas necessitadas.

Tendo em conta as necessidades referidas foi posta de parte a ideia de um sistema de recomendação, que seria mais benéfico para as pessoas necessitadas do que propriamente para as insituições. Apesar disso, a motivação na realização do projeto mantém-se.

1.2 Objetivo

O trabalho a realizar no âmbito desta dissertação tem como propósito desenvolver uma ferramenta de apoio a cuidadores, na prestação de cuidados de saúde e promover a qualidade de vida de pessoas idosas ou com necessidades especiais. O sistema visa auxiliar os cuidadores a proporcionar às pessoas uma melhor qualidade de vida e um envelhecimento ativo.

No âmbito desta problemática não existem muitas ferramentas que façam esse tipo de gestão, prevenção de riscos e cuidados considerados essenciais para melhorar o bem-estar dos idosos e a qualidade do apoio prestado e, assim sendo, o objetivo será desenvolver um sistema informático capaz de corresponder a estas atividades.

Assim sendo foram definidos os seguintes objetivos gerais:

- Desenvolver/contextualizar tecnologicamente o problema no âmbito do projeto AT-Home;
- Efetuar o levantamento e especificação dos requisitos a realizar;
- Estudar bibliografia referente ao tema do projeto;
- Identificar e estudar sistemas/ferramentas existentes no mercado;
- Elaborar proposta conceptual e de *design;*
- Identificar e descrever sucintamente outras alternativas concetuais consideradas;
- Justificar as decisões de *design*/construção tomadas;
- Desenvolver um protótipo funcional de acordo com a proposta conceptual elaborada;

- Especificar/Realizar os testes habituais no desenvolvimento de um sistema informático, nomeadamente testes de integração e de aceitação;

- Selecionar um subconjunto de futuros intervenientes do sistema, disponibilizar o protótipo desenvolvido e, consequentemente, avaliar a sua satisfação com o protótipo e o impacto do mesmo no âmbito do projeto AT-Home.

A solução informática deve contemplar os seguintes objetivos específicos:

- Facilitar a comunicação dos diferentes intervenientes;

- Gerir e controlar o trabalho de cada interveniente. Os cuidadores deverão registar o registo de ocorrências diárias e/ou intervenções realizadas e serem alertados pelos profissionais de saúde acerca de possíveis episódios de doença ou o estado de cada paciente. Já os profissionais de saúde deverão ser responsáveis por obter informação das ocorrências diárias dos cuidadores, registar cuidados especializados e monitorizar os pacientes. Por fim, e não menos importante, os responsáveis da Instituição, que se podem dividir em dois intervenientes distintos: coordenadores e administradores. Os coordenadores deverão registar e gerir as tarefas para cada cuidador e controlar a atividade diária de cada um. Os administradores são responsáveis pela gestão dos utilizadores e verificam dados estatísticos do trabalho realizado;

- Gerir o controlo de acesso aos diferentes tipos de informação associada aos pacientes. Para obtenção dos dados dos pacientes será necessário implementar, na residência de cada um, um método de validação da presença do cuidador no local, com a utilização da tecnologia *QR Code*. Quando os profissionais de saúde se encontrarem no mesmo local deverão ter acesso à informação sensível de cada paciente;

- Generalizar os processos. No futuro, o sistema irá ser utilizado em diferentes países, pelo que, deverá ser o abrangente em determinadas situações devido à diferença de culturas.

Por outro lado, a aplicação deve ter as seguintes restrições:

- Modo de funcionamento: deverá funcionar em modo *online* e *offline,* ou seja, independente da ter ou não ligação à internet. Quando se encontrarem na habitação

de algum paciente, os cuidadores ou profissionais de saúde poderão não ter acesso à internet, devido à região ou local de difícil acesso. Logo, a informação dos pacientes deverá ser previamente carregada na aplicação, previsivelmente todos os dias quando chegam à Instituição antes de visitarem cada paciente;

- Controlo de acesso a determinados conteúdos.

1.3 Contribuição

De forma a contribuir a realização deste projeto, com este documento pretende-se demonstrar como:

- É interpretado o problema a resolver, analisando o seu contexto e objetivos, adotando boas práticas de engenharia informática;
- São avaliadas diferentes abordagens para a sua resolução, através de um estado de arte, onde se referem as diversas ferramentas existentes no mercado, as suas semelhanças e uma avaliação das mesmas;
- É projetado um desenho e construção de uma solução adequada, aplicando ciências e boas práticas de engenharia informática;
- São avaliados os resultados e respetivas conclusões.

1.4 Organização da Dissertação

Ao longo deste documento serão descritas várias etapas para a resolução de uma solução preconizada. Para descrever a solução foi seguida uma estrutura que está dividida em sete capítulos. Este primeiro capítulo contém um enunciado breve do tema, os objetivos e principais contributos a serem realizados.

O capítulo 2 inclui, com mais detalhe, o contexto do projeto AT-Home e o levantamento do estado da arte das abordagens/soluções existentes, através de uma investigação aprofundada do tema e recolha de informação relevante. É, também realizado uma avaliação às soluções, sendo efetuada uma comparação entre os requisitos de cada uma e onde a tecnologia

relevante ao tema é abordada e discutida. Para concluir o capítulo é proposto uma análise de valor à solução.

A conceção e o *design* de uma possível solução à realização do projeto encontra-se no capítulo 3, bem como as justificações da escolha, com o auxílio na apresentação de alternativas.

No capítulo 4 apresenta-se a implementação da solução, com principal foco nas tecnologias utilizadas e a sua utilidade no desenvolvimento da mesma. É também projetado e detetado alguns padrões de *software*, que foram aplicados à solução.

O capítulo 5 exibe uma avaliação à solução realizada, com as metodologias utilizadas para satisfazer os requisitos da solução, bem como os respetivos testes.

O capítulo 6 são as conclusões, onde se realiza o balanço final do trabalho, realçando os aspetos principais do que foi realizado, formulando juízos críticos (positivos e negativos). Como neste caso concreto ainda não se completou o desenvolvimento da solução, as conclusões serão uma projeção daquilo que se estará à espera.

Por fim, o capítulo 7 encontram-se as referências que suportam o trabalho realizado.

2 Contexto e Estado de Arte

Neste capítulo são apresentados, de forma mais detalhada, o contexto do problema e a sua génese, com o auxílio de estudos e investigações relacionadas. De seguida, é realizado o estado de arte, através da pesquisa de soluções/abordagens existentes no mercado e é proposto uma avaliação das mesmas, recorrendo a metodologias de avaliação. Nesta secção tecnologia relevante é apresentada. E, por fim, é apresentada uma análise de valor, onde é analisado o modelo NCD (*New Concept Development Model)*, valor, valor para o cliente e valor percecionado, proposta de valor e o modelo Canvas.

2.1 Contexto

É do senso comum de que todas as pessoas necessitam de uma alimentação equilibrada e de cuidados de saúde apropriados para que possam ter uma vida saudável. Ao longo da vida é necessário tomar precauções ao nível de saúde, como, por exemplo, ter a vacinação em dia, assegurar o controlo de tensões e de glicemia de forma a precaver problemas graves no futuro. A prevenção de doenças consideradas como flagelo da sociedade atual, tais como acidente vascular cerebral (AVC), ataques cardíacos, diabetes e estados depressivos, constitui um objetivo primário e urgente. O estudo "*A Population-Based Study of 13 000 Men and Women With 20 Years of Follow-up*", realizado em 2004, refere que os "pacientes com diabetes […] aumentam a mortalidade de doenças cardiovasculares" (Almdal, T., Scharling, H., Jensen, J., Vestergaard, H., 2004).

Por outro lado, a esperança média de vida tem vindo a aumentar. Através de um artigo publicado na revista "*Social Science & Medicine*", onde são realizadas pesquisas no âmbito de Ciências Sociais em Saúde, é descrito que: "determinadas pesquisas realizadas indicam que condições políticas influenciaram o aumento da expetativa de vida". Os principais fatores desta conclusão são a "recuperação rápida no controlo de doenças infeciosas entre 1920 e 1960 e que se estendeu, nas décadas seguintes, a doenças cardiovasculares, cancro da mama, acidentes de veículos motorizados e outras" (Mackenbach, 2013).

O envelhecimento ativo da população mundial constitui um eixo prioritário de intervenção da União Europeia nos últimos anos: "[…] societies have to […] improve the robustness of health, long-term care, and welfare systems in Europe and to help people to stay healthy and active in old age […] include prevention and health promotion, better self-care, increased coordination of care […]" (Rechel, B., Grundy, E., Robine, Jean-Marie, Cylus, J., Mackenbach, J.P., Knai, C., McKee, M., 2013). O foco tem sido o melhoramento do sistema de saúde, os cuidados de longa duração e na ajuda das pessoas a permanecerem saudáveis e ativas na velhice. E Portugal não foge à regra. Segundo Maria João V. Rosa, através do seu livro "O Envelhecimento da Sociedade Portuguesa" refere que: "Portugal […] hoje é um dos países mais envelhecidos do espaço europeu". A autora chegou a esta conclusão através do estudo, e formou a seguinte opinião: "Os agentes políticos têm querido reagir ao progressivo envelhecimento da população adotado medidas diversas, como incentivo à natalidade […] mas a eficácia das mesmas têm-se relevado duvidosa". Apesar disso, segundo a autora, houve várias iniciativas importantes como: "o aumento da idade da reforma em função da esperança de vida […] ou a abertura de mais serviços de apoio a pessoas idosas em situações de doença ou de outros estados de dependência prolongada" (Rosa, 2016).

Para garantir uma maior oferta de serviços existem várias instituições sociais de apoio ao domicílio que prestam cuidados e auxiliam as famílias, que por impedimento não conseguem assegurar o bem-estar dos membros mais idosos. A instituição portuguesa ODPS (Obra Diocesana de Promoção Social) é uma delas. É uma instituição particular de solidariedade social, com sede no Porto, que tem como objetivo "proporcionar melhoria de vida, nas suas principais latitudes, a muitas pessoas desfavorecidas: bebés, crianças, jovens e adultos em escalão de terceira idade, muitos dos quais encontrando-se completamente dependentes". (Ribeiro, 2018) Um dos principais serviços da ODPS é o serviço de apoio domiciliário, cujo objetivo é "contribuir para a manutenção do cliente no seu meio sociofamiliar, prestando

cuidados individualizados e personalizados no domicílio a indivíduos e famílias quando por motivo de doença ou outro impedimento não possam executar as suas atividades de vida diária". (ODPS, 2018). Este serviço é "uma resposta social da Obra Diocesana que serve a terceira idade do Porto" (ODPS, 2018).

Para além da ODPS existem outras organizações internacionais e parceiros também interessados neste projeto. De momento, o coordenador do projeto é o intermediário com estas parcerias, porque ele pretende tornar as aplicações mais comerciais.

2.2 Problema

Os responsáveis da ODPS desejam gerir de melhor forma o fluxo de informação de todo o seu trabalho. No caso concreto de serviço ao domicílio, os ajudantes da Instituição, também conhecidos por cuidadores, têm um papel fundamental na realização de determinadas tarefas nas residências dos clientes. As principais funções dos cuidadores são a prestação de cuidados de higiene e conforto, fornecimento de refeições, tratamento de roupas, transporte, entre outros. No final de cada visita, os cuidadores realizam o registo de Serviços Contratualizados efetuados a cada paciente, através do preenchimento de uma ficha denominada por "Processo de Desenvolvimento Individual – Planificação Individualizada de Serviços". Caso tenham realizado o tratamento de roupas, os cuidadores devem preencher outra ficha, com o nome de "Lavagem de Roupa", onde indicam os artigos de roupa e o estado dos mesmos. Por outro lado, caso um paciente não esteja no seu estado normal ou necessite de algum tratamento especializado, os cuidadores devem informar um dos responsáveis da instituição para que um profissional de saúde vá ao seu encontro. Os profissionais de saúde, que podem ser enfermeiros, psicólogos, entre outros, são técnicos especializados, aptos a realizar determinadas intervenções aos clientes. Ao realizarem um tipo de intervenção devem completar o registo de Serviços Contratualizados do paciente, indicando o que foi realizado.

Por vezes, os cuidadores esquecem-se de registar algumas intervenções e que acaba por ser um problema, porque não há uma gestão efetiva quando há falta de informação. Por outro lado, a comunicação entre os diferentes intervenientes - cuidadores, responsáveis da instituição e profissionais de saúde - não é de todo eficaz, pelo que é fundamental registar todas as intervenções.

Com o intuito de auxiliar a prestação destes serviços foi identificada a necessidade de uma solução informática capaz de fazer a gestão destes processos.

2.3 Estado de Arte em Soluções/Abordagens Existentes

Através deste capítulo pretende-se analisar diferentes soluções / ferramentas que existem no mercado. Existem algumas que têm semelhanças nas funcionalidades propostas e outras que podem contribuir, de outra forma, para a realização da solução.

De forma a obter informação acerca das diferentes soluções existentes foram pesquisadas palavras chave relacionadas com o tema, tais como cuidador de idosos, tratamentos médicos, instituições sociais de apoio a idosos, entre outras. As fontes de pesquisa utilizadas foram essencialmente em bibliotecas digitais, como ACM Digital library e Google Scholar e com o auxílio da aplicação Zotero para obter, de uma forma mais organizada, as referências bibliográficas obtidas.

2.3.1 Penbase – Domi Connect

Penbase - Domi Connect é uma aplicação móvel que fornece informação aos cuidadores e profissionais externos. Tem como finalidade completar ou substituir os registos em papel que são entregues no destinatário que realizou tratamento. (Penbase.com, 2018)

É uma solução que abrange três módulos: (Penbase.com, 2018)

- Aplicação móvel para familiares;
- Aplicação web para completar a informação registada na aplicação móvel;
- Guia suplementar interligada a outra aplicação da Penbase, denominada Dalyo Domi.

Tem como principais objetivos: (Penbase.com, 2018)

- Automatizar o envio de informações aos familiares do paciente;
- Digitalização do livro de registo doméstico;

- Partilha de informações com familiares, cuidadores, independentemente da sua localização;
- Contacto direto com os familiares, utilizando um sistema de mensagens de fácil utilização.

Esta aplicação é aquela que mais se aproxima das necessidades do projeto AT-Home. Contém uma boa parte dos requisitos definidos, mas não abrange a diversidade das diferentes culturas, nem contempla um sistema de recomendação, nem é gratuita.

2.3.2 Greatcall - Healthcare

Greatcall é um conjunto de soluções de segurança e saúde para adultos e idosos, que premeia o atendimento ao cliente e ajuda os consumidores mais envelhecidos a serem mais independentes e usufruir das suas vidas. (Greatcall, 2018)

Greatcall – Healtcare é um serviço de suporte a idosos que faz a monitorização de pessoas necessitadas, com custos reduzidos, através de um dispositivo na residência e de um dispositivo móvel. (Healthcare Greatcall, 2018).

Os objetivos deste serviço são:

- Ajudar na redução dos custos mensais;
- Melhorar os resultados de saúde para pessoas em risco;
- Promover o crescimento através da inovação;
- Assistir e atuar perante emergências.

Esta aplicação tem o benefício de oferecer a monitorização de pessoas necessitadas, mas não contempla a maior parte dos requisitos necessários do projeto AT-Home, nomeadamente o registo de ocorrências e intervenções dos cuidadores.

2.3.3 CareZone

CareZone é um aplicativo que organiza a informação em ficheiros, contactos e medicações das pessoas necessitadas. Também garante uma comunicação, coordenação e sincronismo nos dados registados pelas famílias, cuidadores e médicos, utilizando um diário. (CareZone, 2018)

A aplicação tem como objetivos:

- Criar facilmente uma lista completa de medicamentos;
- Obter sempre a lista de medicamentos disponíveis a tomar;
- Imprimir e partilhar o agendamento de medicação;
- Notificação para não haver esquecimento na tomada dos medicamentos.

A aplicação CareZone tem uma forte componente na comunicação dos diferentes intervenientes e organização de toda a informação, o que ajuda imenso na agilização ao nível processual. A incorporação destas características na solução a desenvolver poderá permitir satisfazer os requisitos identificados.

2.3.4 AARP Caregiving

AARP Caregiving oferece um conjunto de ferramentas web, que ajudam a encontrar informações de forma rápida e fácil, para atender a uma variedade de necessidades relacionadas com os cuidados familiares. (AARP, 2018)

Tem como principais focos:

- Apoiar um familiar a assumir o papel de cuidador;
- Personalizar o tipo de tratamento consoante o tipo de doença do paciente;
- Ter acesso à comunidade AARP para obter respostas a perguntas realizadas;
- Acesso a questionários de aprendizagem.

Este conjunto de ferramentas é subdividida em várias aplicações: móvel, web e tablet, pelo que existe uma sincronização permanente da informação.

A solução AT-Home poderá seguir uma abordagem semelhante no foco de acesso a questionários de aprendizagem e, utilizando a abordagem da comunidade existente, como sistema de recomendação.

2.4 Avaliação de Critérios das Soluções/Abordagens Existentes

Para que sejam tomadas as melhores decisões, com este capítulo pretende-se que se avalie, sucintamente, as soluções abordadas na secção 2.3, referindo as vantagens e desvantagens de cada uma e como estas podem contribuir na realização do projeto AT-Home.

Como nem todas as soluções são gratuitas, o critério seguido baseou-se na identificação das funcionalidades que foram consultadas em artigos, páginas web e nas plataformas de distribuição para dispositivos móveis (lojas), onde estas podem ser descarregadas.

Os critérios apresentados são aqueles que vão de encontro aos requisitos pretendidos no projeto AT-Home.

Na tabela 1 é apresentada uma avaliação, ao nível dos principais critérios definidos, para a solução pretendida, em relação às soluções existentes. Os valores da avaliação são um "Sim", como funcionalidade aplicável e um "Não", como não aplicável.

Tabela 1 - Avaliação dos principais critérios entre diferentes soluções

Soluções / **Critérios**	**Penbase - Domi Connect**	**Greatcall - Healthcare**	**CareZone**	**AARP Caregiving**
Aplicação web	Sim	Não	Não	Sim
Aplicação móvel Android ou iOS	Sim	Sim	Sim	Sim
Modo *offline*	Não	Não	Não	Não
Registo de ocorrências e intervenções diárias	Sim	Não	Sim	Sim
Registo médico	Sim	Sim	Sim	Sim
Obter informação sensível de pacientes	Não	Não	Não	Não
Monitorização de pacientes	Sim	Sim	Sim	Sim
Comunicação	Sim	Sim	Sim	Sim

entre intervenientes				
Personalização para diferentes culturas	Não	Não	Não	Sim
Multilíngua	Sim	Não	Não	Sim
Sistema de recomendação	Não	Não	Não	Sim
Proteção Dados Maio 2018	Não	Não	Não	Não

Tal como se pode verificar, existem muitas semelhanças e é possível conseguir conciliar estas diferentes soluções, adicionando funcionalidades às mesmas de forma a contemplar todos os requisitos requeridos. No entanto, tendo em conta a generalidade de processos para diferentes países, distintas culturas e na opinião das Instituições interessadas, não há uma solução integral que satisfaça completamente a ideia pretendida, pelo que é proposta uma nova aplicação.

2.5 Estado de Arte em Tecnologia Relevante

Cada vez mais, a tecnologia faz parte no dia a dia de qualquer ser humano. Utilizamos diversos equipamentos que nos ajudam a resolver determinados problemas. Existe uma parte benéfica no seu uso, mas também pode trazer prejuízo, pelo que a sua utilização deve ser equilibrada.

No caso concreto da tecnologia que dá suporte a cuidadores e/ou profissionais de saúde, esta pode trazer grandes benefícios, tal como Helen Hoenig, Donald Taylor e Frank A. Solan referem no seu caso de estudo: "[…] *technological assistance or personal assistance might be used only some of the time, and use of technological assistance or personal assistance for certain tasks might vary over time*". (Hoenig, Helen and Taylor, Donald and Solan, Frank A., 2003).

De uma maneira geral, a tecnologia que assiste esta problemática veio apoiar o modo de trabalho dos assistentes pessoais e reduzir o tempo de assistência: "[…] *the potential for appropriate equipment to reduce use of personal assistance is apparent…*", como se pode comprovar no resultado conclusivo do estudo: "*Among people with disability, use of assistive technology was associated with use of fewer hours of personal assistance.*" (Hoenig, Helen and Taylor, Donald and Solan, Frank A., 2003).

Por outro lado, a tendência de pesquisa, no âmbito do estudo dos cuidados de saúde na IOT (*Internet Of Things*), rede de partilha e recolha de informação entre dispositivos, tem vindo a melhorar na qualidade de cuidados médicos, pois ajuda no cuidado preventivo e reduz o risco de erro humano. (Kadarina, T. M. and Priambodo, R., 2017). O foco principal deste estudo é reduzir a taxa de mortalidade materna e infantil na Indonésia. Através de dispositivos sensoriais que detetam a condição do paciente, são enviadas informações para a rede que, posteriormente, poderem serem acedidas numa aplicação móvel pelos cuidadores, pacientes e médicos. Uma abordagem semelhante a este estudo é a solução Greatcall – Healthcare, referida na secção 2.3.2.

No enquadramento do projeto AT-Home foi considerada a aplicação do Regulamento Geral de Proteção de Dados, também conhecido por RGPD, que "*vem proporcionar maior controlo sobre os dados pessoais para os cidadãos europeus […]"* (Cordeiro, S., Gouveia, L. B., 2018). Para entender melhor esta problemática, o autor deste documento assistiu a várias palestras sobre o tema, nomeadamente algumas formações ao nível empresarial, em contexto laboral e a Conferência / Encontro Internacional das Transformações do Direito do Trabalho Ibérico, no dia 27 de Abril de 2018, no Tribunal da Relação do Porto. Esta Conferência juntou diferentes intervenientes especializados na área, como Diretores de Serviço de Apoio Jurídico da Universidade do Porto, Professores Universitários das Faculdades de Direito da Universidade do Porto e da Universidade Rey Juan Carlos de Madrid. O principal tema abordado foi a aplicação do Regulamento Geral de Proteção de Dados em organizações, que entrou em vigor em 25 de Maio de 2018 e que substituiu a antiga lei de proteção de dados.

Os objetivos propostos na Conferência referida foram a maior transparência da informação para quem tem direito à mesma, de forma prévia e precisa, e a proporcionalidade da sua aplicação. Esta proporcionalidade pode variar consoante os mecanismos de proteção existentes ou não existentes nas organizações, como, por exemplo, Base de Dados ou documentos pessoais dos colaboradores, pelo que é sugerido que à medida que se necessite de informação cada vez mais sensível, a sua proteção ao acesso seja cada vez maior.

Por outro lado, a proteção de dados dos clientes é importante e deve responder a quatro questões fundamentais:

- Quem tem acesso à informação?

- Quando têm acesso à informação?
- Como têm acesso à informação?
- Onde têm acesso à informação?

De forma a responder às questões anteriores devem ser realizados mecanismos de prevenção e proteção ao nível de *hardware* e de *software* como, por exemplo, acessos privados a recursos físicos por autenticações biométricas e/ou password, recorrer a políticas de privacidade e permitir ao cliente descarregar toda a sua informação num local seguro.

Data Protection Officer (DPO) foi um conceito também abordado na Conferência acima mencionada pois, representa o Encarregado de Proteção de Dados Empresarial e é essencial por ser o responsável pela gestão dos dados de uma organização. Um DPO deve possuir três características principais: Imparcialidade, Independência e Dinamismo. Os deveres de um DPO consistem em:

- Acompanhar a evolução do tema e o seu desenvolvimento;
- Ter dinamismo na aplicação das metodologias do RGPD, devendo investigá-las e aplicá-las em conformidade, isto é, há um regulamento, mas a aplicação das mesmas não se cinge a um conjunto de regras ou tópicos a seguir;
- Deverá trabalhar em conjunto com os colaboradores diretamente ligados aos dados, nomeadamente Recursos Humanos, Administrativos, Administradores de Bases de Dados, Programadores e também os Diretores Executivos, para que estes sejam sensibilizados para a importância do tema.

Em 2010, na Conferência D8 sobre privacidade, Steve Jobs já dizia que "a privacidade dos dados depende sempre do utilizador […] o utilizador pode ou não consentir o envio da sua informação pessoal para a *cloud* […] o utilizador deve ser informado da futura utilização dos seus dados" (Yurieff, 2018). O tema da privacidade já tinha sido levantado previamente nos Estados Unidos da América, mas apenas este ano a União Europeia aprovou com uma Diretiva de Proteção de Dados (RGPD) para ser aplicada nos países europeus.

Um dos objetivos principais do projeto AT-Home é o cumprimento do RGPD, de forma a garantir que quem acede a essa informação esteja devidamente autenticado e autorizado. Para isso foram realizadas diversas validações, verificações e/ou proteções que irão ser explicitadas no capítulo 5 (Implementação).

Em suma, para os assistentes sociais o importante é agilizar as visitas aos pacientes e, ao mesmo tempo garantir que a informação seja efetivamente salvaguardada de forma segura, pelo que a tecnologia é fundamental nesses pontos.

2.6 Análise de Valor

A Análise de Valor é um processo de estudo do produto, onde é referenciado o seu valor para o cliente, tentando obter um mínimo custo.

"Value analysis is a method that provides the means for industrial disciplines to cross departmental lines in a joint study of a product's function in relation to its cost." (National Economic and Development Authority, 2009)

"The creation of value is key to any business, and any business activity is about exchanging some tangible and/or intangible good or service and having its value accepted and rewarded by customers or clients, either inside the enterprise or collaborative network or outside." (Nicola, Susana and Ferreira, Eduarda Pinto and Ferreira, J. J. Pinto, 2012)

"*The primary objective of value analysis is assess how to increase the value of an item or service at the lowest cost without sacrificing quality*." (Nicola, Susana, 2017).

É objetivo deste trabalho descrever e analisar a geração do valor do projeto AT-Home para o cliente, tendo como mais-valia a diversidade de alguns processos consoante o país a ser utilizado. As instituições diferem no seu modo de funcionamento e não existe uma aplicação no contexto atual no mercado que consiga abranger as diferentes partes.

2.6.1 Modelo NCD

O modelo NCD (*New Concept Development Model*) é utilizado para representar a primeira de três fases do processo de inovação, que também pode ser chamado por *Fuzzy Front End*. As outras duas fases são o desenvolvimento e a comercialização do produto.

O modelo NCD "*Provides a common language and terminology necessary to optimize the "Front End of Innovation""*. (Koen, 2004)

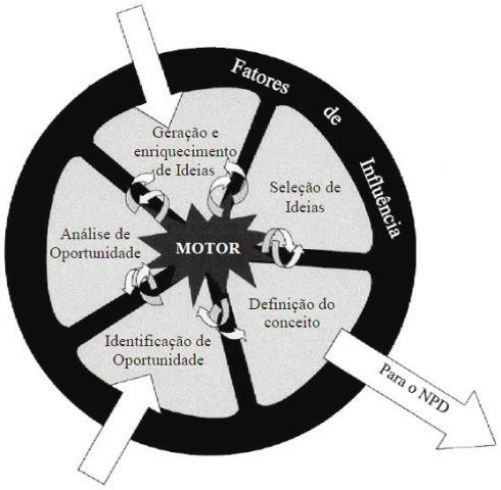


Figura 1 – Modelo New Concept Development (NCD) - Fonte: (ResearchGate, 2018)

O modelo é composto por três componentes: o Motor, os Fatores de Influência e as Cinco Atividades. O movimento circular do modelo corresponde à forma de como os componentes estão interligados e as suas dependências. O Motor é o "gestor" dos Cinco Elementos, que representa a estratégia de liderança, cultura e negócios de uma organização. Os Fatores de Influência são tudo aquilo que não é "controlável" dentro e fora de uma organização, tais como, o contexto e o ambiente em que uma organização está inserida, a sociedade, fatores políticos, legislativos, entre outros. Por fim, as Cinco Atividades que aumentam o valor do produto e a sua comercialização são a Identificação e Análise de Oportunidade, Geração e Seleção de Ideias e Desenvolvimento/Definição de Conceito.

Com o auxílio deste modelo, a análise do projeto AT-Home segue a mesma estrutura.

Identificação de Oportunidade

A oportunidade do projeto AT-Home surge da necessidade de algumas instituições de cuidadores de saúde portuguesas e estrangeiras que prestam auxílios a pessoas necessitadas garantirem:

- Apoio no registo de intervenções e tratamentos aos utentes;
- Facilidade de comunicação entre cuidadores e profissionais de saúde;
- Auxílio a gestão de cuidados de saúde prestados;
- Monitorização de pacientes;
- Gestão de tarefas para cuidadores.

Análise de Oportunidade

Através de uma análise de mercado não foi possível encontrar uma ferramenta gratuita que vá de encontro às totais necessidades do cliente, mas foram encontradas outras ferramentas que disponibilizam algumas funcionalidades.

Geração de Ideias

Tendo em conta o problema identificado foram realizadas várias reuniões entre os diferentes interessados na obtenção de uma solução. O principal foco, como já foi previamente referido, será informatizar o processo diário das Instituições Sociais, através de uma aplicação informática. A primeira ideia surgida foi o desenvolvimento de uma aplicação web responsiva, tendo em conta o custo/benefício. De seguida, tendo em conta o funcionamento da instituição ODPS, foi sugerido uma aplicação móvel, *software* instalado e utilizado num dispositivo móvel, com capacidade para funcionar sem necessidade de ligação permanente à internet, porque existem várias restrições de nenhum acesso em determinadas regiões de Portugal.

Posteriormente foram obtidos os requisitos funcionais e não funcionais, após a presença nas várias reuniões. Um dos requisitos que mais despertou interesse foi a necessidade de controlo no acesso a determinadas informações sensíveis dos pacientes, isto porque os profissionais de saúde lidam com bastante informação minuciosa. Outro tipo de requisitos são:

- Centralizar a informação num único local (base de dados);
- Ter opção multilíngua para vários países;
- Registo de início e fim de vista a um cliente;

- Obter dados estatísticos das informações registadas;
- Notificação aos cuidadores com possíveis recomendações de tratamento, apenas em modo *online.*

Seleção de Ideias

Como não existe um cenário ideal, uma das possíveis soluções foi a divisão em dois aplicativos: o desenvolvimento de uma aplicação móvel e de uma aplicação web. A primeira terá que ser capaz de produzir todos as opções que os cuidadores e profissionais de saúde necessitem, quando se encontrarem nas residências dos pacientes, que principalmente funcionará *offline*. A última, uma aplicação web, de gestão de *backoffice*, que será uma plataforma que gere todo o processo da instituição e que permite a parametrização de vários aspetos da aplicação móvel. As aplicações deverão enquadrar-se no novo Regulamento Geral sobre a Proteção de Dados de Portugal (RGPD), que irá ser aplicado a partir de 25 de Maio de 2018.

Desenvolvimento/Definição de Conceito

A partir da ideia selecionada foram definidas os meios e tecnologias que irão suportar o desenvolvimento da solução, bem como todos os requisitos e especificações que deverão garantir a satisfação do problema.

2.6.2 Valor, Valor Para o Cliente e Valor Percecionado

O valor é um conceito subjetivo que depende de vários fatores. Neste contexto, o valor deste projeto para o cliente vai depender dos benefícios e dos custos, das suas necessidades e expetativas para obter o produto final.

"Value has been defined in a different theoretical contexts as need, desire, interest, standard / criteria, beliefs, attitudes, and preferences." (Nicola, Susana and Ferreira, Eduarda Pinto and Ferreira, J. J. Pinto, 2012)

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

weighed combination of sacrifice and benefit (determined and expressed either rationally or intuitively); or an aggregation, over time, of any or all of these." (Woodall, 2003)

Para as instituições, os benefícios destas aplicações assumem contornos económicos, de forma a otimizar os recursos materiais e humanos das instituições, e para ser integrado no dia a dia dos cuidadores e profissionais de saúde, de acordo com as suas práticas habituais. Funcionarão como ferramenta de apoio à atividade diária dos colaboradores e facilitará a comunicação entre os diferentes intervenientes. Além disso, a informação ficará centralizada, assim que o dispositivo móvel estiver *online* e terá o benefício do controlo da informação dos pacientes num único local.

Os custos serão o alojamento para a aplicação web, a aprendizagem ao utilizar uma aplicação móvel, o desenvolvimento e a manutenção. A aplicação móvel poderá ser mitigado através da conceção de uma interface com o utilizador de fácil utilização e envolvendo os utilizadores finais na sua definição. Por outro lado, a introdução da ferramenta poderá ou deverá introduzir uma alteração nos hábitos de trabalho dos cuidadores. Referir que estes atualmente acabam por preencher as fichas depois da prestação dos serviços, mas pretende-se, com a nova aplicação, que os cuidadores passem a preencher as folhas de registo logo a seguir à prestação do serviço, garantido o registo de informação mais fiável.

"Perceived value is a subjective estimation of the trade-off between all that is accepted and all that is given up in the act of getting, using or consuming a product." (Balasubramanian, Kannan and Mala, K. and Rajakani, M. , 2016)

O valor percecionado do produto difere de cliente para cliente. Através da utilização de um determinado produto ou serviço deverá ser possível realizar uma avaliação dos benefícios e/ou funcionalidades que o cliente estará à espera.

Pretende-se que esta solução pretende-se que seja de fácil utilização, tanto para os cuidadores, como para os profissionais de saúde, assim como para os coordenadores.

2.6.3 Proposta de Valor

A solução apresentada será uma ferramenta útil para todos os intervenientes das instituições.

Para os cuidadores, a aplicação móvel servirá como registo diário das atividades de cada paciente e para que seja notificado sobre possíveis episódios de doença ou o estado de saúde do paciente.

Para os profissionais de saúde, a aplicação móvel irá assentar na interação com os cuidadores, ao perceber as atividades de cada paciente e o seu estado de saúde. Poderá registar tratamentos especializados.

Para os coordenadores, a aplicação web será útil para gerir o processo de dia a dia dos cuidadores.

Para os administradores, a aplicação web servirá para gerir os utilizadores e será útil para analisar dados estatísticos ao nível processual.

2.6.4 Modelo de Negócio CANVAS

"A business model describes the rationale of how an organization creates, delivers and capture value". (Osterwalder, Alexander and Pigneur, Yves, 2010)

O modelo de negócio Canvas tem por base 9 componentes:

- Parceiras-Chave – descreve os fornecedores e parceiros de negócio;
- Atividades-Chave – são as ações determinantes na realização do modelo de negócio;
- Recursos-Chave – conjunto de recursos físicos, financeiros ou humanos;
- Proposta de Valor – representa o(s) produto(s) ou serviço(s) que cria(m) valor aos clientes;
- Relacionamento com Clientes – descreve as atividades de comunicação que se estabelece com os clientes;
- Canais – são os meios de comunicação do negócio com os clientes;
- Segmentos de Clientes – público-alvo do negócio;
- Custos – refere a todos os custos envolvidos na realização do modelo de negócio;
- Receitas – geração de valor monetário / lucro do negócio.

A tabela 2 apresenta o modelo Canvas alinhado com o projeto AT-Home.

Tabela 2 - Modelo Canvas projeto AT-Home

<table>
<tr>
<td rowspan="2">Parcerias-Chave
Partes interessadas (Stakeholders)
Cuidadores de Idosos
Parceiros de negócio - Lar de Idosos, Centros de Saúde Fornecedores</td>
<td>Atividades-Chave
Análise de valor
Comparação com ferramentas semelhantes de mercado
Levantamento de requisitos
Análise e Design
Desenvolvimento
Teste
Manutenção
Suporte a cliente
Avaliação e Experimentação</td>
<td rowspan="2">Proposta de Valor
Para os Cuidadores, a aplicação móvel serve como registo diário das atividades de cada paciente. Seja notificado sobre possiveis episódios de doença ou o estado de saúde do paciente.
Para os Profissionais de Saúde, a aplicação móvel serve para interagir com os cuidadores, ao perceber as atividades de cada paciente e o seu estado de saúde. Regista tratamentos.
Para os Coordenadores, a aplicação web será útil para gerir o processo de dia-a-dia dos cuidadores</td>
<td>Relacionamento com Clientes
Formação ao cliente
Inquéritos de satisfação
Questionários de adaptabilidade e usabilidade
Apoio na definição de requisitos</td>
<td rowspan="2">Segmentos de Clientes
Cuidadores de Idosos
Terapeutas
Enfermeiros
Psicólogos</td>
</tr>
<tr>
<td>Recursos-Chave
Programador(es)
Servidor de Base de Dados
Servidor de Internet
Internet
Computador / Dispositivo móvel
Navegador de Internet
Aplicações móvel e web</td>
<td>Canais
Publicidade
Redes sociais
Revistas de engenharia
Eventos
Reuniões com parceiros</td>
</tr>
<tr>
<td colspan="3">Custos
Internet
Alojamento / Domínio
DPO – prestação de auxílio na área de proteção de dados</td>
<td colspan="3">Receitas
Possíveis Avenças</td>
</tr>
</table>

3 Análise e Design

O objetivo deste capítulo é analisar, descrever e elaborar uma proposta conceptual e de design. É exposta a ideia da solução, com uma breve explicação dos requisitos funcionais e não funcionais do problema e analisada diversas formas possíveis de chegar à sua resolução. São apresentadas várias abordagens ao problema ao nível do *design* da solução, tendo em conta as regras e restrições de negócio. De várias alternativas de *design* é decidida aquela que parece ser a mais adequada.

Para apresentação de artefactos na construção da solução é utilizada a vista de modelo 4+1 de Philippe Kruchten, que consiste em "organizar uma descrição de uma arquitetura de *software*, utilizando cinco vistas em simultâneo, cada uma abordando um conjunto específico de preocupações" (Kruchten, 1995). As cinco vistas representadas são as Vistas de Cenário (Casos de Uso), Lógica, Física, de Processo e de Desenvolvimento.

Por fim é apresentado um Modelo de Domínio que irá servir como base para o Diagrama de Classes e para o Modelo Conceptual de Dados.

Para representação gráfica das vistas e dos restantes modelos é utilizado a notação UML (*Unified Modeling Language*).

3.1 Visão Geral

O projeto AT-Home insere-se numa aplicação que fornece apoio a cuidadores de pessoas idosas. O principal objetivo da aplicação é facilitar a comunicação entre os Profissionais de Saúde, Responsáveis das Instituições e Cuidadores e, ao mesmo tempo, ajudar na gestão de cuidados de saúde prestados.

A aplicação será concebida de forma a ser fácil de utilizar. Uma grande atenção será dada à ergonomia da aplicação, o que não deve ser um obstáculo para os cuidadores, que serão aqueles que mais vão interagir com ela. Estes últimos devem integrar a aplicação nas suas vidas diárias e de acordo com as suas práticas habituais.

A aplicação será projetada com um *design* responsivo, para se adaptar aos meios digitais disponíveis, *smartphones* e *tablets* de preferência e estará disponível em dois sistemas operativos móveis: *Android* e *iOS*. Estará acessível gratuitamente nas lojas *Android - PlayStore* e *iOS - iTunes*, com os idiomas português, espanhol, francês e inglês.

3.2 Partes Interessadas

As partes interessadas na aplicação serão as Instituições Sociais de Apoio a Idosos. Como já foi referido, a instituição ODPS constitui o grupo de organizações nacionais e internacionais interessadas no projeto.

Por outro lado, os outros interessados são todos aqueles que estão envolvidos direta e indiretamente na organização do projeto, bem como a equipa de desenvolvimento, coordenador e orientadores.

3.3 Actores

A aplicação terá como intervenientes responsáveis das Instituições, que serão divididos em quatro papéis:

- Coordenadores – gerem todas de tarefas dos cuidadores. Monitorizam toda a atividade dos cuidadores, utilizando a aplicação *backoffice*;

- Administradores – gestores dos utilizadores do sistema. Podem verificar dados estatísticos do trabalho realizado, utilizando a aplicação *backoffice*;

- Profissionais de Saúde – técnicos de saúde responsáveis pela monitorização de todos os pacientes. Monitorizam a atividade dos cuidadores, utilizando a aplicação móvel;

- Cuidadores - papel importante no acompanhamento dos pacientes. Podem ser alertados pelos profissionais de saúde sobre possíveis episódios de doença ou o estado atual de saúde ou doença dos pacientes. Fazem o registo diário das suas atividades na aplicação móvel e são monitorizados remotamente pelos Profissionais de Saúde.

3.4 Requisitos

Após várias reuniões com a Instituição ODPS e coordenador do projeto, que é o intermediário com as organizações internacionais, foi possível definir as necessidades de cada um. Para validação dos requisitos propôs-se um documento respetivo, posteriormente validado.

3.4.1 Requisitos Não Funcionais

- Disponibilidade de dispositivos móveis recentes, com base em sistemas operativos Android e iOS, com acesso à Internet;

- Realizar todos os requisitos legais, ética e segurança dos dados registados e fornecidos, de acordo com as leis vigentes, bem como os regulamentos que começarão a ser aplicados em Maio de 2018 nos Estados Membros da União Europeia (RGPD);

- Ergonomia na interface com o utilizador.

3.4.2 Requisitos Funcionais

- Possibilidade de funcionamento *online* e *offline*: no modo *offline*, os recursos disponíveis ficam limitados ao registo de ocorrências e no acesso à informação do paciente;
- Compatível com dispositivos Android e iOS;
- Interface com o utilizador em quatro idiomas: português, espanhol, francês e inglês;
- Carregamento diário das informações dos pacientes para a memória interna do dispositivo móvel no início do dia, na maior parte das vezes nas instalações das Instituições;
- Identificação dos pacientes por *QR Code* nas suas habitações. Deve identificar o paciente e a hora de início da prestação de serviço e, possivelmente, o tempo de término;
- Aplicação com autenticação e autorização a conteúdos;
- Gestão de utilizadores pelo Administrador:
    - Criar perfis de utilizador: profissionais de saúde, cuidadores, coordenadores e administradores;
    - Criação personalizada de utilizadores com permissões específicas;
    - Eliminação de utilizadores.

As funcionalidades estão divididas em três grupos:

- Registo de Intervenções:
    - Permitir o registo de intervenções pelos cuidadores e profissionais de saúde, através do preenchimento de uma ficha de intervenção;
    - Permitir o registo de ocorrências imprevistas anteriormente tipificadas, escolhendo uma lista predefinida de ocorrências - será necessário definir a lista de ocorrências, por cuidadores e por profissionais de saúde;

- Permitir o acesso à informação de serviços contratados pelo paciente (cuidadores e profissionais de saúde);
- Permitir registar o tratamento da roupa com 3 eventos:
  - Recolha de roupas nas casas dos pacientes para o centro de reabilitação;
  - Registo de roupas para a lavandaria;
  - Registo de roupas devolvidas aos pacientes.
- Permitir acesso a informações registadas pelo profissional de saúde:
  - Acesso a todas as informações disponíveis aos cuidadores;
  - Acesso aos registos vitais dos pacientes;
  - Acesso à ficha de cuidados de enfermagem.
- Monitorização de todas as intervenções aos pacientes, pelos profissionais de saúde na aplicação móvel e pelos coordenadores na aplicação de *backoffice;*
- Gestão de tarefas para cada cuidador pelos coordenadores.

- Registo Eletrónico de Atendimento Clínico
  - Registos vitais dos pacientes pelos profissionais da saúde manualmente;
  - Registo de cuidados de saúde pelos profissionais de saúde;
  - Permitir descarregamento de informação sensível dos pacientes em PDFs pelos profissionais de saúde.

- Suporte a Aprendizagem a Cuidadores (Recomendações e Tutoriais):
  - Acesso a vídeos, questionários e tutoriais para permitir a aprendizagem à distância como, por exemplo, transferência de pacientes pesados ou

avaliação de estados mentais. Tudo quando a aplicação estiver no modo *online.*

3.5 Vista de Cenário - Casos de Uso (UML)

Os Casos de Uso são um modo representativo da interação dos intervenientes com o sistema.

Na Figura 2 é apresentada uma representação dos principais casos de uso do sistema AT-Home.

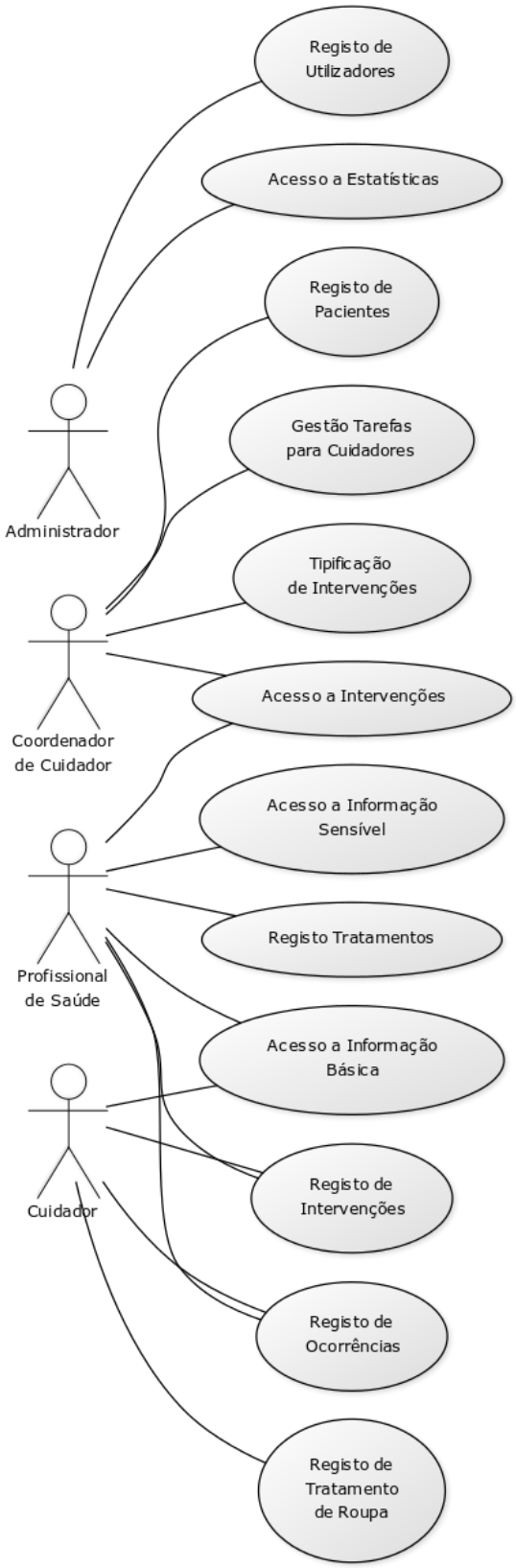


Figura 2 - Diagrama de Casos de Uso (UML)

- Registo de Utilizadores – registo das contas de utilizadores da(s) Instituição(ões) e associação por tipo. Haverá a possibilidade de personalização de perfil por utilizador;

- Acesso a Estatísticas – visualização gráfica de estatísticas de vários critérios e tipificações pré-definidas, tais como, estatísticas por paciente, por tipo de intervenções, entre outros. Só poderá ser acedida pelos administradores;

- Registo de Pacientes – registo de todos os pacientes ao cuidado das Instituições, pelos coordenadores. Deve ser registada a informação básica de cada paciente, nomeadamente nome, morada, idade, sexo, estado de saúde do paciente, entre outros;

- Gestão de Tarefas para Cuidadores – gestão de tarefas geridas pelo coordenadores, onde irá referir, a cada cuidador, o que deve realizar no seu dia a dia de trabalho e atribuir o respetivo paciente. Pode estar dependente da tipificação de intervenções;

- Tipificação de Intervenções – registo de intervenções tipificadas a serem utilizadas pelos cuidadores e profissionais de saúde. É um registo realizado pelo coordenadores. O tipo de intervenções a contemplar será nutrição (pequeno-almoço, almoço, lanche, jantar, seia), higiene (pessoal, habitacional), transporte, apoio nas refeições, entre outros;

- Acesso a Intervenções – visualização das intervenções realizadas pelos cuidadores, acedidas pelos profissionais de saúde e coordenadores;

- Acesso a Informação Sensível – visualização de informação de saúde dos pacientes. É um tipo de informação que é reservada pelos profissionais de saúde, que está cingida por lei. Pela Instituição ODPS, este tipo de informação será acedida pela aplicação através de um PDF, que pode ser submetido pelo profissional de saúde. Como se trata de informação sensível deve-se garantir a integridade e proteção dos dados. No caso de organizações internacionais poderá haver hipótese deste tipo de informação seja registada pelo profissional de saúde na aplicação;

- Registo de Tratamentos – registo dos tratamentos realizados aos pacientes, pelos profissionais de saúde. Estão englobados os registos de cuidados de enfermagem, serviço de psicologia, sinais vitais, medicação, entre outros. Consoante o tipo de problema que poderá ter um determinado paciente, o profissional de saúde deve registar o estado de saúde do mesmo na aplicação e o cuidador, que estará associado ao paciente, receberá uma notificação da alteração do estado;

- Registo de Ocorrências – registo de determinadas ocorrências inesperadas do paciente, que estão fora do âmbito do tipo de intervenções realizadas pelos cuidadores. Um exemplo: quando o cuidador chega a casa de um paciente e este está desmaiado;

- Acesso a Informação Básica – visualização da informação básica de cada paciente, pelos cuidadores e profissionais de saúde. Será aquela informação referida no registo de pacientes;

- Registo de Intervenções – registo dos serviços contratualizados pelas instituições, nomeadamente nutrição, higiene, transporte, entre outros. Será um registo realizado principalmente pelos cuidadores, quando estes se deslocarem às residências dos clientes. Os profissionais de saúde podem realizar este registo também. Estes tipos de registos estão previamente tipificados será de fácil utilização. Os cuidadores só necessitarão de selecionar aqueles que foram realizados;

- Registo de Tratamento de Roupa – registo de peças de roupas que foram recolhidas nas casas dos pacientes para o centro de reabilitação, pelos cuidadores. Posteriormente, esse tipo de roupa irá ser reencaminhada para uma respetiva lavandaria. Por fim, a mesma roupa será devolvida ao paciente. É importante saber que peças de roupa são e onde estas se encontram.

3.6 Modelo de Domínio

O Modelo de Domínio é uma visão conceptual do domínio do sistema, sendo representado pelas entidades, as suas associações e multiplicidades.

Na Figura 3 é proposto um modelo representativo do domínio do sistema.

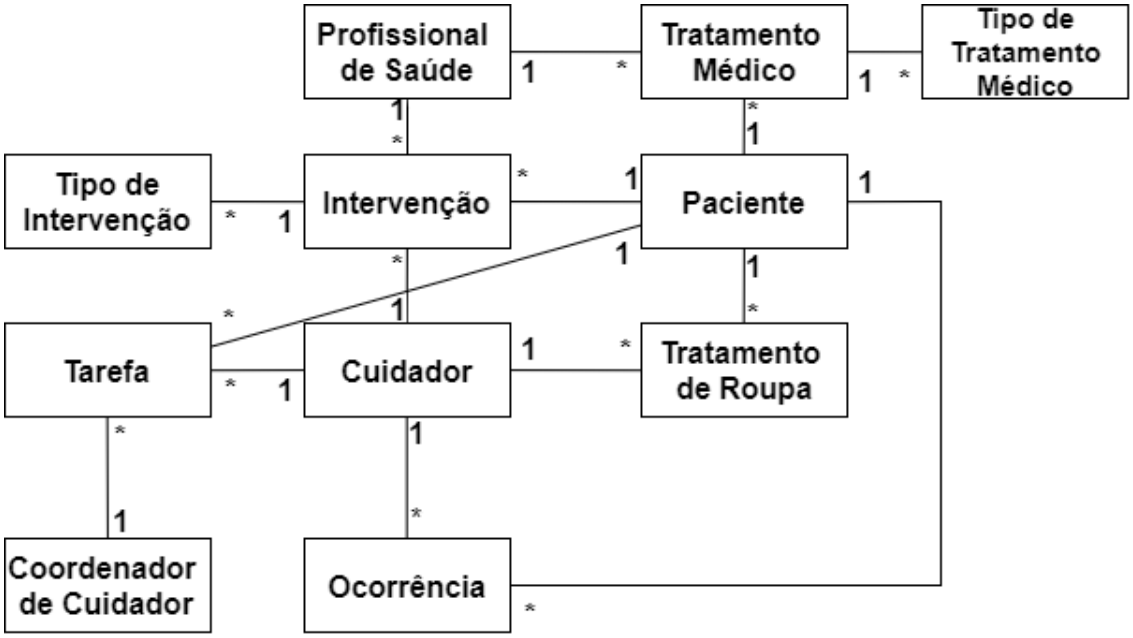


Figura 3 - Modelo de Domínio (UML)

O Coordenador de Cuidador gere várias Tarefas a realizar pelos Cuidadores, de modo a gerir o seu trabalho. As diferentes Tarefas estão associadas a um Paciente.

As Intervenções podem ter vários Tipos de Intervenção, que podem ser realizadas ou pelos Cuidadores ou pelos Profissionais de Saúde a um Paciente.

O Profissional de Saúde pode realizar vários Tratamentos Médicos a um Paciente, utilizando diferentes Tipos de Tratamentos Médico.

Já o Cuidador realiza vários Tratamentos de Roupa e regista Ocorrências associadas a um Paciente.

Tendo em conta a opção tomada de uma base de dados relacional e, tomando como base esta representação de domínio, os próximos passos serão a realização do Diagrama de Classes na notação UML e, ao mesmo tempo, a construção do Modelo Conceptual de Dados.

3.7 Modelo Relacional

A aplicação do modelo relacional no projeto surgiu como base do modelo de domínio, utilizando as entidades de domínio e as suas ligações.

Tendo em conta que o modelo é algo extenso foi anexado a este documento - anexo 1.

As entidades representadas no modelo relacional, a maioria delas foram extraídas do modelo de domínio, visto que estas são entidades de domínio do sistema e representam tabelas de base de dados. As relações das mesmas são semelhantes às do modelo de domínio. Por exemplo, uma intervenção (*interventions*) contém sempre um tipo de intervenção (*intervention*_types). No modelo relacional, esta situação está referenciada por uma chave estrangeira na coluna "*intevention_type_id*". Por consequência, um tipo de intervenção pode estar referênciado em várias intervenções.

As entidades que não estão representadas no modelo de domínio, mas que estão representadas neste modelo relacional são:

- Os utilizadores e tipos de utilizador, representados por *users* e *user_types*, que caracterizam os utilizadores do sistema. Aqui estão englobados os pacientes, profissionais de saúde, coordenadores e administradores;
- As permissões e o tipo de permissões, representados por *permissions* e *permission_types* respetivamente, que irão ser utilizadas na criação de um perfil de utilizador personalizado, que foi um dos requisitos requiridos pelo coordenador do projeto;
- Os idiomas, representados por *languages*, para o sistema permitir multi-idiomas;
- As lavandarias, representadas por *laundries,* para identificar as lavandarias utilizadas na lavagem da roupa dos clientes;
- O tipo de roupa, representado por *clothing_type,* que pode servir para categorizar as roupas recolhidas pelos cuidadores. Esta funcionalidade ainda não foi devidamente especificada*;*
- O estado do tratamento de roupa, representado por *clothing_treatment_status,* que associa ao tratamento de roupa, o estado atual da situação da roupa dos clientes.

3.8 Vista Lógica (UML)

A Vista Lógica seguida neste projeto é apresentação dos principais componentes do sistema e como estes se interligam, tal como se pode comprovar na Figura 4:

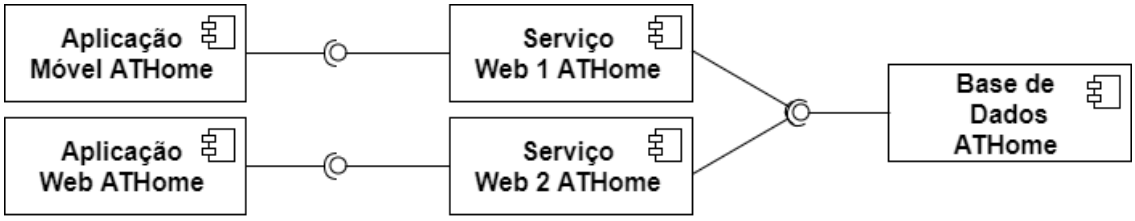


Figura 4 - Vista Lógica (UML)

A representação da vista lógica é dividida por cinco componentes:

- Base de Dados AT-Home – componente que contém um Sistema de Gestão de Base de Dados (SGBD), onde irá ser persistida toda a informação das aplicações. Disponibiliza uma interface para os Serviços Web 1 e 2 AT-Home possam implementar. Permitirá operações CRUD (*Create, Read, Update, Delete);*
- Serviço Web 1 AT-Home – componente de serviço web que disponibiliza uma API (*Application Programming Interface*) com um conjunto de operações realizadas pela aplicação móvel. É o local onde se encontra os casos de usos relativos aos cuidadores e profissionais de saúde. Tem um acesso à componente de Base de Dados;
- Serviço Web 2 AT-Home – componente de serviço web que disponibiliza uma API com um conjunto de operações realizadas pela aplicação web. É o local onde se vão encontrar os casos de usos relativos ao coordenadores e administradores. Tem um acesso à componente de Base de Dados;
- Aplicação Móvel AT-Home – componente de interface gráfica, onde o utilizador pode realizar as funcionalidades disponibilizadas pelo Serviço Web 1 AT-Home;
- Aplicação Web AT-Home – componente de interface gráfica, onde o utilizador pode realizar as funcionalidades disponibilizadas pelo Serviço Web 2 AT-Home.

Uma possível alternativa à vista lógica referida é unificar os dois Serviços Web num único serviço, tal como se pode verificar na Figura 5:

Figura 5 - Vista Lógica (UML) - alternativa 1

Em comparação com a Figura 4, a principal vantagem desta abordagem é o menor custo, isto é, todas as operações e lógica de negócio encontram-se num único local, para além de que apenas existe um único acesso à componente de base de dados.

Já as desvantagens desta abordagem são a fraca escalabilidade e manutenção do serviço ***web***. Quando se deseja acrescentar uma nova operação ao serviço este ficará mais sobrecarregado e vai contra um dos princípios da escalabilidade, que é decomposição. Quando se pretende alterar uma operação é necessário voltar a implantá-lo, pelo que se a operação estiver referenciada a outras, estas podem ser afetadas. Além disso, a implantação do serviço demorará mais tempo consoante o número de operações.

Outra alternativa possível às vistas lógicas anteriores é suprimir o serviço Web.

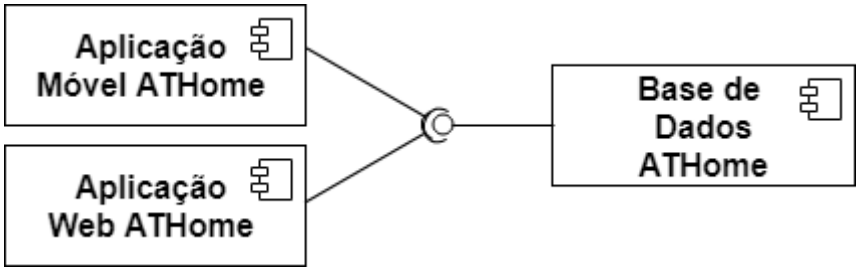


Figura 6 - Vista Lógica (UML) - alternativa 2

Não se consegue prever uma vantagem efetiva desta solução em detrimento das anteriores.

As desvantagens desta abordagem é que a lógica de negócio estará "espalhada" ou pelas aplicações ou pela base de dados, o que tornará a manutenção muito mais difícil de se fazer. Continua a existir o problema de escalabilidade da alternativa anterior.

3.9 Vista Física (UML)

A Vista Física reflete a implantação física dos componentes. Na Figura 7 é proposta uma possível representação da topologia dos componentes da solução.

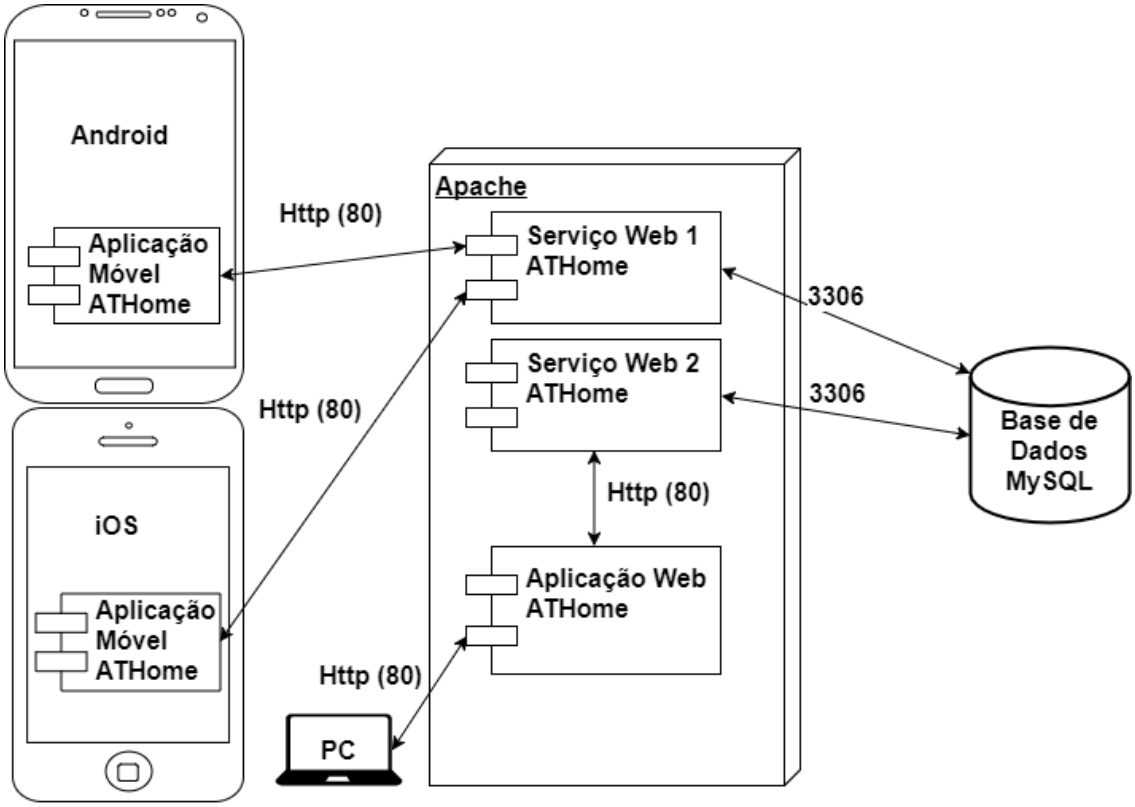


Figura 7 - Vista Física (UML)

A representação da vista física é dividida por três grandes componentes:

- Base de Dados MySQL – componente SGBD que utiliza a linguagem SQL (*Structured Query Language*) para realizar operações na base de dados. Por defeito, a sua ligação será na porta 3306;
- Apache - componente servidor web *Apache,* onde estão instalados os componentes Serviços Web 1 e 2 AT-Home, bem como o componente da Aplicação Web AT-Home. Apenas os componentes de Serviços Web fazem ligação com a base de dados, utilizando o porto definido. Por outro lado, o componente Aplicação Web é aquele que se interliga com o Serviço Web 2, para efetuar apenas operações relativas à Aplicação Web, através do protocolo HTTP (*Hypertext Transfer Protocol*) – protocolo de comunicação entre sistemas – no porto 80. É expetável que seja utilizado um PC (Computador) para aceder à Aplicação Web, através do protocolo HTTP no porto 80, não invalidando que possa ser acedido por qualquer outro dispositivo, desde que tenha acesso à internet;
- Aplicação Móvel AT-Home – componente que está dividida consoante o número de dispositivos móveis utilizados, tanto em versão *Android* como *iOS*. Só deve conhecer a

ligação ao Serviço Web 1 AT-Home, por ser aquele serviço que realiza as operações para a aplicação móvel.

Uma possível alternativa à representação da Figura 7 é repartir os serviços Web e a aplicação Web por diferentes servidores Apache, tal como demonstra a Figura 8:

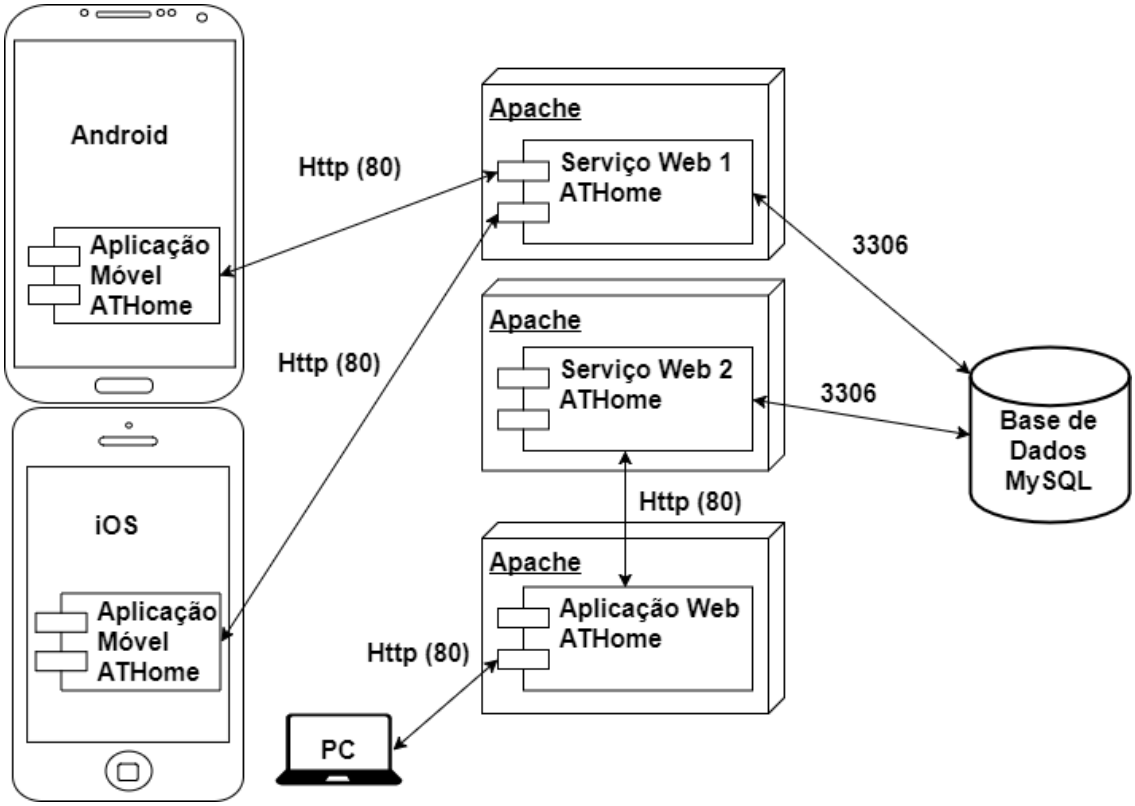


Figura 8 - Vista Física (UML) - alternativa 1

Esta alternativa de vista física é mais viável que a representação anterior pelo maior isolamento dos diferentes componentes. É bastante mais fácil o sistema ser escalável nesta abordagem. Por exemplo, se se projetar um aumento de pedidos ao Serviço Web 2, houver uma sobrecarga e caso se tenha múltiplas instâncias deste componente, através desta abordagem é mais fácil reencaminhar a ligação deste serviço para uma outra instância idêntica. Por outro lado, caso se queira fazer uma nova implantação, por exemplo, no Serviço Web 1, não existe afetação nos outros componentes, porque são todos fisicamente independentes.

A principal desvantagem desta abordagem é o custo. Ao seguir esta alternativa será necessário obter três alojamentos web, o que representaria mais investimento por parte das

Instituições. Como o custo influencia diretamente a escolha, esta abordagem, apesar de ter mais benefícios que a anterior, terá que ser descartada.

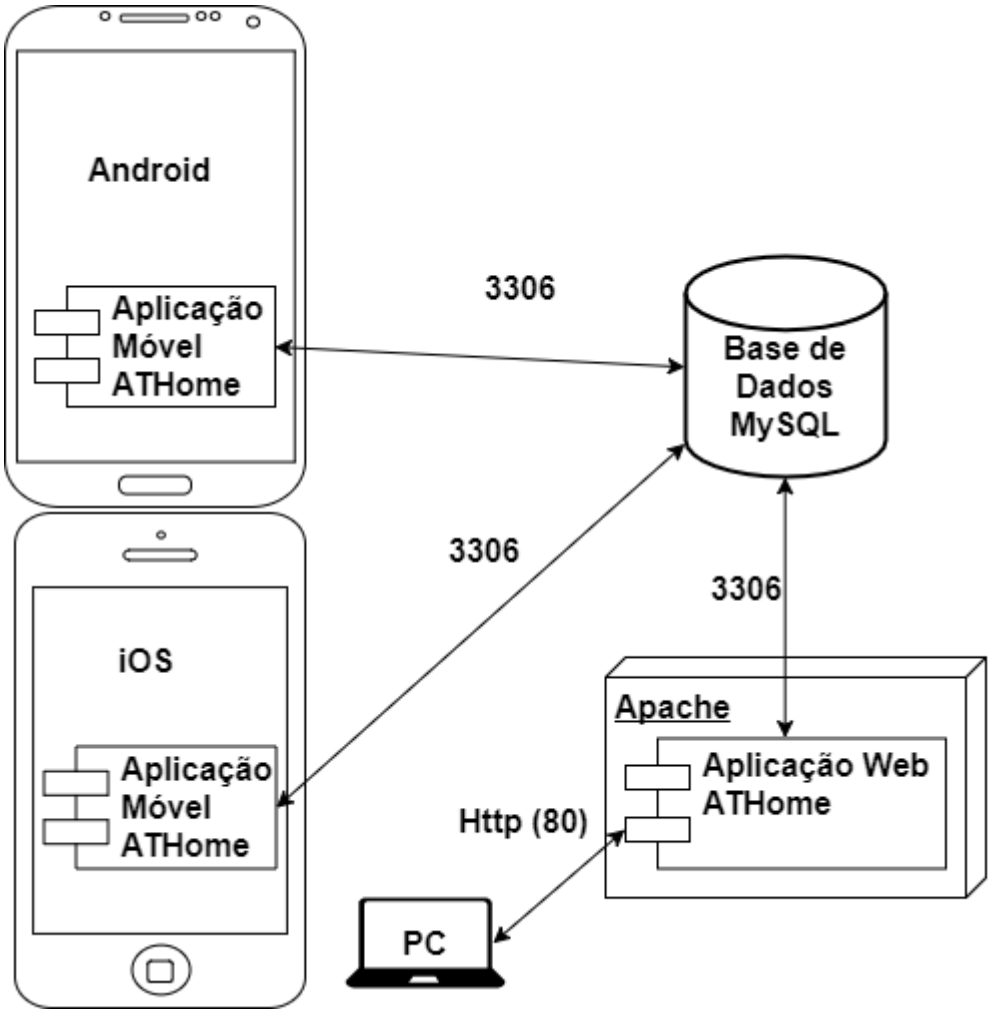


Figura 9 - Vista Física (UML) - alternativa 2

Outra alternativa à primeira abordagem de Vista Física é a não existência de Serviços Web. A vantagem desta abordagem é a não dependência de uma máquina intermédia, o que traria uma redução de custo. No entanto, não é muito viável que a lógica de negócio esteja nas aplicações de interface ao utilizador ou na base de dados ou repartida em ambas, pois carece de fraca escalabilidade e manutenção.

Uma terceira alternativa de Vista Física é a alteração da tecnologia. A Figura 10 está representada nesse sentido:

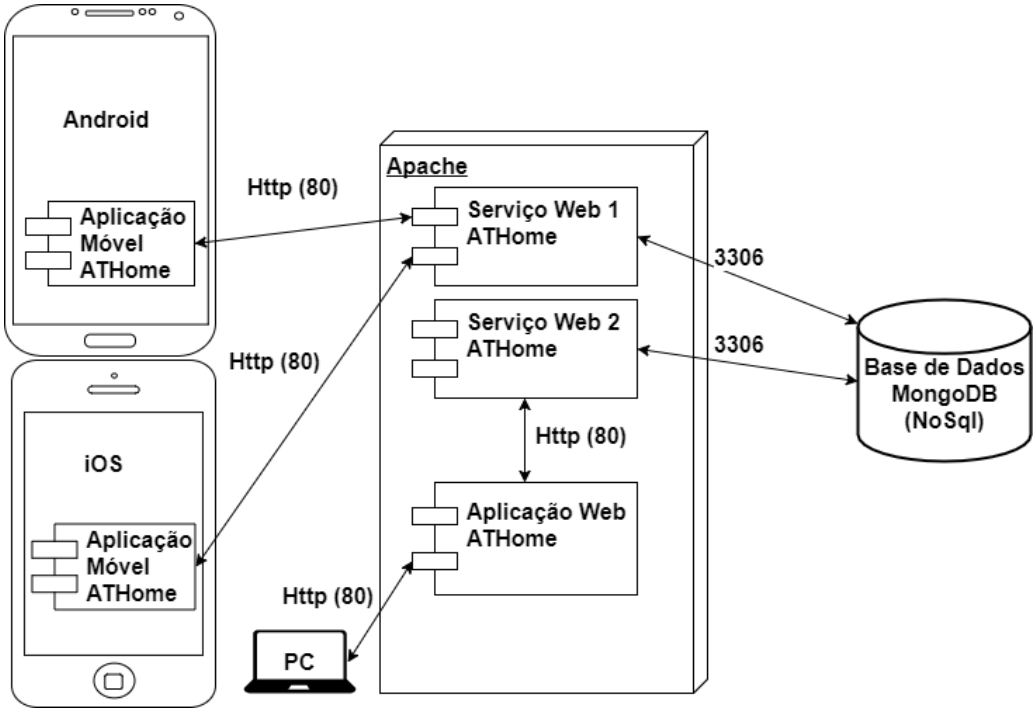


Figura 10 - Vista Física (UML) - alternativa 3

A mudança de paradigma do componente de base de dados, passando de uma base de dados relacional MySql para uma base de dados não relacional como, por exemplo, MongoDB é uma das possíveis soluções.

Uma base de dados relacional é uma estrutura de dados que permite interligar informações de diferentes tabelas. Já uma base de dados não relacional, ou NoSql, apenas armazena dados sem mecanismos explícitos e estruturadas. (Padhy, R. P. and Patra, M. R. and Satapathy, S. C, 2011).

As bases de dados não relacionais vêm colmatar as deficiências das bases de dados relacionais, como os problemas de escalabilidade. Por outro lado têm a limitação na dificuldade de manutenção. (Leavitt, 2010)

Na tabela 3 é feita uma comparação das bases de dados não relacionais para as bases de dados relacionais:

Tabela 3 - Vantagens e desvantagens de base de dados não relacionais (Leavitt, 2010)

Vantagens	**Desvantagens**
Fornece uma ampla escolha de modelos de dados	Imaturidade
Facilmente escalável	Inexistência da linguagem Sql

Não são necessários administradores de base de dados	Sem interface padrão
Maior rapidez, eficiência e flexibilidade	Difícil manutenção

As bases de dados não relacionais têm melhores vantagens que as bases de dados relacionais relativamente à rapidez, eficiência, flexibilidade e escalabilidade. No entanto, a diferença no custo em comparação com as bases de dados relacionais é muito grande e a instituição não consegue disponibilizar meios financeiros para isso, desconsiderando a curva de aprendizagem da equipa de desenvolvimento, referente à tecnologia.

3.10 Vista de Processo (UML)

Através da vista de processo consegue-se perceber quais as atividades do sistema ao nível dos processos e como estes se comunicam, em tempo de execução.

Na Figura 11 é representada uma das principais atividades do sistema, nomeadamente o registo de tratamentos realizado por um profissional de saúde ligado à enfermagem.

Figura 11 - Diagrama de Atividades - Registo de Tratamentos

Quando um profissional de saúde, nomeadamente um enfermeiro, efetuar um tratamento a um paciente deve registar os cuidados de enfermagem que efetuou. De seguida terá que verificar os sinais vitais do paciente. Posteriormente pode receitar uma medicação consoante o tipo de problema encontrado. Por fim, caso haja alguma alteração no estado do paciente e seja registado no sistema deve notificar os cuidadores da mudança do estado de saúde.

3.11 Vista de Implementação (UML)

Na implementação de *software* irão ser seguidas algumas boas práticas, tais como, adoção de um processo iterativo e incremental, estilos e *standards* apropriados, desenvolvimento e aplicação de testes e controlo de versões *GIT*.

Para além das boas práticas recomendadas na implementação de uma solução de *software*, neste momento e, mesmo havendo algumas restrições nas vistas anteriores, ainda não foram tomadas decisões na vista de implementação, nomeadamente linguagens de programação orientadas a objeto a utilizar, padrões de *software* ou ferramentas de desenvolvimento.

4 Implementação

Com este capítulo pretende-se demonstrar como se irá chegar à implementação da solução. É apresentada a metodologia utilizada, com o auxílio do capítulo 3 (Análise e Design), para se justificar as decisões tomadas. São também identificadas as tecnologias utilizadas, bem como a importância das mesmas no desenvolvimento do projeto.

Para suporte ao desenvolvimento das funcionalidades são ilustrados, através de diagramas de sequência, os principais fluxos de controlo efetuados, desde os pedidos efetuados pela aplicação móvel até a resposta do serviço web.

No desenvolvimento da aplicação foi adotado um processo iterativo e incremental, suportado numa metodologia ágil, isto é, à medida que se pretende desenvolver as funcionalidades, estas passarão pelos processos de análise, codificação e teste.

4.1 Metodologia

Foi projetado a realização de *sprints*, periodos de realização de tarefas para serem revistas, no mínimo quinzenais e no máximo mensais. Como suporte a *sprints* foi previsto um controlo de versões *Git*, com o auxílio do *Bitbucket* como repositório do código fonte.

Tendo em conta a arquitetura definida no capítulo 3 (Análise e Design) foram definidas duas API's em serviços web, como forma de comunicação entre a aplicação móvel e base de dados, e entre a aplicação web e base de dados respetivamente. Os serviços web terão as nuances e

operações diretas aos dados. A tecnologia utilizada para contemplar as APIs foi o estilo arquitetural REST (*Representational State Transfer*), tornando as APIs *Restful*.

A equipa de desenvolvimento do projeto são dois alunos do Instituto Superior de Engenharia do Porto, Roberto Silva, aluno de Mestrado de Engenharia de Software e Bruna Teixeira, aluna de Licenciatura em Engenharia Informática.

Por decisão do coordenador do projeto foi dada a prioridade ao desenvolvimento da aplicação móvel, pelo que o serviço web que a suporta também passou a ser prioritário. Neste momento, estas duas componentes são as duas que estão desenvolvidas e testadas.

A aplicação web e o serviço que a vai suportar a aplicação encontram-se em desenvolvimento.

4.2 Tecnologias utilizadas

Para a realização do código fonte do serviço web, que suporta a aplicação móvel utilizou-se a tecnologia *open-source* Laravel, baseada em PHP (*Hypertext Preprocessor*).

A decisão de escolha da tecnologia e linguagem de programação dependeu da facilidade de aprendizagem e constante evolução, por ser gratuita e também pela consideração de ser a mais adequada num sistema multiplataforma.

A tecnologia Laravel suporta programação orientada a objetos, pelo que foram seguidas boas práticas com o uso de diferentes padrões de *software*. Ao analisar os diferentes padrões detetou-se GRASP (*General Responsibility Assignment Software Patterns*), por facilmente se conseguir detetar os princípios dos padrões através do código fonte desenvolvido.

Para disponibilizar o serviço web e base de dados local foi utilizada a ferramenta XAMPP, que contém um pacote de instalação com um servidor web local Apache, base de dados MySQL e PHP.

4.2.1 PHP

PHP é uma linguagem de código aberto, também conhecido por *open-source,* que é especialmente utilizada no desenvolvimento web e que pode ser embebida na linguagem HTML – editor de hipertextos. A principal utilização do PHP é na criação de *scripts* no

ambiente servidor (*server-side scripting)*, para gerar páginas com conteúdo dinâmico (PHP.NET, 2018).

4.2.2 XAMPP

XAMPP é o ambiente de desenvolvimento mais conhecido de PHP, completamente grátis, fácil de instalação da distribuição *Apache*, contendo *MariaDB*, PHP e *Perl* (XAMPP, 2018).

4.2.3 Laravel

Laravel é uma ferramenta, também conhecida por *framework, open-source*, utilizando por base a linguagem PHP. Tem como principais características a sua fácil utilização e a modularidade. Facilita o uso de tarefas comuns utilizadas no desenvolvimento web, tais como, a autenticação, o controlo de rotas, o uso de sessões e armazenamento em cache (Laravel, 2018).

Para instalação da ferramenta é necessário primeiro a instalação do PHP, superior à versão 5. A versão de PHP utilizada foi 5.5.

Através do *website* da ferramenta Laravel foi seguido o tutorial recomendado de instalação e configuração:

- Instalação do *Composer,* ferramenta de gestão de dependências de PHP, através do *setup* do site oficial. Pode, no entanto, ser instalado via linha de comandos. A versão utilizada foi a 1.6.3;
- Instalação do Laravel, através da linha de comandos: *"composer global require "laravel/installer""*. A versão utilizada foi a 5.2;
- Criação de um projeto Laravel, através da linha de comandos: *"composer create-project laravel/laravel athomews"*;
- Para disponibilizar o serviço, ou se utiliza o XAMPP na máquina local para desenvolvimento, como já foi referenciado no início deste capítulo 4, ou se utiliza o comando *"php artisan serve --port=8080",* que foi utilizado no *deployment* do serviço, numa máquina virtual disponível para aceitar pedidos da aplicação móvel;
- De salientar que as configurações dos ambientes de desenvolvimento e produção sejam diferentes devido às restrições de cada máquina. Para verificar os diferentes

ambientes, cada um contém um ficheiro, com o formato .ENV, com a sua própria configuração de base de dados, entre outras.

Na Figura 12 - Estrutura de diretórios Laravel é representada a estrutura de diretórios de um projeto Laravel:

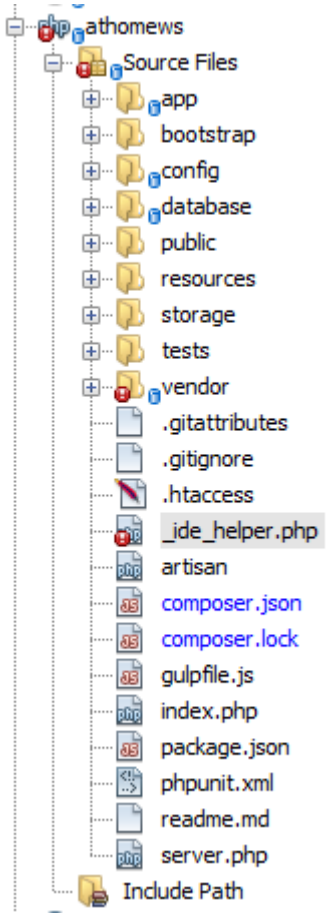


Figura 12 - Estrutura de diretórios Laravel

De salientar, as diretorias "*app*", "*config*", "*database*", "*tests*" e "*vendor*". A diretoria "*app*" contém o controlo de rotas e autenticação que foram importantes na realização desta *API* e, também, outras potencialidades da *framework* Laravel. Ainda na mesma diretoria encontram-se as entidades representadas no modelo de domínio, na Figura 3. Para criação de uma entidade utilizador, por exemplo, o comando "*php artisan make:model Utilizador -m*" tem esse prepósito. O prefixo *"-m"* serve para referir que o modelo terá um ficheiro de migração associado, de forma a que a entidade seja criada na base de dados.

Na diretoria "*config*" encontram-se todas as configurações que suportam a *framework* Laravel. A configuração mais útil surgiu com a utilização de um *plugin* "*JWTAuth*". Este suporta a

autenticação das chamadas às rotas do serviço. Por vezes é necessário limpar a cache das configurações e, para o fazê-lo é necessário executar o comando "*php artisan config:cache*".

A diretoria "*database*" refere-se às migrações e populações ou *seeds* realizadas na base de dados utilizada no serviço. As migrações contém todas as criações, alterações e ligações de tabelas/colunas de cada entidade de domínio e as *seeds* são todas as inserções que devem ser criadas, por defeito, para que as aplicações móvel e *backend* sejam suportadas. Nativamente, tipificações como tipos de utilizador, tipos de tratamento, entre outras são registadas neste módulo.

Para criar uma entidade na base de dados, depois do modelo ter sido criado previamente é necessário editar o ficheiro de migração e criar as colunas da tabela respetivas. Por fim é necessário executar o comando "*php artisan migrate*".

Para criar um "*seeder*" de tipificação na base de dados, por exemplo Tipos de Utilizador deve-se executar o comando "*php artisan make:seeder TiposUtilizador*". Posteriomente editar o ficheiro *seeder,* de modo a configurar os valores das colunas da tabela respetiva. Finalmente, o comando "*php artisan db:seed --class=TiposUtilizadorTableSeeder*" para executar as alterações efetuadas.

A diretoria "tests" contém todos os testes unitários realizados às rotas do serviço.

Finalmente, na diretoria "vendor" encontram-se todos os *plugins* associados à *framework* Laravel que, no caso particular é utilizado o plugin "*JWTAuth*" que serve como suporte a autenticação.

A *framework* Laravel, por defeito contém um módulo de autenticação na sua instalação. No entanto, por incompatibilidade de versão da *framework* e na falha da sua implementação foi escolhida e configurada um outro módulo de autenticação, através do plugin "*JWTAuth*".

Para instalação do *plugin* "*JWTAuth*" foi necessário atualizar a ferramenta *Composer*, através do comando *"composer update".* De seguida, o comando *"php artisan vendor:publish --provider="Tymon\JWTAuth\Providers\JWTAuthServiceProvider""* fez a instalação do *plugin*. Para completar a instalação teve que ser executado o comando "*php artisan jwt:generate*" para ser gerada uma chave aleatória, que "assina" os *tokens* de autenticação do serviço.

4.2.4 Git e Bitbucket

Git é um sistema de controlo de versões *open-source* gratuíto. *Git* tem um excelente suporte a ramificação (*branching*), fusão (*merging*) e reescrita (*rewriting*) do histórico de repositórios (Git, 2018).

Bitbucket é a solução Git para equipas de desenvolvimento, que querem gerir o seu próprio repositório de código fonte (Atlassian, 2018).

O *Git* foi e continua a ser importante no processo de desenvolvimento do projeto, por ser facilmente acedido em qualquer máquina, que tenha acesso ao repositório *Bitbucket*. Em concreto no servidor de produção, para obter as alterações efetuadas ao código fonte foram executados os seguintes comandos *Git*:

- *"git clone",* para obter, na primeira vez, o código fonte do repositório *Bitbucket*;
- *"git pull",* para obter, o código fonte do repositório *Bitbucket* nas vezes seguintes.

Já na máquina de desenvolvimento do código fonte, os comandos *Git* mais executados foram:

- *"git commit -m "First commit"",* para criar uma nova versão ou alteração do código fonte;
- *"git push origin master",* para submeter a alteração do código fonte no repositório *Bitbucket.*

4.3 Serviços implementados

Tendo em conta os casos de uso já referenciados nem todos foram implementados, visto que alguns deles são relativos à aplicação web e esta ainda se encontra em desenvolvimento. Assim sendo, os serviços mais relevantes foram aquelas que suportam a aplicação móvel:

- Registo e acesso a ocorrências;
- Registo e acesso a intervenções e tipificação das mesmas;
- Registo e acesso à informação básica do paciente;
- Registo e acesso a tratamentos de saúde e tipificação das mesmas.

Os serviços são implementados por um sistema de rotas (*routing*), baseado no estilo arquitetural Rest, ou seja, seguem a seguinte especificação:

- http://193.136.62.28:8080/patient/{id}, em http://193.136.62.28:8080 corresponde ao servidor, *patient* corresponde ao nome do recurso, que se quer efetuar um pedido e {*id*} corresponde aos paramêtros.

Para definir o acesso aos serviços extraíram-se os domínios a que estas pertencem, isto é, deram-se nomes aos recursos a efetuar:

- Ocorrência corresponde ao recurso "*occurrence*";
- Intervenção corresponde ao recurso "*intervention";*
- Tipificação de intervenção corresponde ao recurso "*interventiontype";*
- Paciente corresponde ao recurso "*patient*";
- Tratamento de saúde corresponde ao recurso "*healthtreatment*";
- Tipificação de tratamento de saúde corresponde ao recurso "*healthtreatmenttype*".

4.3.1 Registo e acesso a ocorrências

De forma a visualizar as ocorrências efetuadas, o cuidador deve solicitar na aplicação móvel o menu respetivo. Assim que esse menu é acionado, um conjunto de passos é efetuado. Na Figura 13 é representado esse conjunto de passos.

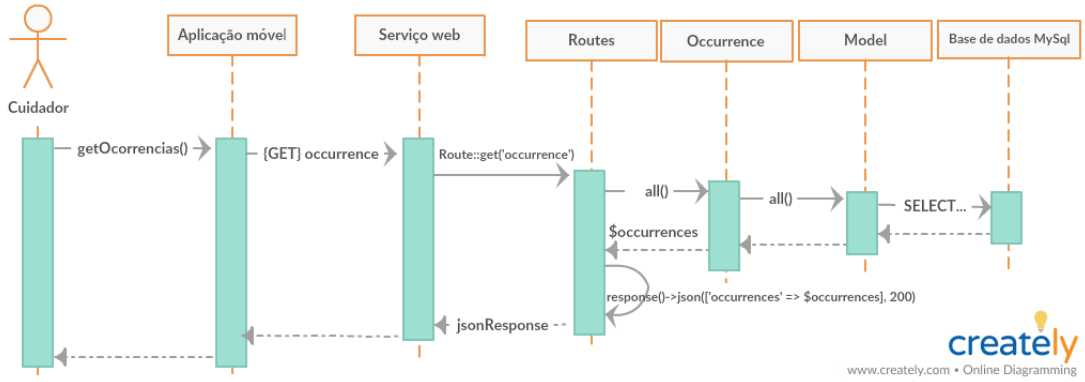


Figura 13 - Diagrama de sequência acesso a ocorrências

O primeiro passo da aplicação móvel é chamar o serviço web, efetuando um pedido http, através do método Get, ao recurso "*occurrence*". Para este caso, o *link* do pedido é http://193.136.62.28:8080/occurrence, em que 193.136.62.28:8080 representa o *IP* da máquina em que o serviço web está alojado. Para além disso, é necessário definir os cabeçalhos e o tipo de método do pedido:

```
“Method” : “GET”,
“Headers” : [{
      “Content-Type” : “application/json”
}]
```

Código 1 – Exemplo de pedido http, método Get

Todos os recursos encontram-se na componente “*Routes*”, que consoante o método do pedido e o nome do recurso reencaminha para a rota respetiva. A rota “*occurrence*” tem associada a si a sua própria classe modelo “*Occurrence*” e faz a chamada ao método *“all()”*, para obter toda as ocorrências, sem qualquer filtro. Por sua vez, a classe “*Occurrence*”, como estende da classe “*Model*”, efetua a mesma chamada ao método *“all()”*. Como a classe “*Model*” é genérica a todas as classes modelo é a única que tem acesso direto à base de dados. Logo, é esta classe que obtém os dados relativos às ocorrências, que estão registadas na base de dados. Assim que se obtenham os dados é realizado o caminho inverso até à componente “*Routes*”. Na componente “*Routes*”, os dados recebidos são transformados no formato *JSON* e devolvidos em resposta http Ok (200). Por fim, essa resposta é recebida na aplicação móvel e transformada numa tabela para que o cuidador consiga interpretá-la.

Um exemplo do resultado obtido do pedido é apresentado de seguida:


```
{
    "occurrences": [
        {
            "id": 19,
            "occurrence_type_id": 1,
            "patient_id": 1,
            "caregiver_id": 12,
            "healthcare_id": null,
            "date": "2018-06-02 20:27:38",
            "description": "tipo de ocorrencia 1 de cuidador; desmaio",
            "created_at": "2018-06-02 20:27:58",
            "updated_at": "2018-07-16 16:24:30"
        },
        {
            "id": 20,
            "occurrence_type_id": 1,
            "patient_id": 1,
            "caregiver_id": 12,
            "healthcare_id": null,
            "date": "2018-06-02 20:27:38",
            "description": "tipo de ocorrencia 1 de cuidador; segundo
banho",
            "created_at": "2018-06-02 20:28:19",
            "updated_at": "2018-07-16 16:25:46"
        },
        {
            "id": 21,
            "occurrence_type_id": 4,
            "patient_id": 1,
```

```
            "caregiver_id": null,
            "healthcare_id": 16,
            "date": "2018-06-03 14:47:39",
            "description": "tipo de ocorrencia 1 de profissional de saúde;
liga",
            "created_at": "2018-06-03 14:48:02",
            "updated_at": "2018-07-16 16:28:01"
        },
        {
            "id": 22,
            "occurrence_type_id": 4,
            "patient_id": 1,
            "caregiver_id": null,
            "healthcare_id": 16,
            "date": "2018-06-05 14:15:54",
            "description": "tipo de ocorrencia 1 de profissional de saúde;
",
            "created_at": "2018-06-05 14:18:20",
            "updated_at": "2018-07-16 16:28:12"
        }
 ]
```


Código 2 – Exemplo de resultado da obtenção de ocorrências

Através do diagrama da Figura 13 consegue-se perceber todo o fluxo, desde o pedido do utilizador até à obtenção dos dados requeridos.

De seguida é exposto, em detalhe, vários excertos de código fonte para que este pedido de ocorrências seja realizado.

```
Route::get('occurrence', function() {
      $occurrences=Occurrence::all();
      return response()->json(['occurrences' => $occurrences], 200);
});
```

Código 3 – Exemplo 1 de extrato de código da componente "*Routes*"

Como já foi referenciado, este componente recebe pedidos http. Neste extrato de código, o pedido deve obedecer às regras definidas: método Get e recurso "*occurrence*". Assim que o pedido cumpra estes requisitos, todo o código envolvido dentro da função é executado. É utilizado o modelo "*Occurrence*" para obter todos os dados das occorrências, sem filtro. Assim que se obtenham os dados, estes são transformados num *array* em formato *JSON*, num tipo de resposta http Ok (200).

Relativamente ao registo de uma nova ocorrência, o cuidador deve registar através da funcionalidade "Registar ocorrência", na aplicação móvel. Assim que o botão de submissão é pressionado, um conjunto de passos é efetuado. Na Figura 14 é representado esses passos:

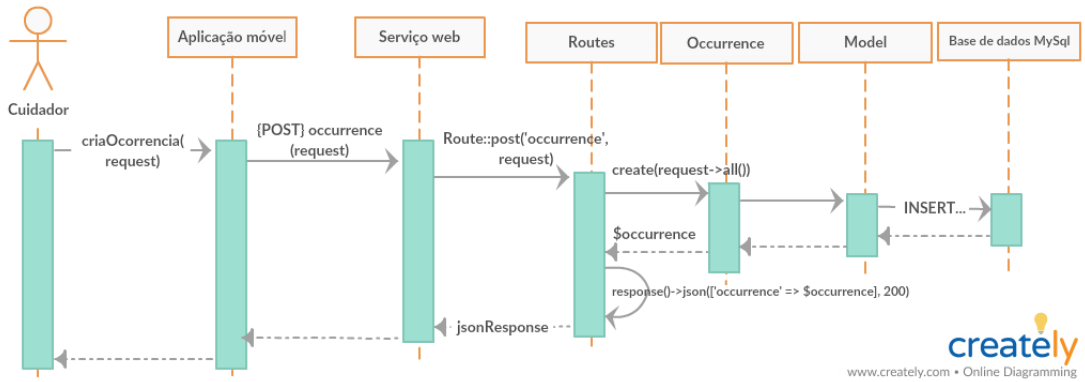


Figura 14 - Diagrama de sequência criação de ocorrência

O primeiro passo da aplicação móvel é chamar o serviço web, efetuando um pedido http, através do método Post, ao recurso "*occurrence*". O *link* do pedido é o mesmo da obtenção de ocorrências, http://193.136.62.28:8080/occurrence, em que as únicas diferenças estão no método a utilizar e no corpo do pedido, que deve conter os dados relativos a uma ocorrência:

```
"Method" : "POST",
"Headers" : [{
      "Content-Type" : "application/json"
}],
"Body" : {
    "occurrence_type_id": 1,
    "patient_id": 1,
    "caregiver_id": 12,
    "healthcare_id": null,
    "date": "2018-10-10 20:27:38",
    "description": "tipo de ocorrencia 1 de cuidador; desmaio"
}
```

Código 4 – Exemplo de pedido http, método Post

Como já foi descrito, todos os recursos encontram-se na componente "*Routes*", que consoante o método do pedido e o nome do recurso reencaminha para a rota respetiva. A rota "*occurrence*" tem associada a si a sua própria classe modelo "*Occurrence*" e faz a chamada ao método *"create(request->all())"*, para registar uma ocorrência, através do corpo (*body*) do pedido. Por sua vez, a classe "*Occurrence*", como estende da classe "*Model*", efetua a mesma chamada ao método *"create(request->all())"*. Logo, é esta classe que regista os dados relativos à ocorrência na base de dados. Assim que a inserção for positiva é realizado o caminho inverso até à componente "*Routes*". Na componente "*Routes*", é obtido a ocorrência registada na base de dados e esta é transformada no formato *JSON* e devolvida em resposta http Ok (200). Por fim, essa resposta é recebida na aplicação móvel e é apresentada como registada.

Um exemplo do resultado obtido do pedido é apresentado de seguida:

```
{
    "occurrence": {
        "occurrence_type_id": 1,
        "patient_id": 1,
        "caregiver_id": 12,
        "healthcare_id": null,
        "date": "2018-10-10 20:27:38",
        "description": "tipo de ocorrencia 1 de cuidador; desmaio",
        "updated_at": "2018-10-10 20:48:13",
        "created_at": "2018-10-10 20:48:13",
        "id": 35
    }
}
```

Código 5 – Exemplo de resultado da inserção de uma ocorrência

Através do diagrama da Figura 14 consegue-se perceber todo o fluxo, desde o pedido do utilizador até ao registo da ocorrência.

De seguida é exposto, em detalhe, vários excertos de código fonte para que este pedido de registo de ocorrência seja realizado.

```
Route::post('occurrence', function(Request $request) {
       $occurrences=Occurrence::create($request->all());
       return response()->json(['occurrence' => $occurrence], 200);
});
```

Código 6 – Exemplo 2 de extrato de código da componente "*Routes*"

Como já foi referenciado, este componente recebe pedidos http. Neste extrato de código, o pedido deve obedecer às regras definidas: método Post, recurso "*occurrence*" e receber um corpo de pedido correspondente aos dados de uma ocorrência. Assim que o pedido cumpra estes requisitos, todo o código envolvido dentro da função é executado. É utilizado o modelo "*Occurrence*" para registar a occorrência. Assim que a ocorrência for inserida na base de dados, a mesma é devolvida e transformada num objeto em formato *JSON*, num tipo de resposta http Ok (200).

4.3.2 Registo e acesso a intervenções e tipificação das mesmas

Em termos de estrutura de pedido e receção da informação, o registo e acesso a intervenções é tudo semelhante ao registo e acesso a ocorrências. Os únicos critérios que os diferenciam são, para além do nome da rota ser "*intervention*", os dados enviados e recebidos cingem a uma estrutura de uma intervenção.

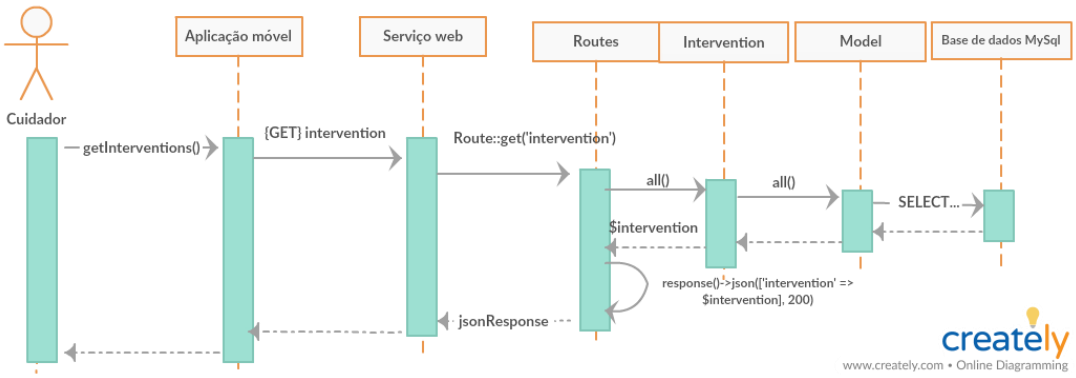


Figura 15 - Diagrama de sequência acesso a intervenções

Utilizando o exemplo 1 de pedido http, método Get, a única diferença está no *link* do pedido: http://193.136.62.28:8080/intervention. Os passos seguidos para obtenção das intervenções são os mesmos da obtenção das ocorrências.

Um exemplo do resultado obtido do pedido é apresentado de seguida:


```
{
    "intervention": [
        {
            "id": 10,
            "intervention_type_id": 1,
            "patient_id": 1,
            "caregiver_id": 12,
            "healthcare_id": null,
            "date": "2018-05-21 21:46:54",
            "description": "intervenção 1",
            "created_at": "2018-05-22 19:16:14",
            "updated_at": "2018-05-22 19:16:14"
        },
        {
            "id": 14,
            "intervention_type_id": 3,
            "patient_id": 1,
            "caregiver_id": 12,
            "healthcare_id": null,
            "date": "2018-05-21 21:46:54",
            "description": "teste",
            "created_at": "2018-05-25 21:10:11",
            "updated_at": "2018-05-25 21:10:11"
        }
    ]
}
```


Código 7 – Exemplo de resultado da obtenção de intervenções

De seguida é exposto, em detalhe, vários excertos de código fonte para que este pedido de ocorrências seja realizado.

```
Route::get('patient', function() {
```

```
        $patients=Patient::all();
        return response()->json([‘patients’ => $patients], 200);
});
```

Código 8 – Exemplo 3 de extrato de código da componente “*Routes*”

A única diferença neste excerto de código é a chamada ao modelo “*Patient*”, que referencia o paciente na base de dados.

Relativamente ao registo de uma nova intervenção, o cuidador deve registar através da funcionalidade “Registar intervenção”, na aplicação móvel. Tendo em conta que o processo em si, é semelhante, o conjunto de passos são equivalentes à submissão de uma ocorrência.

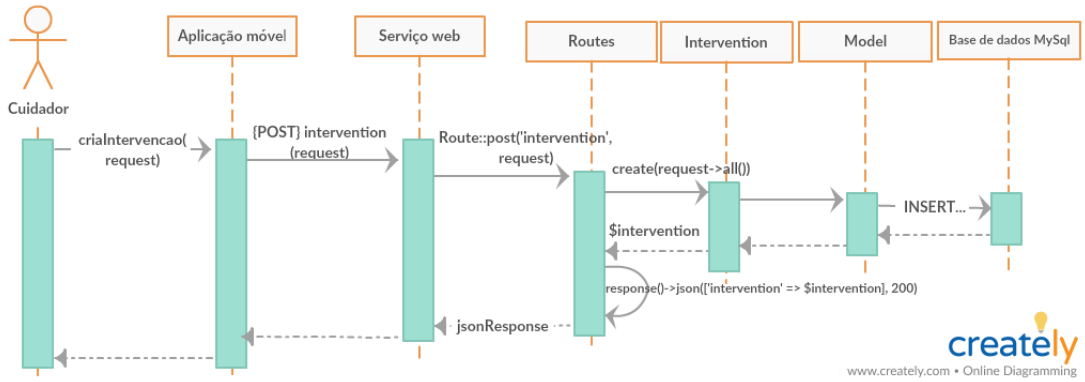


Figura 16 - Diagrama de sequência criação de intervenção

Utilizando o exemplo 4 de pedido http, método Post, a única diferença está no *link* do pedido: http://193.136.62.28:8080/intervention. Os passos seguidos para registo de uma intervenção é o mesmo do registo de uma ocorrência. O *body* do pedido pode ser semelhante a este:

```
“Body” : {
    "intervention_type_id": 1,
    "patient_id": 1,
    "caregiver_id": 12,
    "healthcare_id": null,
    "date": "2018-10-10 10:06:54",
    "description": "teste de intervenção"
}
```

Código 9 – Exemplo de pedido http, método Post

Um exemplo do resultado obtido do pedido é apresentado de seguida:

```
{
    "intervention": {
        "intervention_type_id": 1,
        "patient_id": 1,
        "caregiver_id": 12,
        "healthcare_id": null,
```

```
            "date": "2018-10-10 10:06:54",
            "description": "teste de intervenção",
            "updated_at": "2018-10-10 10:52:33",
            "created_at": "2018-10-10 10:52:33",
            "id": 43
        }
}
```

Código 10 – Exemplo de um resultado da inserção de uma ocorrência

De seguida é exposto, em detalhe, vários excertos de código fonte para que este pedido de registo de ocorrência seja realizado.

```
Route::post('intervention', function(Request $request) {
        $intevention=Intervention::create($request->all());
        return response()->json(['intevention' => $intervention], 200);
});
```

Código 11 – Exemplo 4 de extrato de código da componente "*Routes*"

As tipificações de intervenção foram desenvolvidas para suporte à inserção de uma intervenção. No entanto, não estão configuradas para serem acedidas através da aplicação móvel, visto que são funcionalidades que estarão disponíveis nas tipificações da aplicação de *backoffice*.

No seguimento das intervenções, as tipificações de intervenção funcionem da mesma forma, como na obtenção de resultados, como no seu registo.

De seguida é exposto, em detalhe, vários excertos de código fonte para o pedido de registo de tipificação de intervenção e para a obtenção das intervenções.

```
Route::get('interventiontype ', function() {
        $inteventiontypes = InterventionType::all();
        return response()->json(['interventiontypes' => $inteventiontypes],
        200);
});

Route::post('interventiontype', function(Request $request) {
        $inteventiontype=InterventionType::create($request->all());
        return response()->json(['inteventiontype' => $interventiontype],
        200);
});
```

Código 12 – Exemplo 5 de extrato de código da componente "*Routes*"

4.3.3 Registo e acesso à informação básica do paciente

O acesso à informação do paciente, em comparação com os acessos a ocorrências e intervenções, tem a particularidade de utilizar um identificador específico de um determinado

paciente. Normalmente, quando se utiliza um identificador, este será o campo *id*, campo identificativo na base de dados, também conhecido por chave primária.

De forma a visualizar a informação básica de um paciente, o cuidador deve solicitar na aplicação móvel o menu respetivo. Assim que esse menu é acionado, um conjunto de passos é efetuado. Na Figura 17 é representado esse conjunto de passos.

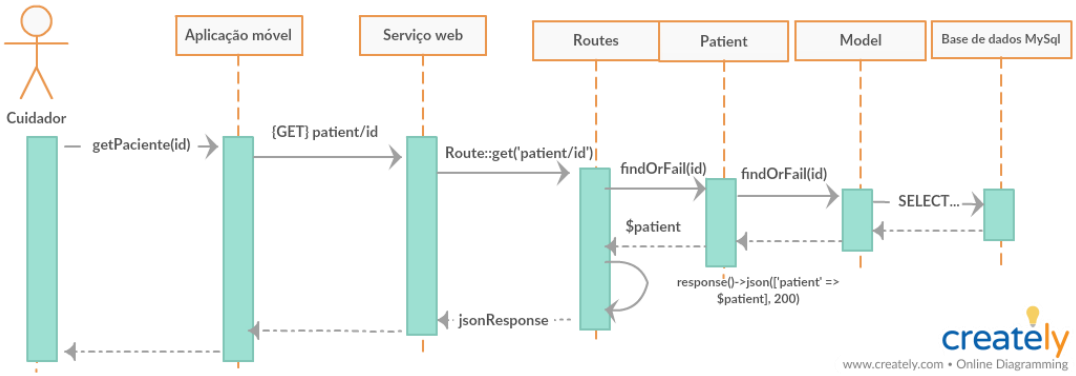


Figura 17 - Diagrama de sequência acesso a informação básica do paciente

Utilizando o exemplo 1 de pedido http, método Get, a única diferença está no *link* do pedido: http://193.136.62.28:8080/patient/id, em que 193.136.62.28:8080 representa o *IP* da máquina em que o serviço web está alojado e *id* representa o identificador do paciente. Os passos seguidos para obtenção da informação básica são semelhantes às ocorrências e intervenções.

Todos os recursos encontram-se na componente "*Routes*", que consoante o método do pedido e o nome do recurso reencaminha para a rota respetiva. A rota "*patient*" tem associada a si a sua própria classe modelo "*Patient*" e faz a chamada ao método *"findOrFail(id)"*, para obter a informação de um determinado paciente, com o filtro *id*. Por sua vez, a classe "*Patient*", como estende da classe "*Model*", efetua a mesma chamada ao método *"findOrFail(id)"*. Logo, é esta classe que obtém os dados relativos dos pacientes, que estão registadas na base de dados. Assim que se obtenham os dados de um paciente é realizado o caminho inverso até à componente "*Routes*". Na componente "*Routes*", os dados recebidos são transformados no formato *JSON* e devolvidos em resposta http Ok (200).

Um exemplo do resultado obtido do pedido http://193.136.62.28:8080/patient/1 é apresentado de seguida:

```
{
    "patient": {
        "id": 1,
        "name": "paciente1",
        "identification_number": 2147483647,
        "date_of_birth": "1958-05-17",
        "phone": "910920394",
        "email": "paciente1@gmail.com",
        "address": "morada1",
        "postal_code": "codigo1",
        "city": "cidade1",
        "country": "pais1",
        "active": 1,
        "created_at": "2018-05-18 12:45:48",
        "updated_at": "2018-05-18 12:45:48"
    }
}
```

Código 13 – Exemplo de resultado da obtenção da informação básica do paciente 1

De seguida é exposto, em detalhe, vários excertos de código fonte para que este pedido de ocorrências seja realizado.

```
Route::get('patient/{id}', function($id) {
       $patient=Patient::findOrFail($id);
       return response()->json(['patient' => $patient], 200);
});
```

Código 14 – Exemplo 6 de extrato de código da componente "*Routes*"

Como já foi referenciado, este componente recebe pedidos http. Neste extrato de código, o pedido deve obedecer às regras definidas: método Get e recurso "*patient/{id}*". Assim que o pedido cumpra estes requisitos, todo o código envolvido dentro da função é executado. É utilizado o modelo "*Patient*" para obter a informação de um paciente, filtrando pelo seu identificador. Assim que se obtenham os dados, estes são transformados num objeto em formato *JSON*, num tipo de resposta http Ok (200).

De momento, apenas a pesquisa da informação básica de um paciente pelo seu identificador está ativa. No futuro prevê-se que a pesquisa seja efetuada pelo número de utente ou até por email.

Se, por acaso, o pedido a esta informação compreenda um identificador inválido ou inexistente, como, por exemplo, http://193.136.62.28:8080/patient/0, a resposta ao pedido será um objeto do tipo *error*, com o descritivo do erro associado:

```
{
    "error": "Resource not found"
}
```

Código 15 – Exemplo de resultado da obtenção da informação básica de uma paciente inválido

O processo de registo do paciente é tudo semelhante aos registos de ocorrências e intervenções. O pedido http é realizado pelo método Post, ao recurso "*patient*" O *body* do pedido pode ser semelhante a este:

```
"Body" : {
    "name": "paciente xpto",
    "identification_number": 99999999,
    "date_of_birth": "1958-05-17",
    "phone": "965412347",
    "email": "pacientexpto@gmail.com",
    "address": "morada xpto",
    "postal_code": "codigo xpto",
    "city": "cidade xpto",
    "country": "pais xpto",
    "active": 1
}
```

Código 16 – Exemplo de pedido http, método Post

Um exemplo do resultado obtido do pedido é apresentado de seguida:

```
{
    "patient": {
        "name": "paciente xpto",
        "identification_number": 99999999,
        "date_of_birth": "1958-05-17",
        "phone": "965412347",
        "email": "pacientexpto@gmail.com",
        "address": "morada xpto",
        "postal_code": "codigo xpto",
        "city": "cidade xpto",
        "country": "pais xpto",
        "active": 1,
        "updated_at": "2018-10-10 14:13:29",
        "created_at": "2018-10-10 14:13:29",
        "id": 19
    }
}
```

Código 17 – Exemplo de um resultado da inserção de um paciente

De seguida é exposto, em detalhe, vários excertos de código fonte para que este pedido de registo de ocorrência seja realizado.

```
Route::post('patient', function(Request $request) {
       $patient=Patient::create($request->all());
       return response()->json(['patient => $patient], 200);
});
```

Código 18 – Exemplo 6 de extrato de código da componente "*Routes*"

4.3.4 Registo e acesso a tratamentos de saúde e tipificação das mesmas

Como os princípios dos pedidos das rotas do serviço são os mesmos, o registo, acesso a tratamentos de saúde e tipificação seguem a mesma estrutura das funcionalidades de intervenções. Nesta secção são mostradas apenas as diferenças que prevalecem entre os pedidos de tratamento e os pedidos de intervenção. Na Figura 18 é representado o diagrama de sequência do acesso a tratamentos de saúde.

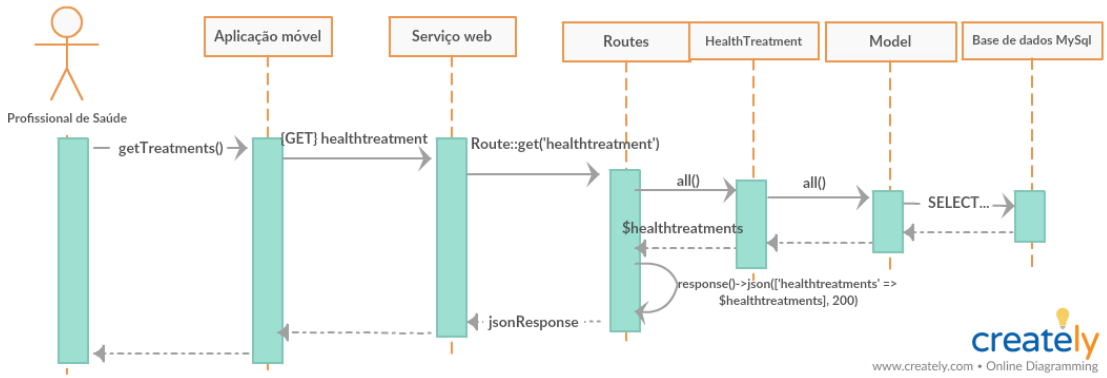


Figura 18 - Diagrama de sequência acesso a tratamentos de saúde

O nome do recurso é "*healthtreatment*"(http://193.136.62.28:8080/healthtreatment), em que o pedido http utiliza o método Get. A diferença aqui neste pedido é que o mesmo é realizado pelo profissional de saúde, porque é o único tipo de utilizador que tem acesso a este menu.

Um exemplo do resultado obtido do pedido é apresentado de seguida:


```
{
    "health_treatment": [
        {
            "id": 3,
            "health_treatment_type_id": 2,
            "patient_id": 1,
            "healthcare_id": 16,
            "date": "2018-06-03 12:25:00",
            "description": "Tratamento de Saúde x",
            "created_at": "2018-06-03 11:25:50",
            "updated_at": "2018-06-03 11:25:50"
        },
        {
            "id": 4,
            "health_treatment_type_id": 2,
            "patient_id": 1,
            "healthcare_id": 16,
            "date": "2018-06-03 14:31:26",
            "description": "Tipo de tratamento de saúde 2; Observação 2"
            "created_at": "2018-06-03 14:31:46",
```

```
            "updated_at": "2018-06-03 14:31:46"
        },
        {
            "id": 5,
            "health_treatment_type_id": 2,
            "patient_id": 1,
            "healthcare_id": 16,
            "date": "2018-06-03 14:47:17",
            "description": "Tipo de tratamento de saúde 2; Observação 2”
            "created_at": "2018-06-03 14:48:02",
            "updated_at": "2018-06-03 14:48:02"
        }
]
```

Código 19 – Exemplo de resultado da obtenção de tratamentos de saúde

De seguida é exposto, em detalhe, vários excertos de código fonte para que este pedido de ocorrências seja realizado.

```
Route::get(‘healthtreatment‘, function() {
      $healthtreatments=HealthTreatment::all();
      return response()->json([‘healthtreatments’ => $healthtreatments],
     200);
});
```

Código 20 – Exemplo 7 de extrato de código da componente “*Routes*”

Relativamente ao registo de tratamento de saúde, um profissional de saúde com acesso à aplicação de escolher o menu respetivo para o efeito. Na Figura 19 é ilustrado os passos que são realizados para efetuar o pedido.

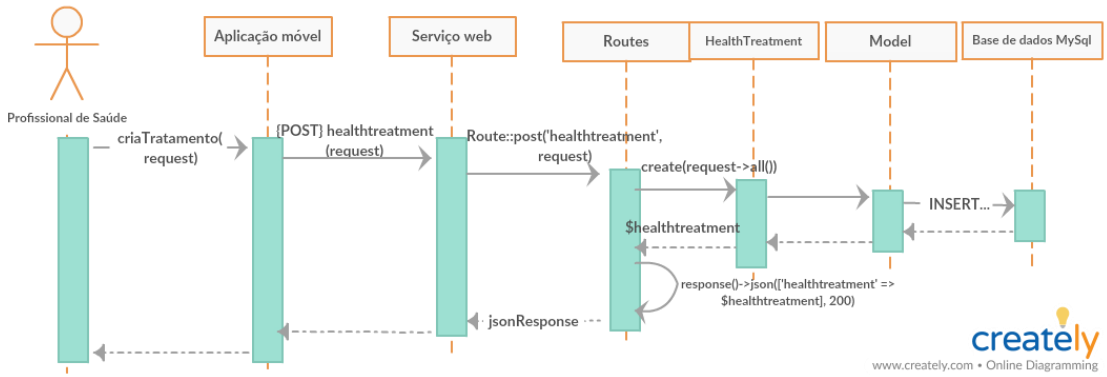


Figura 19 - Diagrama de sequência de registo de tratamento de saúde

O pedido http é realizado pelo método Post, ao recurso “*healthtreatment*” O *body* do pedido pode ser semelhante a este:

```
“Body” : {
    "health_treatment_type_id": 1,
    "patient_id": 1,
    "healthcare_id": 16,
```

```
    "date": "2018-10-10 09:30:25",
    "description": "teste de tratamento"
}
```

Código 21 – Exemplo de pedido http, método Post

Um exemplo do resultado obtido do pedido é apresentado de seguida:

```
{
    "health_treatment": {
        "health_treatment_type_id": 1,
        "patient_id": 1,
        "healthcare_id": 16,
        "date": "2018-10-10 09:30:25",
        "description": "teste de tratamento",
        "updated_at": "2018-10-10 14:39:24",
        "created_at": "2018-10-10 14:39:24",
        "id": 10
    }
}
```

Código 22 – Exemplo de um resultado da inserção de um tratamento médico

De seguida é exposto, em detalhe, vários excertos de código fonte para que este pedido de registo de tratamento médico seja realizado.

```
Route::post('healthtreatment', function(Request $request) {
       $healthtreatment=HealthTreatment::create($request->all());
       return response()->json(['healthtreament' => $healthtreament], 200);
});
```

Código 23 – Exemplo 9 de extrato de código da componente "*Routes*"

O uso da ferramenta Laravel foi importante neste âmbito, para facilmente se conseguir definir as rotas pretendidas, bem como o acesso rápido à base de dados.

4.4 Aplicação web *backoffice*

Apesar de aplicação web de *backoffice* não estar concluída foi realizada uma previsão e aprovação do *layout* da mesma.

A página de entrada, depois do *login* efetuado, terá um aspeto semelhante à Figura 20.

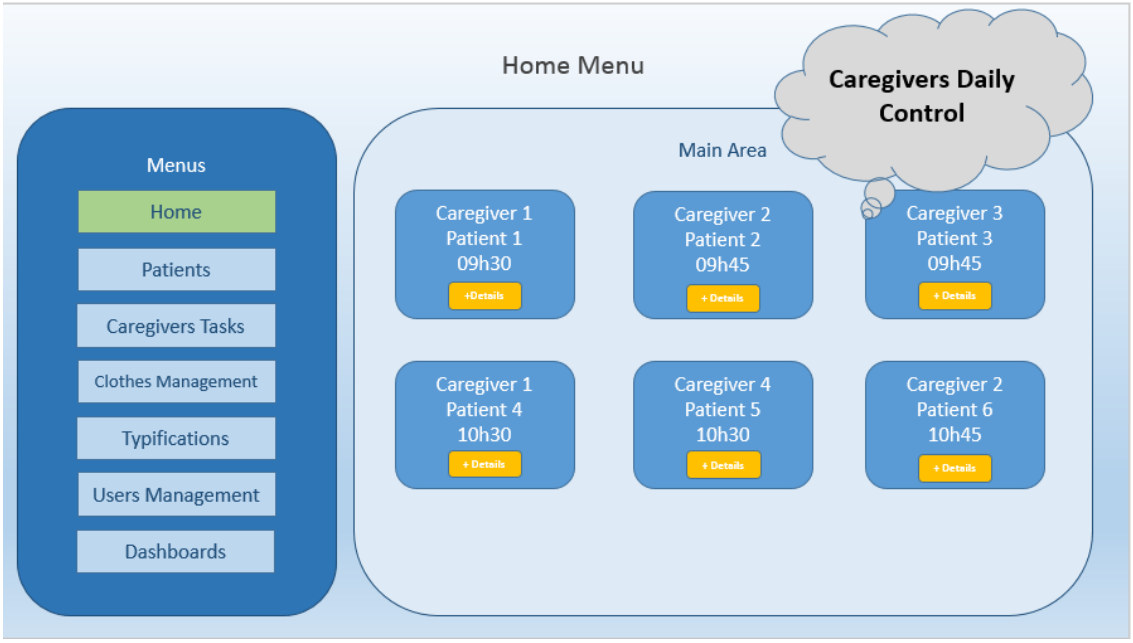


Figura 20 - Menu principal da aplicação *backoffice*

O que se pretende neste ecrã é que mostre para o coordenador da instituição uma vista geral do dia, ou seja, quais os cuidadores que estão a realizar vistas aos pacientes ou clientes, no momento. Em cada um, pode-se aceder aos detalhes do mesmo.

A Figura 21 ilustra a lista de pacientes ou clientes da instituição.

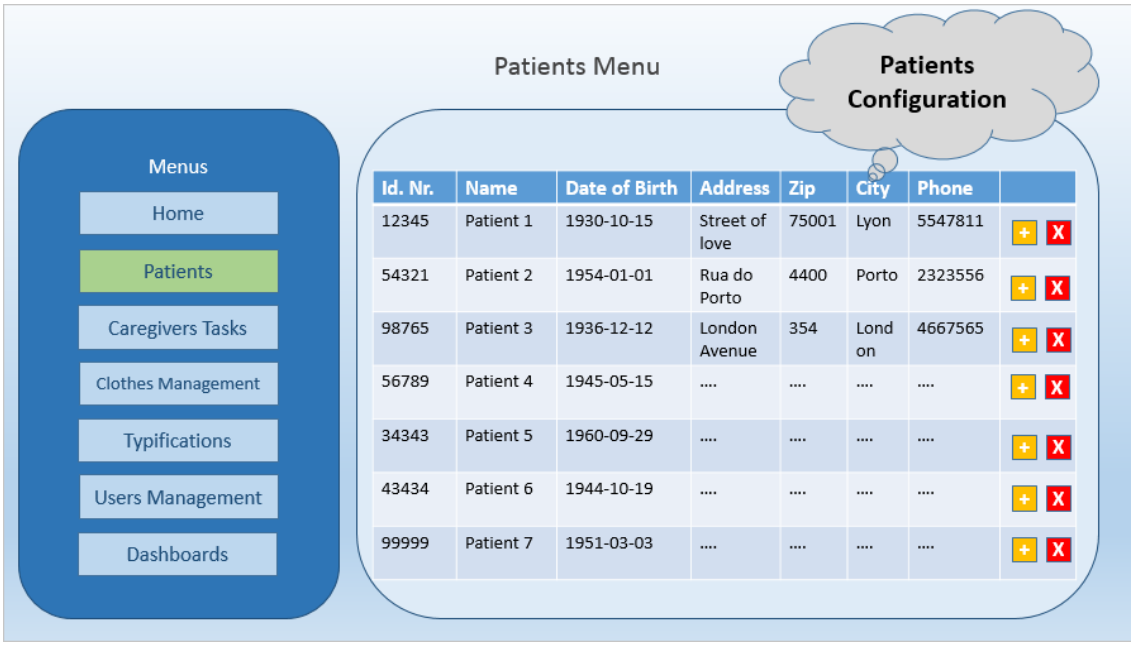


Figura 21 - Menu de pacientes

No presente menu, o coordenador tem acesso aos dados básicos de todos os pacientes. Pode configurá-los através dos botões à direita da listagem, ou acrescentar um novo, ou editar dados de um paciente ou eliminar.

O menu seguinte é relativo à configuração de tarefas para os cuidadores.

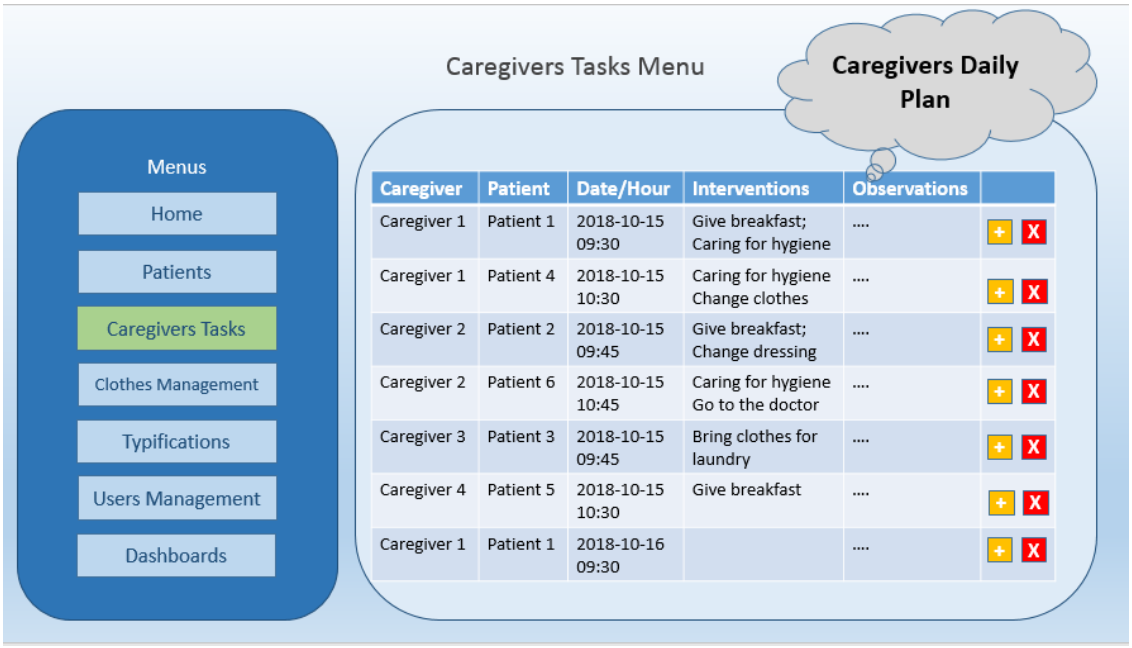


Figura 22 - Configuração de tarefas para cuidadores

Tal como o menu anterior, é possível acrescentar, editar ou eliminar. Neste caso, o contexto é relativo ao plano para cada cuidador.

Na Figura 23 é representado, em listagem, todas as roupas que foram levadas para as lavandrias ou roupas a trazidas da lavandaria, por cada cuidador, de cada paciente.

Figura 23 - Configuração de roupas

Para o coordenador é importante esta informação para controlar se há ou não perda de roupas. Pode configurar a informação através dos botões à direita.

O menu seguinte representa as tipificações associadas às funcionalidades da aplicação móvel.

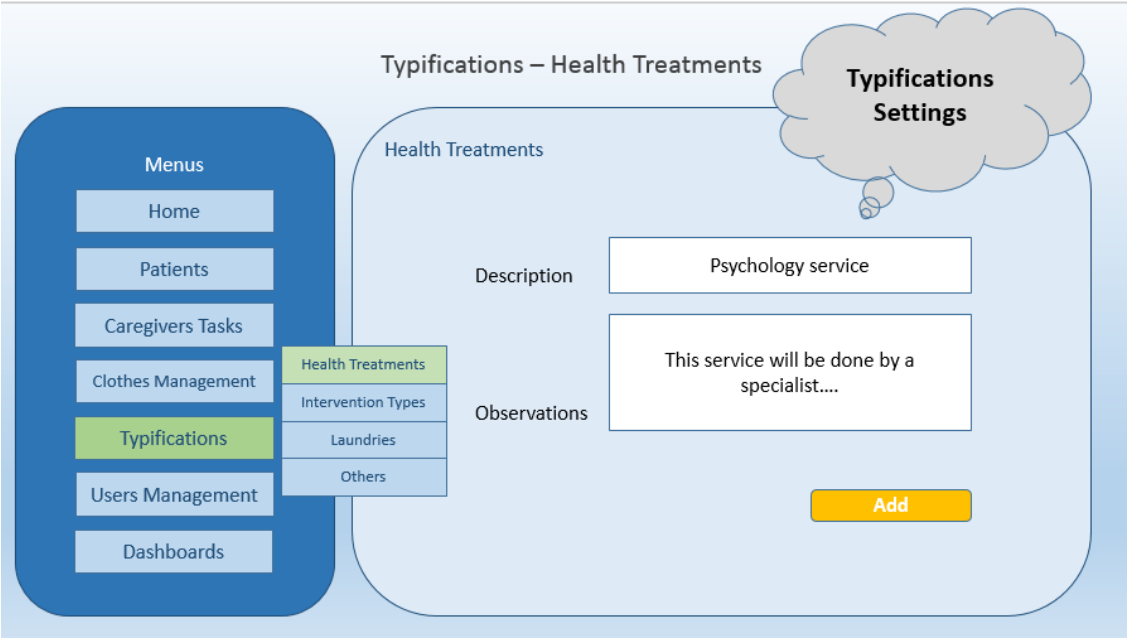


Figura 24 - Menu de tipificações

As tipificações podem ser relativas aos tratamentos de saúde para os profissionais de saúde conseguirem classificar o tipo de tratamento realizado ao paciente, o tipo de intevenções realizadas aos pacientes, pelos cuidadores, o registo de lavandarias que os cuidadores auferem, quando fazem o tratamento de roupa e outros tipos de tipificações.

Na figura 25 é representa a lista de utilizadores que têm acesso à aplicação móvel e à aplicação de *backoffice*.

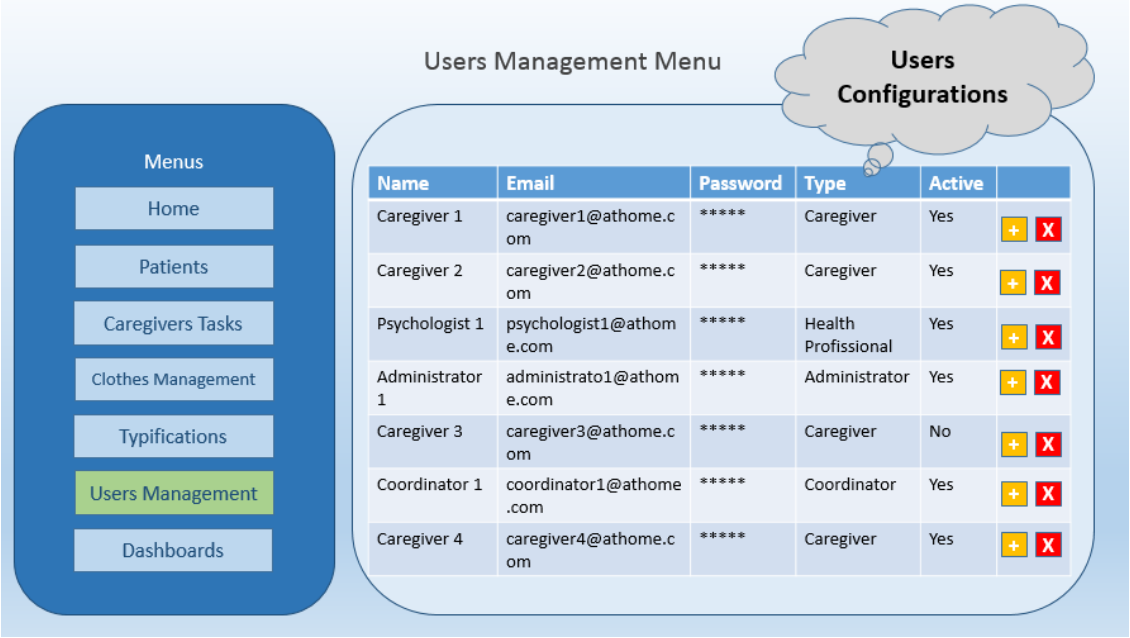


Figura 25 - Configuração de utilizadores

Neste ecrã é possível criar utilizadores e caracterizá-los em géneros, ou seja, na criação de um utilizador atribuir se o mesmo é um cuidador, ou um profissional de saúde, ou um coordenador ou um administrador.

Por fim, o último menu representa várias estatísticas da informação do sistema.

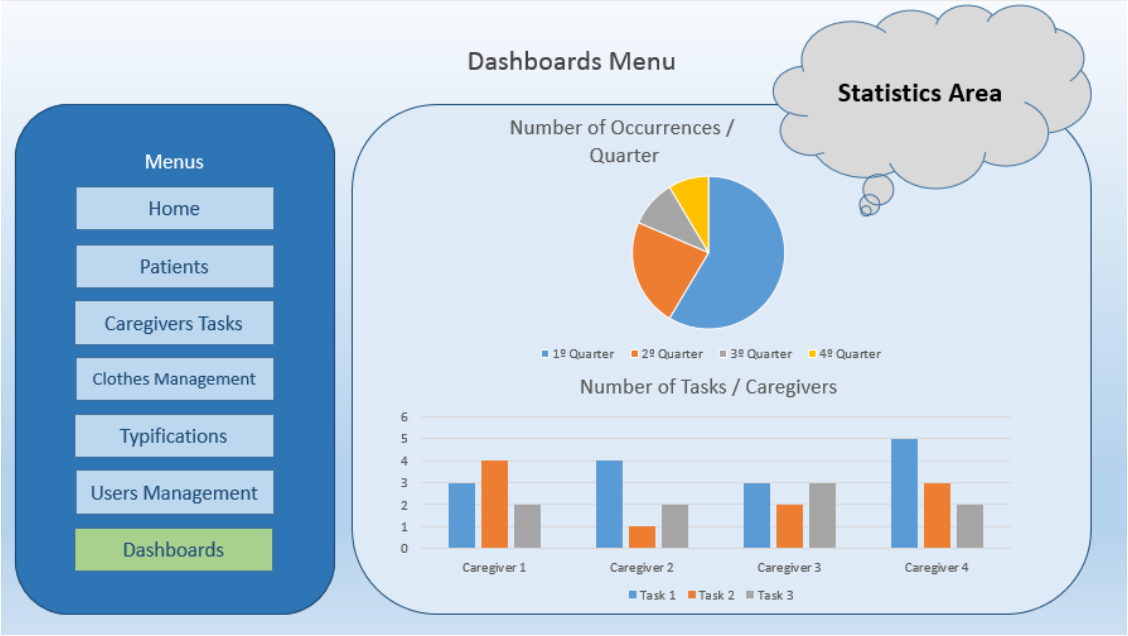


Figura 26 - Menu *dashboards*

Este menu é destinado aos administradores, onde podem averiguar várias informações sobre, por exemplo, número de ocorrências por trimestre ou a quantidade de tarefas de cada cuidador, entre outras.

Na Figura 27 é representada as permissões de acesso de determinados tipos de utilizador aos menus do *backoffice*.

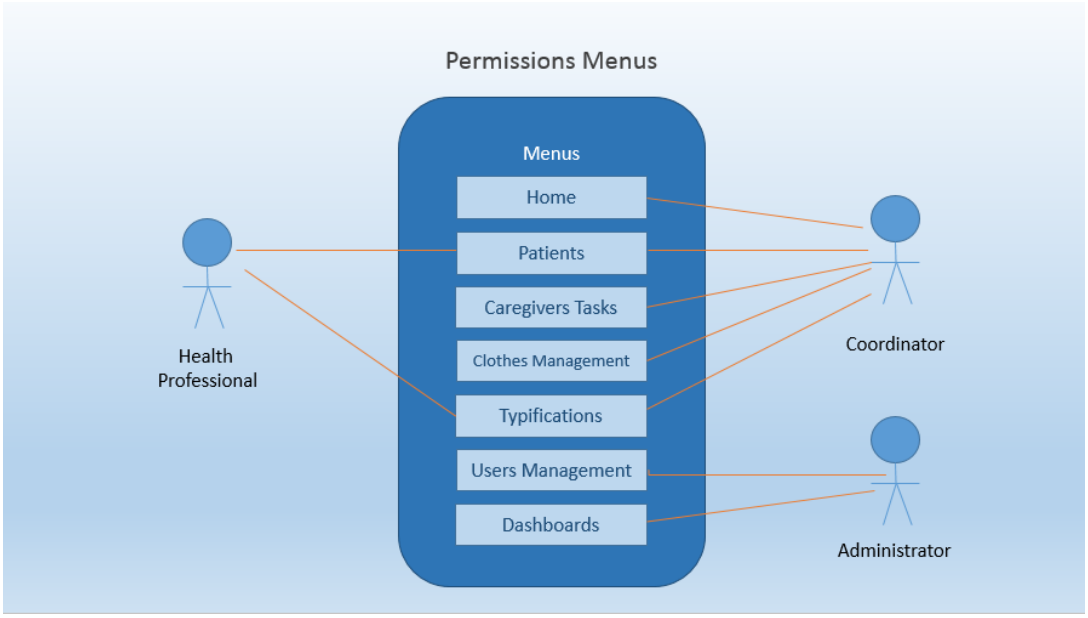


Figura 27 - Permissões de menus

O administrador é o único que aceder aos menus de gestão de utilizadores e estatísticas. O profissional de saúde pode ter acesso às tipificações relacionadas com tratamentos de saúde, bem como à informação dos pacientes. O tipo de utilizador que mais utilizará esta aplicação é o coordenador, que tem acesso aos restantes menus.

4.5 Padrões de *software*

Ao longo da realização do projeto foram identificados vários padrões de *software.* Para mais facilmente representar esta situação detetou-se vários princípios de atribuição de responsabilidade a objetos e, por consequência, verificou-se que ia ao encontro das diretrizes GRASP (Duncan, 2012). GRASP referencia 9 padrões de *software*:

- *Creator* – Classe que é responsável pela criação de objetos relaciona objetos;

Este princípio é o mais básico quando se utiliza uma linguagem orientada a objetos. A Figura 28 representa a criação de uma nova instância de uma aplicação Laravel:

```
$app = new Illuminate\Foundation\Application(
    realpath(__DIR__.'/../')
);
```

Figura 28 - Exemplo do padrão *Creator*

- *Information Expert* – Delegação de responsabilidades para fomentar a alta coesão e baixo acoplamento;

Para representar este princípio, a classe *DatabaseSeeder* é aquela que delega responsabilidades às classes "*HealthTypeTableSeeder*", "*PermissionTypeTableSeeder*" e "*UserTypeTableSeeder*", para executar as suas próprias operações à base de dados, através do método *"run()"* herdado da classe abstrata "*Seeder"*, tal como demonstra a Figura 29.

```php
class DatabaseSeeder extends Seeder
{
    /**
     * Run the database seeds.
     *
     * @return void
     */
    public function run()
    {
        $this->call(HealthTypeTableSeeder::class);
        $this->call(PermissionTypeTableSeeder::class);
        $this->call(UserTypeTableSeeder::class);
        //$this->call(ClothingTreatmentStatusSeeder::class);
    }
}
```

Figura 29 - Exemplo do padrão *Information Expert*

- *Low Coupling* – Baixo acoplamento é a fraca dependência entre objetos. Quanto mais baixo for mais é possível reutilizar os objetos;

O baixo acoplamento reproduzido nas rotas do serviço web consiste na independência que as rotas devem ter perante os respetivos domínios. Na Figura 30 espelha a utilização do domínio utilizador "User" em que, consoante o tipo da rota, utiliza operações distintas do domínio.

```php
Route::get('user', function () {
    $user = User::all();
    return response()->json(['user' => $user], 200);
});

Route::get('user/{id}', function ($id) {
    $user = User::findOrFail($id);
    return response()->json(['user' => $user], 200);
});

Route::post('user', function(\Illuminate\Http\Request $request) {
    $user = User::create($request->all());
    return response()->json(['user' => $user], 200);
});

Route::put('user/{id}', function(\Illuminate\Http\Request $request, $id) {
    $user = User::findOrFail($id);
    $user->update($request->all());

    return response()->json(['user' => $user], 200);
});

Route::delete('user/{id}', function($id) {
    $user = User::findOrFail($id)->delete();

    return response()->json(['user' => 'deleted'], 200);
});
```

Figura 30 - Exemplo do padrão *Low Coupling*

- *Controller* – Controlo na manipulação das operações do sistema. Faz a delegação das atividades entre a interface de utilizador e a lógica de negócio;

Como exemplo na utilização deste princípio foi criada uma classe de autenticação "*AuthController*", que realiza o controlo de autenticação nas chamadas a rotas do serviço Web At-Home, ou seja, é o mediador que valida se o *token* de autenticação do serviço é ou não válido. A Figura 31 representa esse cenário.

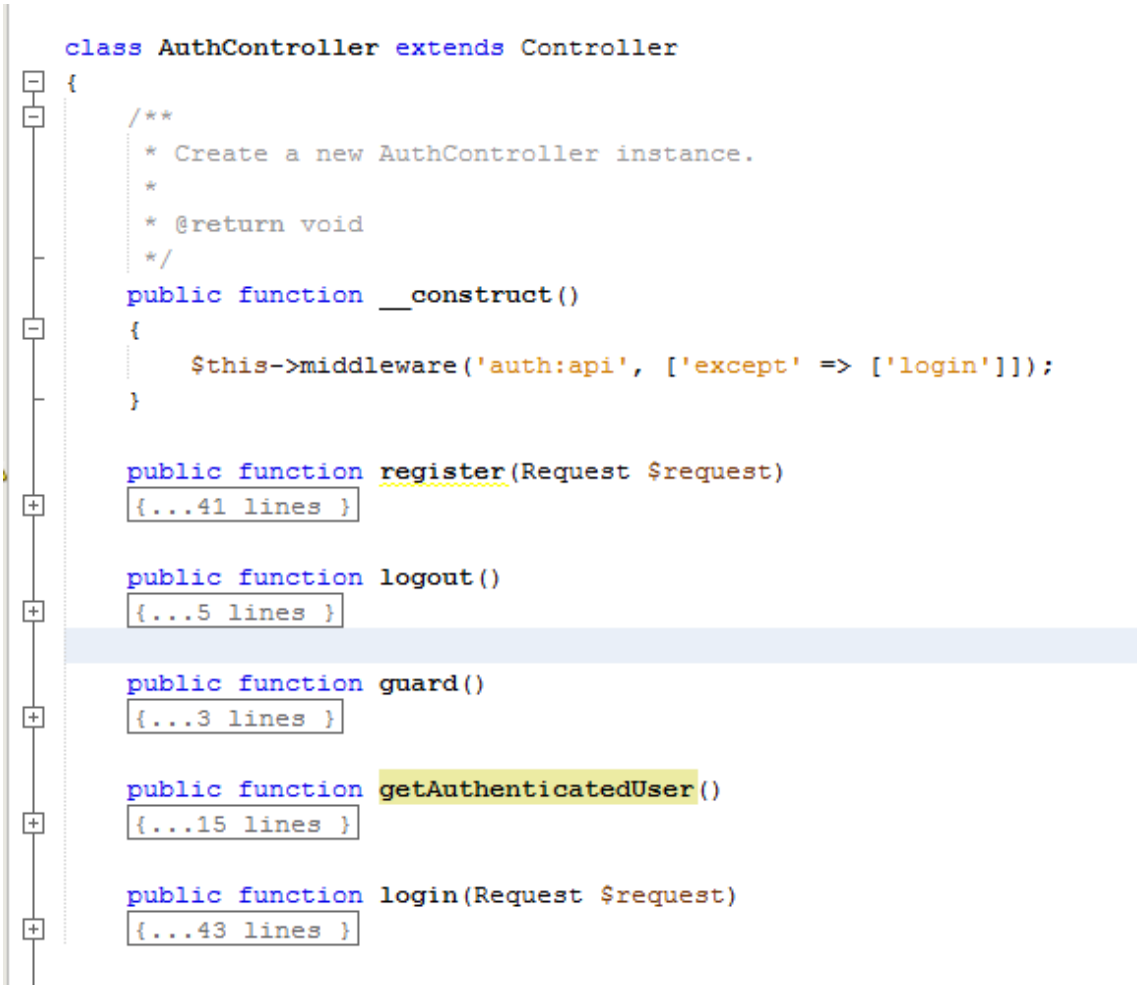


Figura 31 - Exemplo do padrão *Controller*

- *High Cohesion* – Semelhante ao baixo acoplamento (*low coupling*). Existe uma forte relação e foco nos objetos, com poucas relações entre outros objetos;

Na Figura 32, as propriedades utilizadas na classe *Controller* garantem alta coesão, devido à sua utilização ser restringida na própria classe.

```php
abstract class Controller
{
    /**
     * The middleware registered on the controller.
     *
     * @var array
     */
    private $middleware = [];

    /**
     * The router instance.
     *
     * @var \Illuminate\Routing\Router
     */
    private static $router;

    /**
     * Register middleware on the controller.
     *
     * @param  array|string  $middleware
     * @param  array   $options
     * @return \Illuminate\Routing\ControllerMiddlewareOptions
     */
    public function middleware($middleware, array $options = [])
    {
        foreach ((array) $middleware as $middlewareName) {
            $this->middleware[$middlewareName] = &$options;
        }
```

Figura 32 - Exemplo do padrão *High Cohesion*

- *Polymorphism* – Polimorfismo é a variação dos comportamentos das operações a realizar;

Existem vários exemplos deste princípio no serviço web, nomeadamente na criação de tabelas tipo da base de dados. Todas elas utilizam o método *"run()"*, que é herdado através de uma classe abstrata "*Seeder*" e que realiza operações à base de dados. Diversos exemplos de classes que utilizam este princípio são o tipo de tratamento médico "*HealthTreatmentType*", e o tipo de utilizador "*UserTypeTableSeeder*", demonstrados nas Figuras 33 e 34:

```php
class UserTypeTableSeeder extends Seeder
{
    /**
     * Run the database seeds.
     *
     * @return void
     */
    public function run()
    {
        UserType::create([
            'description' => 'SuperAdministrator',
        ]);

        UserType::create([
            'description' => 'Administrator',
        ]);

        UserType::create([
            'description' => 'Coordinator',
        ]);

        UserType::create([
            'description' => 'HealthProfissional',
        ]);

        UserType::create([
            'description' => 'Caregiver',
        ]);
    }
}
```

Figura 33 - Exemplo 1 do Padrão Polymorphism

```php
class HealthTypeTableSeeder extends Seeder
{
    /** Run the database seeds ...5 lines */
    public function run()
    {
        HealthTreatmentType::create([
            'description' => 'Serviço de psicologia',
            'observations' => null
        ]);

        HealthTreatmentType::create([
            'description' => 'Medicação',
            'observations' => null
        ]);
    }
}
```

Figura 34 - Exemplo 2 do padrão Polymorphism

- *Pure Fabrication* – "Fábrica" é uma classe artificial que não deve representar um conceito de domínio para contemplar o baixo acoplamento, alta coesão e reutilização;

A melhor representação deste princípio é a classe *Seeder,* que é caracterizada por ser um "serviço" orientado ao domínio da base de dados:

```php
abstract class Seeder
{
    /** The container instance ...5 lines */
    protected $container;

    /** The console command instance ...5 lines */
    protected $command;

    /** Run the database seeds ...5 lines */
    abstract public function run();

    /** Seed the given connection from the given path ...6 lines */
    public function call($class)
    {
        $this->resolve($class)->run();

        if (isset($this->command)) {
            $this->command->getOutput()->writeln("<info>Seeded:</info> $class");
        }
    }

    /** Resolve an instance of the given seeder class ...6 lines */
    protected function resolve($class)
    {
        if (isset($this->container)) {
            $instance = $this->container->make($class);

            $instance->setContainer($this->container);
        } else {
```

Figura 35 - Exemplo do padrão *Pure Fabrication*

- *Indirection* – Mecanismo de indireção para dar uma responsabilidade a outro objeto entre componentes ou serviços para permitir o baixo acuplmaneto;

Este princípio de Indireção está representado de igual forma no exemplo do princípio *Controller.*

- *Protected Variations* – Protege variações dos objetos, utilizando interfaces e polimorfismo.

O exemplo deste princípio pode ser revisto nos princípios *Pure Fabrication* e *Polymorphism.*

5 Avaliação

A avaliação do projeto, como um todo não foi efetuada em tempo útil, devido a atrasos nos tempos programados. Até ao momento foram realizados alguns testes de *software* ao serviço web da aplicação móvel, através de testes unitários, de integração e desistema. Posto isto, no final do capítulo é previsto de como se iria efetuar os restantes testes ao projeto e a sua respetiva avaliação.

5.1 Testes

Os testes do serviço web implementado estão projetados em cinco vertentes:

- Testes unitários a partir da implementação;
- Testes de integração a partir da arquitetura;
- Testes de sistema a partir da análise;
- Testes de aceitação a partir do documento de requisitos;
- Testes de usabilidade.

5.1.1 Testes Unitários

Para realizar os testes unitários ao serviço web foi necessário utilizar o *PHPUnit*, ferramenta de testes da linguagem PHP, que já vem associada ao projeto *Laravel*. Visto que o estilo

arquitetural seguido foi o *REST,* os pedidos de teste realizados são no formato *JSON* (*JavaScript Object Notation*), formato de dados simples para troca de mensagens entre sistemas.

O primeiro teste unitário realizado foi ao diretório associado à migração e *seed* da base de dados, para verificar se efetivamente a base de dados foi ou não criada. Para isso foi adicionada uma nova configuração para uma nova base de dados de teste, no ficheiro de configuração de base de dados do *Laravel*:

```
'connections' => [

    'unittests' => [
        'driver' => 'mysql',
        'host' => 'localhost',
        'port' => '3306',
        'database' => 'athometests',
        'username' => [redacted]
        'password' => [redacted]
        'charset' => 'utf8',
        'collation' => 'utf8_unicode_ci',
        'prefix' => '',
        'strict' => false,
        'engine' => null,
    ],

    'mysql' => [
```

Figura 36 - Configuração base de dados de teste

De seguida, teve que associada esta nova configuração no ficheiro PHPUnit:

```
<php>
    <env name="APP_ENV" value="testing"/>
    <env name="CACHE_DRIVER" value="array"/>
    <env name="SESSION_DRIVER" value="array"/>
    <env name="QUEUE_DRIVER" value="sync"/>
    <env name="DB_CONNECTION" value="unittests"/>
</php>
</phpunit>
```

Figura 37 - Associação base de dados de teste

Por fim, é necessário configurar o caso de teste. Na Figura 38 é representada a chamada de suporte à migração e *seed* da base de dados, através do método *"setUp()"*.

```php
use Illuminate\Foundation\Testing\DatabaseMigrations;
use Illuminate\Foundation\Testing\TestCase as BaseTestCase;
use Illuminate\Support\Facades\Artisan;

class TestCase extends BaseTestCase
{
    use DatabaseMigrations;
    /** The base URL to use while testing the application ...5 lines */
    protected $baseUrl = 'http://localhost';

    /** Creates the application ...5 lines */
    public function createApplication()
    {...7 lines }

    public function setUp()
    {
        parent::setUp();
        Artisan::call('db:seed');
    }
}
```

Figura 38 - Configuração de caso de teste

Para testar efetivamente o caso de teste foi adicionada uma nova classe, que faz a verificação se os mesmos valores dos tipos de utilizador "*UserType*", que foram registados na nova base de dados são os mesmos dos do teste.

```php
use App\UserType;

class UserTypeTest extends TestCase
{
    /** A basic test example ...5 lines */
    public function testExample()
    {
        echo "############ START TEST UserType ##############" . PHP_EOL;
        //DB selection - User Type table
        $usersTypes = UserType::all();
        //Test scenario
        $usersTypesTest = array('SuperAdministrator','Administrator',
            'Coordinator','HealthProfissional','Caregiver');

        echo "compare if size of db data \"" . count($usersTypes). "\" is equal "
                . "than size of test array \"" . count($usersTypesTest)  ."\"" . PHP_EOL;
        $this->assertEquals(count($usersTypesTest), count($usersTypes));

        for ($i = 0; $i < count($usersTypesTest); $i ++)
        {
            echo "compare if \"" . $usersTypes[$i]->description. "\" is equal "
                    . "than \"" .$usersTypesTest[$i]."\"" . PHP_EOL;
            $this->assertEquals($usersTypesTest[$i], $usersTypes[$i]->description);
        }
        echo "############ END TEST UserType ##############" . PHP_EOL;
    }
}
```

Figura 39 - Exemplo de teste unitário de tipos de utilizador

Para comprovar isso, na Figura 40 mostra a classe *seeder*, que regista esses tipos de utilizador na base de dados.

```php
use App\UserType;

class UserTypeTableSeeder extends Seeder
{
    /** Run the database seeds ...5 lines */
    public function run()
    {
        UserType::create([
            'description' => 'SuperAdministrator',
        ]);

        UserType::create([
            'description' => 'Administrator',
        ]);

        UserType::create([
            'description' => 'Coordinator',
        ]);

        UserType::create([
            'description' => 'HealthProfissional',
        ]);

        UserType::create([
            'description' => 'Caregiver',
        ]);
    }
}
```

Figura 40 - Classe *seeder* do tipo de utilizador

Para provar que o teste é valido, é necessário executar o comando *"vendor\bin\phpunit",* tal como demonstra a Figura 41.

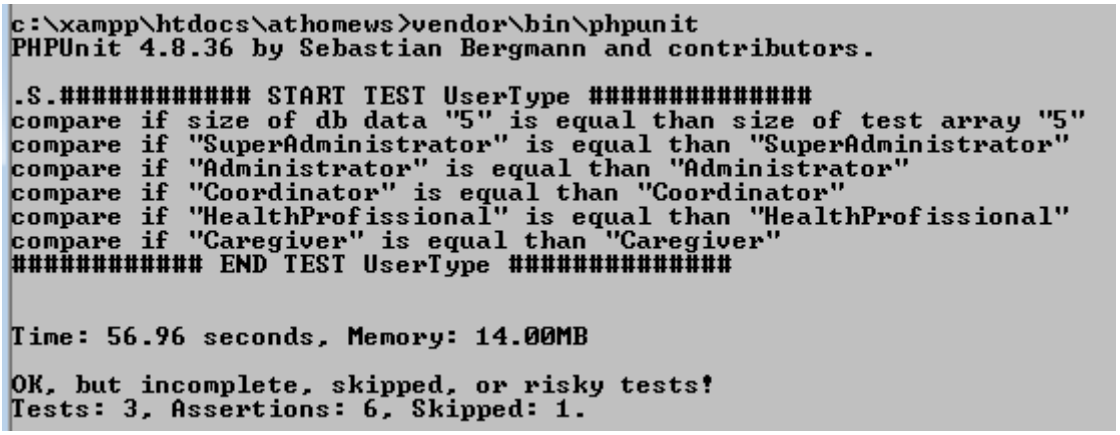


Figura 41 - Teste unitário tipo de utilizador

Este teste demorou 56 segundos a ser executado devido à criação completa da base de dados, bem como todas as inserções registadas, por defeito, na sua criação. O número de testes é 3, um deles é o teste que corre internamente do sistema, outro é o teste de registo de utilizador, que irá ser pormenorizado mais à frente nesta secção e o outro é efetivamente aquele que foi testado. O número de verificações efetuadas foram 6. A primeira está referênciada ao número de registos dos tipos de utilizador criados na base de dados, em comparação com o número de registos criados para teste. As restantes 5 são relativas aos valores dos registos que, por defeito, foram criados na base de dados, em comparação com os valores de teste.

Os próximos testes unitários efetuados foram testes às rotas da *API.* Realizaram-se testes aos tipos de resposta recebidos, nomeadamente aos códigos de estado Http 200 (Ok) e 422 (*Unprocessable Entity*), que significam que os pedidos foram aceites ou não respetivamente, pelas chamadas *POST* aos serviços de registo de utilizador e login, tal como demonstram as Figuras 42 e 43.

```php
echo "############ START TEST Register ##############" . PHP_EOL;

echo "compare if register is valid and returns HTTP 200 OK" . PHP_EC
echo "compare if register response shows the User object "
. "and the Auth token " . PHP_EOL;

$headers = [
    'Content-Type' => 'application/json',
    'Accept' => 'application/json'
];

$body = [
    'name' => 'Teste',
    'password' => 'teste123',
    'email' => 'teste@isep.ipp.pt',
    'user_type_id' => 1,
    'language_id' => 1,
    'active' => 1
];

$resp = $this->call('POST', 'register', $body, [], [], $headers);

$this->assertEquals(200, $resp->status());

$json = json_decode($resp->content(), true);
$this->assertArrayHasKey('user', $json);
$this->assertArrayHasKey('token', $json);

echo "############ END TEST Register ##############" . PHP_EOL;
```

Figura 42 - Teste unitário à chamada registo de utilizador

Esta configuração de teste verifica se o pedido de registo de utilizador, com os dados transmitidos no corpo (*body*) do pedido, é aceite pela *API.* A verificação realiza-se de duas formas:

- A resposta ao pedido deve devolver o estado Html 200 (Ok);
- O conteúdo da resposta *JSON* deve ter duas chaves: *user* e *token.* A chave *user* representa a entidade utilizador, com todos os campos a ela associados como, por exemplo, nome e email. A chave *token* traduz a entidade utilizador numa chave única e pessoal (*token* de autenticação).

```
echo "############ START TEST Login ##############" . PHP_EOL;

echo "compare if login request with no body data "
. "returns HTTP 422 error" . PHP_EOL;
echo "compare if login response shows the required fields " . PHP_EOL;

$this->post('login', [], $headers)->seeJson([
    'email' => ['The email field is required.'],
    'password' => ['The password field is required.'],
])->assertResponseStatus(422);

echo "compare if login is valid " . PHP_EOL;
echo "compare if login is valid and returns HTTP 200 OK" . PHP_EOL;

$body2 = [
    'email'      => 'teste@isep.ipp.pt',
    'password'   => 'teste123'
];

$resp2 = $this->call('POST', 'login', $body2, [], [], $headers);

$this->assertEquals(200, $resp2->status());

$json2 = json_decode($resp2->content(), true);
$this->assertArrayHasKey('token', $json2);

echo "############ END TEST Login ##############" . PHP_EOL;
```

Figura 43 - Teste unitário à chamada login

A configuração de teste de *login* faz duas verificações distintas. A primeira confirma se o pedido *login* efetuado, sem os parâmetros de entrada *email* e *password* não é valido. Para tal, a resposta é analisada de duas formas:

- Se a reposta devolvida, no formato *JSON*, contém duas chaves *email* e *password*, com os valores dos campos a serem requeridos – "*The email|password field is required*";
- Se o tipo do estado de resposta é 422 (*Unprocessable Entity*), que significa que, apesar de aceitar o pedido não o consegue processar, por falta de informação.

A segunda verificação é a aceitação de um *login* com dados corretos. A resposta a esse pedido é averiguada de duas formas:

- A resposta ao pedido deve devolver o estado Html 200 (Ok);
- O conteúdo da resposta *JSON* deve ter a chave *token,* que representa a entidade utilizador numa chave única e pessoal (*token* de autenticação).

Na Figura 44 demonstra a validação dos testes unitários de *login* e registo de utilizador.

```
c:\xampp\htdocs\athomews>vendor\bin\phpunit
PHPUnit 4.8.36 by Sebastian Bergmann and contributors.

..############## START TEST Register ################
compare if register is valid and returns HTTP 200 OK
compare if register response shows the User object and the Auth token
############## END TEST Register ################
############## START TEST Login ################
compare if login request with no body data returns HTTP 422 error
compare if login response shows the required fields
compare if login is valid
compare if login is valid and returns HTTP 200 OK
############## END TEST Login ################
S

Time: 1.05 minutes, Memory: 16.50MB

OK, but incomplete, skipped, or risky tests!
Tests: 3, Assertions: 7, Skipped: 1.
```

Figura 44 - Validação de testes unitários de login e registo de utilizador

O número de testes é 3, um deles é o teste que corre internamente do sistema e os outros dois são os testes referênciados. O número de verificações efetuadas foram 7. As primeiras 3 estão relacionadas com o teste de registo de utilizador, uma para validar o código de estado Html 200 (Ok) e as outras duas para o resultado *JSON* contér as chaves *user* e *token.* As restantes 4 verificações são do teste de *login.* Uma para aprovar se o resultado *JSON* contém as chaves *user* e *password* e os seus valores, outras duas para os códigos de estado Html 200 (Ok) e 422 (*Unprocessable Entity*) e, por fim, a última como confirmação que a chave devolvida, pelo resultado *JSON*, é um *token*.

5.1.2 Testes de Integração

Para realizar os testes de integração foi utilizado a aplicação *Postman*, que simula pedidos Http a ambientes de desenvolvimento. Esta aplicação vem com um *plugin* adicional para testes de integração.

Um dos testes efetuados foi à rota login, como mostra a Figura 45.

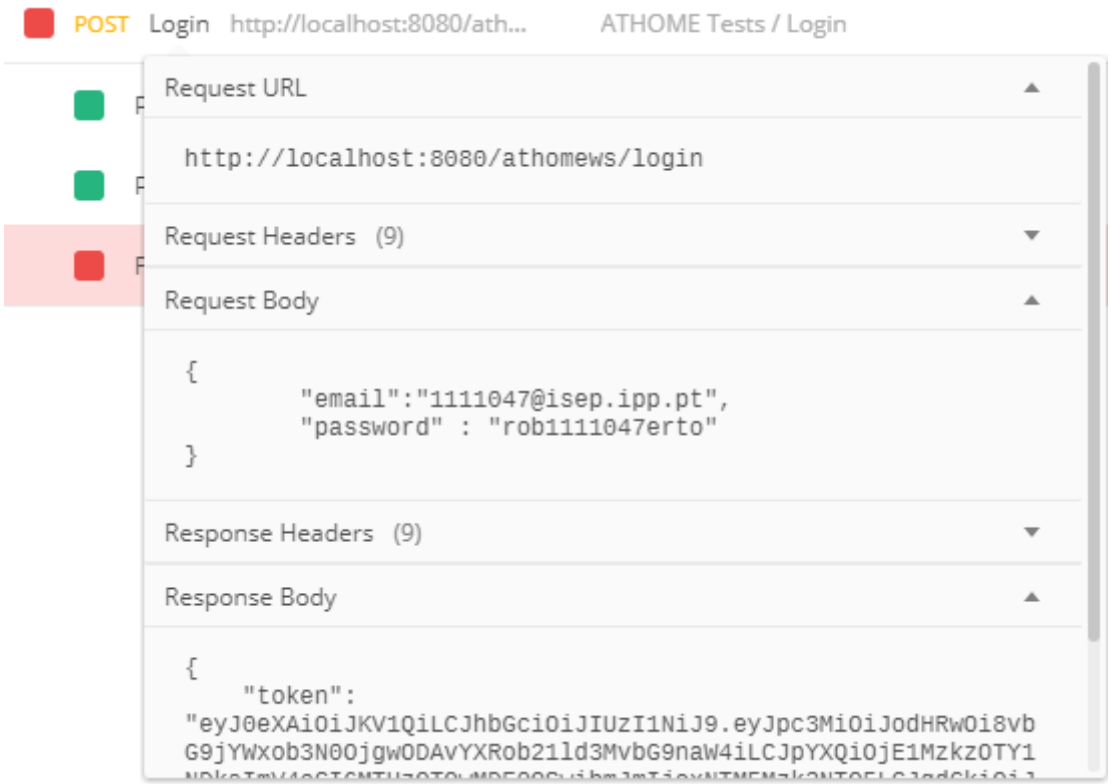


Figura 45 – Dados de teste de integração

O pedido foi realizado com os dados de autenticação corretos, para devolver o *token* de autenticação do utilizador referenciado.

A configuração do teste é mostrado na Figura 46.

```
1  var jsonData = JSON.parse(responseBody);
2
3  tests["Successful POST request"] = responseCode.code === 200;
4  tests["Response body has token"] = jsonData.token !== undefined;
5  tests["Response time is less than 200ms"] = responseTime < 200;
6
```

Figura 46 - Configuração de teste de integração

O tipo de testes efetuados foram:

- Se a reposta ao pedido é devolvido um código de estado Http 200 (Ok)*;*
- Se a resposta ao pedido contém um *token* de autenticação;
- Se a reposta demora mais que 200 milissegundos a ser recebida.

Na Figura 47 pode-se comprovar este tipo de testes.

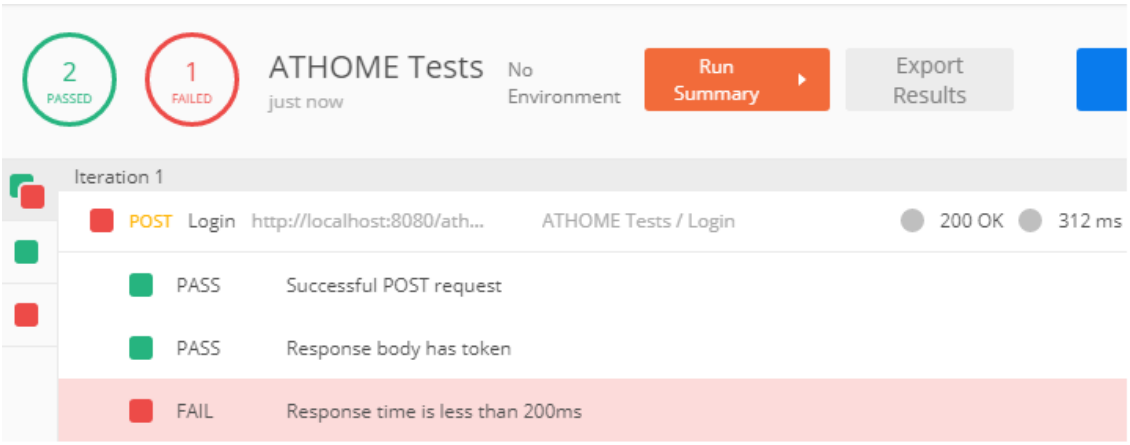


Figura 47 - Resultados de teste de integração ao login

5.1.3 Testes de Sistema

Com a ajuda do Departamento de Engenharia Informática do Instituto Superior de Engenharia do Porto foi possível o uso de uma máquina virtual, que conseguisse simular um ambiente de produção. Para além disso, a máquina ficou disponível para disponibilizar os serviços web, para que a aplicação móvel conseguisse ser desenvolvida e testada mais rapidamente.

Para realizar os testes de sistema utilizou-se novamente o *Postman.*

Foram testados dois cenários com 100 pedidos em simultâneo:

- Obter todas as intervenções efetuadas;
- Registar um novo tipo de tratamento.

Os tipos de testes efetuados foram:

- Para o cenário da obtenção das intervenções:
    - Se a reposta ao pedido é devolvido um código de estado Http 200 (Ok);
    - As respostas devolvidas continham a chave "*intervention";*
    - O tempo de reposta não ultrapassar os 200 milissegundos.
- Para o cenário de registo de um novo tipo de tratamento:
    - Se a reposta ao pedido é devolvido um código de estado Http 200 (Ok);
    - As respostas devolvidas continham a "*health_treatment_types";*
    - O tempo de reposta não ultrapassar os 200 milissegundos.

O resultado final está registado na Figura 48.

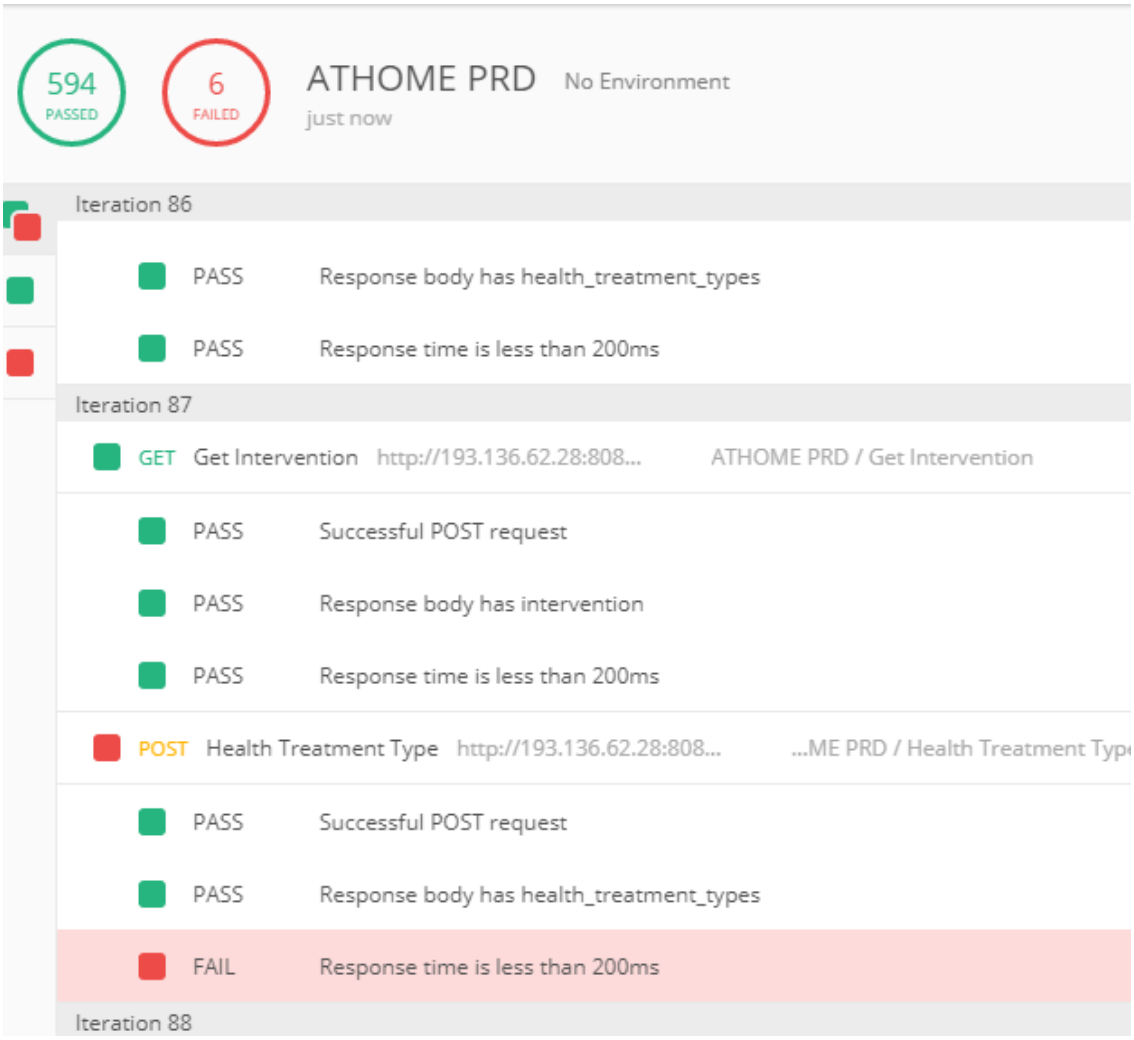


Figura 48 - Teste de sistema

No total, dos 600 testes efetuados, 594 tiveram aprovação e 6 falharam. Os testes falhados são devido ao tempo de execução da inserção de um novo tipo de tratamento na base de dados. De salientar que este erro só começou a ser registado a partir da iteração 88, o que significa que há medida que se regista mais informação na base de dados, tendencialmente as respostas do serviço ficam mais lentas.

5.1.4 Testes de Aceitação

Após uma reunião com o coordenador de projeto, orientadores e com a aluna Bruna Teixeira foram realizados testes de aceitação ao nível dos requisitos funcionais previamente descritos. Na tabela 4 estão referidas as funcionalidades relativas à aplicação móvel:

Tabela 4 - Testes de aceitação da aplicação móvel

Funcionalidades	**Realizado**
Funcionamento *online e offline*	Sim
Compatível com *Android* e iOS	Sim
Interface com o utilizador com 4 idiomas	Sim
Carregamento diário das informações dos pacientes	Sim
Identificação dos pacientes por *QR Code*	Não
Aplicação com autenticação e autorização a conteúdos	Sim
Registo/Visualização de intervenções	Sim
Registo/Visualização de ocorrências não previstas	Sim
Registo do tratamento de roupas	Não
Acesso a dados sensíveis dos pacientes	Não
Registar e aceder a tratamentos de saúde	Sim

Nem todas as funcionalidades requeridas estão descritas nesta tabela, por algumas delas serem associadas à aplicação *backend*.

Na aplicação *backend* não foram verificados os requisitos funcionais, visto que a mesma ainda se encontra numa fase de desenvolvimento e, como tal, está previsto no mês de Novembro ser possível realizar os testes de aceitação de todos os requisitos.

5.1.5 Testes de Usabilidade

Tendo em conta que os testes de aceitação não foram completamente concluídos houveram alguns requisitos que não foram realizados até à data, pelo que se projeta a sua realização futura. Assim que tal aconteça será pedido aos utilizadores o preenchimento dos formulários dos inquéritos de satisfação e usabilidade. Os protótipos dos inquéritos podem ser visualizados no anexo 2 deste documento.

5.2 Processo de Avaliação

Na avaliação final da solução irá ser pedido a um grupo de utilizadores, nomedamente os 10 cuidadores da instituição ODPS, a sua satisfação no seu uso, através de um questionário de usabilidade, fiabilidade e adaptabilidade. Será dada especial atenção na manipulação desses dados, a fim de cumprir todos os requisitos legais e aspectos éticos. No final deste documento poderá ver-se um protótipo deste inquérito, no anexo 2.

5.2.1 Metodologia da Avaliação

A avaliação da usabilidade da avaliação pressupõe vários critérios que avaliam a componente gráfica da aplicação. A escala utilizada para essa avaliação pode ser vista na tabela 5.

Tabela 5 – Classificação das perguntas e respetiva escala para critérios de usabilidade

Classificação	**Escala**
Sem opinião	0
Discordo totalmente	1
Discordo	2
Concordo	3
Concordo totalmente	4

Consoante os resultados a obter deste inquérito irá ser calculada a percentagem da opinião total. Essa percentagem é calculada através da seguinte fórmula:

$$\alpha = (100 * \frac{(\sum \mathrm{Rq})}{4 * nQ * nP}) \quad (1)$$

, em que α é o resultado da percentagem, $\sum \mathrm{Rq}$ é a soma de todas as respostas de todos os questionários dos utilizadores, utilizando a escala representada na tabela 5, nQ é o número total de questões e nP o número de pessoas que responderam ao questionário.

Por exemplo, num universo de 10 pessoas, em que 6 será o número de questões e a soma de todas as escalas das classificações da questões será 200. O resultado da fórmula seria:

$$\alpha = \left(100 * \frac{200}{4 * 6 * 10}\right) \simeq 83.33\% \quad (2)$$

Tendo em conta os seguintes intervalos de usabilidade previstos:

- α > 90% – usabilidade excelente;
- 75% > α ≥ 90% – boa usabilidade;
- 60% > α ≥ 75% – usabilidade média;
- 50% > α ≥ 60% – fraca usabilidade;
- α ≤ 50% – baixa usabilidade ou inaceitável.

Pode-se considerar que, com o exemplo acima descrito, a aplicação enquadra-se no segundo intervalo, ou seja, tem uma boa usabilidade.

A avaliação da fiabilidade do sistema também será uma peça importante. A realização de um questionário, com o grau de utilidade das funcionalidade aos utilizadores irá ser proposto. Este questionário terá classificações diferentes do anterior:

Tabela 6 – Classificação das perguntas e respetiva escala para critérios de fiabilidade

Classificação	**Escala**
Não conheço	0
Inútil	1
Pouco útil	2
Útil	3
Muito útil	4

Como aqui o nível de perguntas está relacionado com as funcionalidade implementadas na aplicação, o tipo de resposta deverá ser diferente.

A fórmula utilizada neste questionário será a mesma da avaliação de usabilidade. No entanto, os intervalos de aceitação serão relativamente diferentes:

- $\alpha > 95\%$ – fiabilidade excelente;
- $85\% > \alpha \geq 95\%$ – boa fiabilidade;
- $75\% > \alpha \geq 85\%$ – fiabilidade média;
- $60\% > \alpha \geq 75\%$ – fraca fiabilidade;
- $\alpha \leq 60\%$ – baixa fiabilidade ou inaceitável.

Pretende-se que a aplicação seja o mais fiável possível e, para que isso aconteça, o resultado deste inquérito terá que ser, no mínimo, superior a 85%.

A avaliação de adaptabilidade terá moldes semelhantes à avaliação de fiabilidade, em relação ao tipo de questões. Nesta avaliação será necessário classificar o grau de dificuldade na utilização das funcionalidades da aplicação. A classificação é diferente, mas a escala mantém-se:

Tabela 7 – Classificação das perguntas e respetiva escala para critérios de adaptabilidade

Classificação	**Escala**
Não conheço	0
Muito difícil	1
Difícil	2
Fácil	3
Muito fácil	4

A fórmula das avaliações anteriores pode ser aplicada nesta avaliação de adaptabilidade. Os intervalos de aceitação serão no mesmo nível dos intervalos da avaliação de fiabilidade, porque é muito importante que os utilizadores sejam capazes de realizar as suas tarefas, sem qualquer auxílio. Esses intervalos podem ser visualizados de seguida:

- $\alpha > 95\%$ – adaptabilidade excelente;
- $85\% > \alpha \geq 95\%$ – boa adaptabilidade;
- $75\% > \alpha \geq 85\%$ – adaptabilidade média;
- $60\% > \alpha \geq 75\%$ – fraca adaptabilidade;
- $\alpha \leq 60\%$ – baixa adaptabilidade ou inaceitável.

Pretende-se que a aplicação seja de fácil manuseamento e, como tal, o resultado deste inquérito terá que ser, no mínimo, superior a 85% para que esteja nos paramêtros aceitáveis.

6 Conclusões

Neste capítulo vão ser apresentados os resultados, até ao momento, do trabalho desenvolvido, os objetivos realizados, as suas limitações e o trabalho futuro ainda a realizar.

6.1 Resultados do projeto

É expetável que os resultados deste projeto sejam úteis para as instituições sociais que necessitem de registar o seu trabalho diário. Neste momento, a aplicação móvel e o serviço web que a suporta compreendem as seguintes funcionalidades:

- Registo e acesso a ocorrências;
- Registo e acesso a intervenções;
- Acesso à informação básica do paciente;
- Registo e acesso a tratamentos de saúde;
- Alteração do idioma da aplicação;
- Sincronização da informação (modo *online*).

No entanto, devido a alterações em alguns requisitos, a aplicação móvel ainda não suporta o registo de tratamento de roupa e o acesso a informação sensível dos pacientes, mas é esperado que essas funcionalidades sejam implementadas em breve.

Por outro lado, a aplicação web e o serviço web que a suporta encontram-se em desenvolvimento, pelo que as restantes funcionalidades previstas e não assinaladas neste

capítulo, fazem parte deste módulo. No entanto, está previsto o fim do seu desenvolvimento no final de Novembro, do ano corrente.

6.2 Objetivos realizados

O objetivo principal deste projeto foi o de desenvolver um sistema que possibilite as instituições a gerir o seu trabalho. Até ao momento, o projeto ainda não está concluido.

Relativamente aos objetivos identificados e propostos no início desde documento, pode-se categorizar os seguintes objetivos realizados:

- Desenvolver/contextualizar tecnologicamente o problema no âmbito do projeto AT-Home;
- Efetuar o levantamento e especificação dos requisitos a realizar;
- Estudar bibliografia referente ao tema do projeto;
- Identificar e estudar sistemas/ferramentas existentes no mercado;
- Elaborar proposta conceptual e de design;
- Identificar e descrever sucintamente outras alternativas concetuais consideradas.

Os objetivos de desenvolver um protótipo funcional de acordo com a proposta conceptual elaborada e especificar/realizar os testes habituais no desenvolvimento de um sistema informático, nomeadamente testes de integração e de aceitação foram maioritariamente realizados. Na aplicação móvel e no serviço web que a suporta, estes objetivos relatados foram realizados, no entanto, os mesmos objetivos não foram realizados na aplicação web e no serviço que a suporta.

Os únicos objetivos não realizados foram a seleção de um subconjunto de intervenientes do sistema e, consequentemente, a avaliação da sua satisfação com o protótipo e o impacto do mesmo no âmbito do projeto AT-Home.

Tanto os objetivos maioritariamente realizados como os objetivos não realizados tiveram atrasos, devido à redefinição de alguns requisitos e na falha de previsão nos tempos previstos de desenvolvimento.

6.3 Limitações e trabalho futuro

A maior limitação deste trabalho foi devido à redefinição de alguns requisitos, tal como foi referido nas seções anteriores. Pretende-se que esta limitação seja ultrapassada brevemente. Para além disso, como os desenvolvimentos da aplicação web e serviço web que a suporta não eram componentes prioritários, as suas previsões de conclusão foram mal efetuadas.

O trabalho futuro a realizar é simples: concluir os requisitos em falta da componente da aplicação móvel e terminar o desenvolvimento da componente da aplicação web.

6.4 Espírito crítico

A escolha desde projeto foi devido ao interesse em ajudar as pessoas necessitadas. A motivação de auxiliar pessoas com necessidades especiais, para que sejam tratadas da melhor maneira possível, é um tema atual da sociedade. A ciência e a tecnologia atual já proporcionam a proteção das pessoas e, também, alguma prevenção de possíveis agravamentos de saúde.

Um sistema de recomendação que alertasse o estado saúde das pessoas e que, consoante determinados critérios de sáude, fosse sugerindo as opções a tomar, pelos cuidadores ou familiares destas pessoas, era uma previsão, depois do tema ter sido levantado. Após reuniões com o coordenador do projeto percebeu-se que esse não era o caminho pretendido e desejava-se mais a gestão do trabalho dos cuidadores.

O estudo do estado de arte ajudou a abrir horizontes, enriquecendo sobre os assuntos relacionados ao tema e ter ideias de como a sociedade atual se comporta com estas situações. Conseguiu-se detetar aplicações existentes que fizessem esse a gestão do trabalho dos cuidadores, mas depois de analisá-las, refenciando as vantagens e desvantagens de cada uma, optou-se por desenvolver uma nova aplicação.

Como qualquer processo de engenharia de *software* foi necessário reunir com os interessados neste projeto, para se perceber os requisitos pretendidos, enumerá-los e validá-los num documento de requisitos. Posteriormente, foi realizada a análise e *design* de uma possível solução, utilizando o documento de requisitos, de forma a relatar a solução escolhida e o conjunto de alternativas à mesma. De seguida, a implementação e os testes da solução e, por fim, a sua avaliação.

Até ao momento, não se foi a tempo de ser disponibilizado um protótipo da aplicação móvel com as suas avaliações. Os tempos projetados na sua conclusão não foram devidamente cumpridos, mas isto servirá de aprendizagem para projeto futuros.

7 Referências

AARP. (2018). Fonte: https://www.aarp.org/: https://www.aarp.org/home-family/caregiving/caregiving-tools/
[Acedido a 14 de Fevereiro de 2018]

Almdal, T., Scharling, H., Jensen, J., Vestergaard, H. (2004). The Independent Effect of Type 2 Diabetes Mellitus on Ischemic Heart Disease, Stroke, and Death: A Population-Based Study of 13 000 Men and Women With 20 Years of Follow-up. *Archives of Internal Medicine*, 1422-1426.

Ambler, S. W. (2003-2006). *http://www.agiledata.org*. Fonte: AgileData: http://www.agiledata.org/images/tddSteps.jpg

Ambysoft Inc. (2002-2018). *http://agiledata.org*. Fonte: AgileData: http://agiledata.org/essays/tdd.html

Astels, D. (2003). *Test-Driven Development: A Practical Guide: A Practical Guide.*

Atlassian. (2018). Fonte: Bitbucket Server: https://www.atlassian.com/software/bitbucket/server

Balasubramanian, Kannan and Mala, K. and Rajakani, M. . (2016). *Cryptographic Solutions for Secure Online Banking and Commerce.*

CareZone. (2018). Fonte: https://carezone.com/: https://carezone.com/features
[Acedido a 14 de Fevereiro de 2018]

Cordeiro, S. &. (2018). *Regulamento Geral de Proteção de Dados (RGPD): o novo pesadelo das empresas?*

Cordeiro, S., Gouveia, L. B. (2018). *Regulamento Geral de Proteção de Dados (RGPD): o novo pesadelo das empresas?*

Duncan, D. (2012). *https://www.cs.colorado.edu*. Fonte: https://www.cs.colorado.edu: https://www.cs.colorado.edu/~kena/classes/5448/f12/presentation-materials/duncan.pdf

Escudeiro, Paula and Bidarra, José. (2008). Quantitative Evaluation Framework (QEF). *RISTI – Revista Ibérica de STI, Nº 1*.

Git. (2018). Fonte: Git: https://www.atlassian.com/git

Greatcall. (2018). Fonte: www.greatcall.com: https://www.greatcall.com/about-us

Healthcare Greatcall. (2018). Fonte: https://healthcare.greatcall.com/: https://healthcare.greatcall.com/
[Acedido a 14 de Fevereiro de 2018]

Hoenig, Helen and Taylor, Donald and Solan, Frank A. (2003). Does Assistive Technology

Substitute for Personal Assistance Among the Disabled Elderly? *American Journal of Public Health*, 330-337.

Kadarina, T. M. and Priambodo, R. (2017). Preliminary design of Internet of Things (IoT) application for supporting mother and child health program in Indonesia. *2017 International Conference on Broadband Communication, Wireless Sensors and Powering (BCWSP)*, 1-6.

Koen, P. A. (2004). *https://web.stevens.edu/cce/NEW/PDFs/FEIEffectiveMethodstoolWorkshop.pdf*. Fonte: http://www.frontendinnovation.com/.

Kruchten, P. B. (1995). The 4+1 View Model of architecture. *IEEE Software*, 42-50.

Laravel. (2018). Fonte: Symfony World: https://symfony.com/projects/laravel
[Acedido a 09 de Outubro de 2018]

Leavitt, N. (2010). Will NoSQL Databases Live Up to Their Promise? *Computer Magazine*.

Mackenbach, J. P. (2013). Political conditions and life expectancy in Europe, 1900–2008. *Social Science & Medicine*, 134-146.

National Economic and Development Authority. (2009). *Value Analysis: Handbook.*

Nicola, Susana and Ferreira, Eduarda Pinto and Ferreira, J. J. Pinto. (2012). *A NOVEL FRAMEWORK FOR MODELING VALUE FOR THE CUSTOMER, AN ESSAY ON NEGOTIATION.*

ODPS. (2018). *http://www.odps.org.pt*. Fonte: ODPS: http://www.odps.org.pt/respostas-sociais/apoio-domiciliario
[Acedido a 13 de Fevereiro de 2018]

Osterwalder, Alexander and Pigneur, Yves. (2010). *Business Model Generation.*

Padhy, R. P. and Patra, M. R. and Satapathy, S. C. (2011). RDBMS to NoSQL: Reviewing Some Next-Generation Non-Relational Database's. *International Journal of Advanced Engineering Science and Technologies 11.1*, 15-30.

Penbase*.com*. (2018). Fonte: https://www.penbase.com: https://www.penbase.com/en/domi-connect/
[Acedido a 14 de Fevereiro de 2018]

Pestana, M. H. and Gageiro, J. N. (2003). *Análise de dados para ciências sociais: a complementariedade do SPSS.* Sílabo.

PHP.NET. (2018). Fonte: PHP.NET: https://secure.php.net/manual/en/intro-whatcando.php
[Acedido a 09 de Outubro de 2018]

Rechel, B., Grundy, E., Robine, Jean-Marie, Cylus, J., Mackenbach, J.P., Knai, C., McKee, M. (2013). Ageing in the European Union. *The Lancet*, 1312-1322.

ResearchGate. (2018). Fonte: www.researchgate.net: https://www.researchgate.net/figure/Figura-1-Modelo-New-Concept-Development-NCD_fig1_317418872

Ribeiro, A. J. (2018). *http://www.odps.org.pt*. Fonte: ODPS: http://www.odps.org.pt/sobre/mensagem-do-presidente
[Acedido a 13 de Fevereiro de 2018]

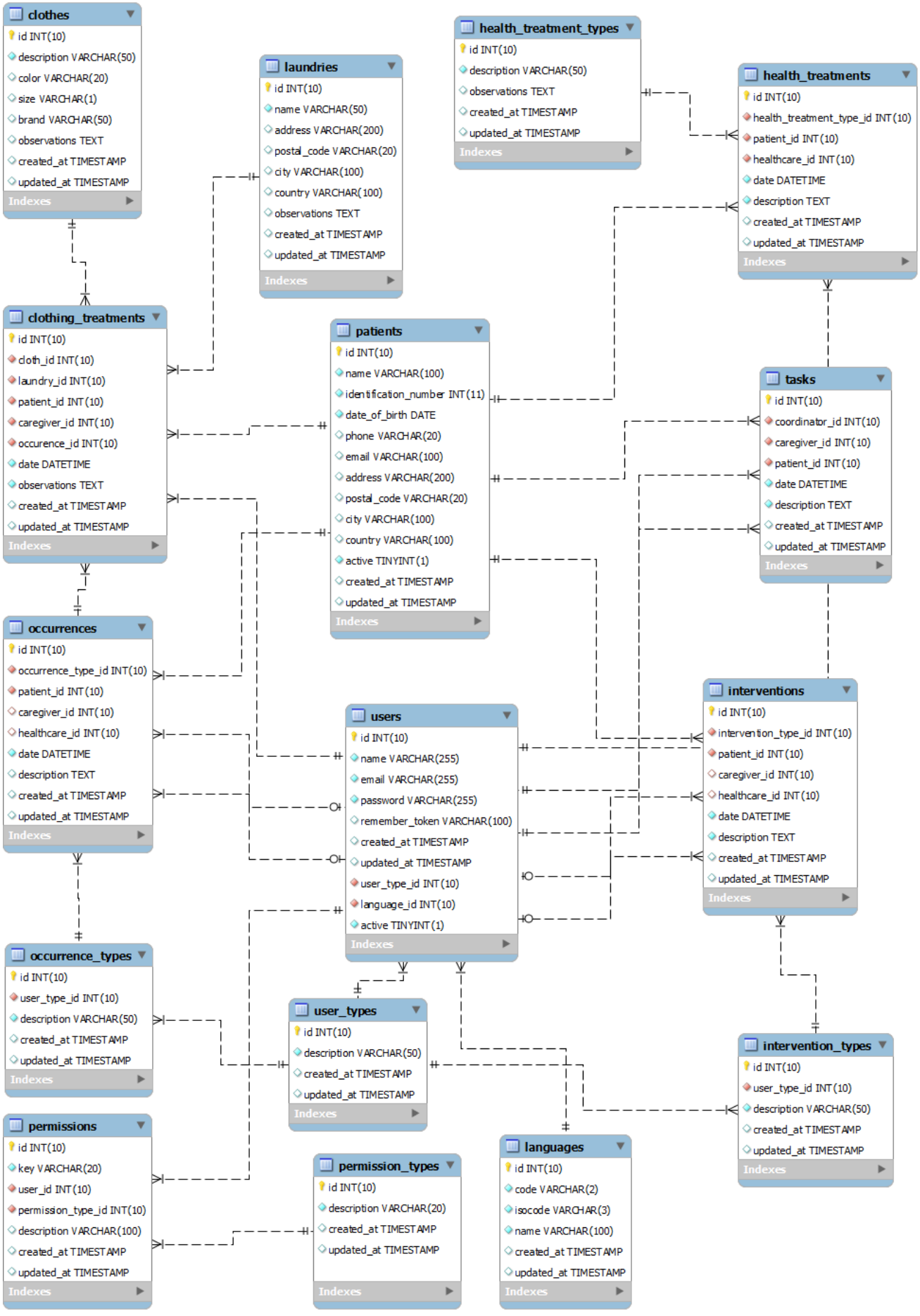
Rosa, M. J. (2016). *O Envelhecimento da Sociedade Portuguesa.* Fundaçao Francisco Manuel Dos Santos.

Woodall, T. (2003). *Conceptualising 'Value for the Customer': An Attributional, Structural*. Fonte: is.muni.cz: https://is.muni.cz/el/1456/jaro2013/MPH_MVPS/39278324/value_Woodall.pdf
[Acedido a 13 de Fevereiro de 2018]

XAMPP. (2018). Fonte: Apache Friends: https://www.apachefriends.org/index.html
[Acedido a 09 de Outubro de 2018]

Yurieff, K. (27 de 03 de 2018). *Steve Jobs warned about privacy issues in 2010*. Fonte: CNN: https://money.cnn.com/2018/03/27/technology/steve-jobs-mark-zuckerberg-privacy-2010/index.html
[Acedido a 08 de Outubro de 2018]

Anexo 1 – Modelo Relacional



Anexo 2 – Inquéritos de Satisfação

1. Usabilidade. Analise o seu grau de concordância relativamente às afirmações sobre a aplicação.

	Discordo totalmente	**Discordo**	**Concordo**	**Concordo totalmente**	**Sem opinião**
A interface apresenta a informação de forma organizada	○	○	○	○	○
A interface facilita o acesso às intervenções/tarefas realizadas	○	○	○	○	○
A interface facilita o acesso à criação de ocorrências	○	○	○	○	○
A interface apresenta um grafismo agradável	○	○	○	○	○
A interface traduz corretamente, na alteração do idioma da aplicação	○	○	○	○	○
Recomenda o uso da aplicação	○	○	○	○	○

2. Fiabilidade. Como classifica o grau de utilidade de cada uma das funcionalides da aplicação.

	Inútil	**Pouco útil**	**Útil**	**Muito útil**	**Não conheço**
Registo e acesso a ocorrências	○	○	○	○	○
Registo e acesso às intervenções/tarefas realizadas	○	○	○	○	○
Acesso à informação básica e sensível dos pacientes	○	○	○	○	○
Registo e acesso a roupas para a lavandaria	○	○	○	○	○
Sincronização da informação (modo *online*)	○	○	○	○	○

Em geral, como considera o nível da aplicação	○	○	○	○	○

3. Adaptabilidade. Como classifica o grau de dificuldade na utilização das principais funcionalides da aplicação.

	Muito difícil	**Difícil**	**Fácil**	**Muito fácil**	**Não conheço**
Registo e acesso a ocorrências	○	○	○	○	○
Registo e acesso às intervenções/tarefas realizadas	○	○	○	○	○
Acesso à informação básica e sensível dos pacientes	○	○	○	○	○
Registo e acesso a roupas para a lavandaria	○	○	○	○	○
Sincronização da informação (modo *online*)	○	○	○	○	○
Em geral, como considera o nível da aplicação	○	○	○	○	○