INSTITUTO SUPERIOR DE ENGENHARIA DO PORTO
MESTRADO EM ENGENHARIA INFORMÁTICA




FD+ Fall Detection

ANDRÉ GOMES DE ALMEIDA
Outubro de 2016

POLITÉCNICO
DO PORTO

Instituto Superior de Engenharia do Porto

FD+ Fall Detection
Deteção de quedas

André Gomes de Almeida

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

Orientador: Jorge Manuel Neves Coelho

Júri:
Presidente:
[Nome do Presidente, Categoria, Escola]
Vogais:
[Nome do Vogal1, Categoria, Escola]
[Nome do Vogal2, Categoria, Escola] (até 4 vogais)

Porto, novembro 2016

Resumo

A expressão "domótica" com origem no latim *domus,* que significa habitação, é a ciência moderna de engenharia das instalações em sistemas prediais, sendo um tema que começa a dominar o mundo tecnológico. Já a *Internet das Coisas* é também um termo cada vez mais familiar e, hoje em dia, é comum verificarmos pequenas "coisas" a interligarem-se ao mundo da Internet, proporcionando assim um maior conforto, uma melhoria substancial da qualidade de vida, bem como uma vasta oferta de aplicações que podem ser úteis para a generalidade da sociedade.

Tendo em consideração o aumento generalizado do interesse pela saúde é fundamental criar valor nesse sentido, utilizando essa mesma tecnologia para oferecer serviços e cooperar com empresas e instituições no âmbito da saúde. Tratando-se de uma área que engloba custos bastante avultados, estes podem vir a ser reduzidos de forma substancial, ao mesmo tempo que se proporciona um aumento da segurança, conforto e bem-estar para os seus utilizadores.

O projeto *FD+ Fall Detection* tem como principal propósito aumentar a segurança de uma habitação, proporcionando a recolha de informação automatizada e a monitorização ativa da ocorrência de quedas numa casa de banho. Possibilita ainda um maior conforto e segurança para quem está responsável pela receção de avisos, permitindo que estes possam ser advertidos sobre a ocorrência de quedas.

Palavras-chave: Domótica, Internet das Coisas, Engenharia, Saúde, Monitorização, Queda.

Abstract

Home automation (also called domotics), emerged from the Latin *domus*, which means home. It's the modern science of engineering facilities in building systems, being a topic that is starting to master the technological world. Furthermore, Internet of Things is also a term increasingly common, and nowadays it's ordinary to recognize little "things" interconnected to the Internet world, providing therefore greater comfort, a substantial improvement in quality of life, as well as a wide range of applications that can be useful to the general society.

Taking into account the widespread interest about health issues, it's essential to create value accordingly, applying that same technology to offer and cooperate on that scope. It concerns a field that implies substantial expenses, which can be reduced significantly, providing an increase in safety, comfort and well-being to its users.

The main purpose of the FD+ Fall Detection project is to increase the safety of a home, providing automated collection of information and actively monitoring falls in a bathroom. It allows greater comfort and reliability for those responsible for the warnings reception, making it possible to alert them upon the occurrence of falls.

Keywords: Domotic, Internet of Things, Engineering, Health, Monitoring, Fall.

Agradecimentos

Gostaria de agradecer ao orientador Jorge Manuel Neves Coelho pela orientação, auxilio, prestabilidade durante a realização deste trabalho.

Aos meus pais e irmão, familiares e amigos por toda a paciência, encorajamento e prestabilidade para comigo, por tudo o que fizeram e continuam a fazer por mim.

"Necessity, who is the mother of invention" – Plato

Índice

Lista de Figuras

Lista de Tabelas

Acrónimos e Símbolos

Lista de Acrónimos

AAL	*Ambient Assisted Living*
AJAX	*Asynchronous JavaScript and XML*
API	*Application Programming Interface*
CSRF	*Cross-site Request Forgery*
CSS	*Cascading Style Sheets*
DOM	*Document Object Model*
HTML	*HyperText Markup Language*
GPS	*Global Position System*
IEEE	*Institute of Electrical and Electronics Engineers*
IoT	*Internet of Things*
IP	*Internet Protocol*
JSON	*JavaScript Object Notation*
MVC	*Model-view-controller*
PHP	*PHP: Hypertext Preprocessor*
PIR	*Passive Infrared Sensor*
QoS	*Quality of Service*
SASS	*Syntactically Awesome Style Sheets*
SQL	*Structured Query Language*
TDD	*Test Driven Development*
WAR	*Web Application Archive*
XML	*eXtensible Markup Language*
XSS	*Cross-Site Scripting*

1 Introdução

Neste capítulo são abordados o contexto, bem como o problema, a Análise de Valor e a abordagem proposta. Contém ainda uma referência da organização do documento. Inicialmente é concebida uma contextualização do tema abordado que permite adquirir noções sobre o mercado e as tecnologias. Posto isto, há uma contextualização, mas ao problema, apresentando assim o real valor da proposta. É ainda abordada uma análise de valor face à contextualização e ao problema seguindo-se a abordagem proposta de forma superficial.

1.1 Contexto

Nos dias de hoje, o avanço das novas tecnologias e o acesso generalizado à informação permitem cada vez mais a possibilidade de utilização dessa mesma tecnologia para o auxílio em determinadas tarefas. Exemplo disso são as habitações, que têm vindo a evoluir nesse sentido, permitindo maior conforto e bem-estar, aos quais se associam cada vez mais a segurança e monitorização de alguns estados e recursos. Desta forma, as casas inteligentes afirmam-se como o próximo passo na evolução do sector. Apesar de não se tratar de um enquadramento totalmente inovador, é possível considerar que, à escala mundial, não é uma realidade presente na maioria dos casos. Consequentemente, existem entraves à sua implementação, como a relação custo/benefício e a necessidade de adaptar todas as habitações já existentes. Considerando que o mercado está repleto de tecnologia e em constante mudança e evolução, a casa inteligente afirma-se como o próximo passo lógico e evolutivo nesse segmento.

É possível assumir que as casas inteligentes são, de facto, uma realidade próxima (Ferguson 2014). Torna-se simples admitir que estas se afirmem como meios dotados de tecnologia útil para a captura de informação em tempo real. Informação essa que, para além de fornecer dados relevantes sobre os habitantes e hábitos comuns, pode vir a proporcionar uma melhoria significativa na qualidade de vida, maximizando o conforto. Posto isto, propõe-se analisar uma forma de esta mesma tecnologia interagir e se interligar com a monitorização de movimentos dentro de uma habitação e respetiva deteção de quedas. Desta forma, a recolha de informação permite que possam ser tomadas ações automatizadas, existindo um sistema capaz de informar que ocorreu uma queda.

1.2 Problema

As novas tecnologias são uma parte integrante e fundamental do dia-a-dia de grande parte da população mundial. Essa mesma tecnologia é cada vez mais uma realidade nas habitações, onde a *Internet das Coisas* (IoT) (Vermesan & Friess 2014) tem vindo a ser integrada de forma gradual. Assim, surgiu a casa inteligente. Uma casa inteligente, sendo dotada de vários sensores e tecnologias, permite que exista tomada de decisão e consequentemente ações para que haja uma melhoria do grau de conforto e de satisfação. Com o aumento da esperança de vida (Nielsen 2015), há também um acréscimo na necessidade do controlo de acontecimentos na habitação, tendo em conta que existem cada vez mais idosos e que estes são mais propícios a quedas.

Assim, surge a necessidade de fazer um acompanhamento mais ativo destes, com o intuito de oferecer um maior controlo e segurança em casos de queda numa divisão da habitação. Com uma monitorização ativa, o habitante pode ser monitorizado na sua habitação, sabendo exatamente o compartimento em que se encontra e caso exista uma queda, é despoletado um alarme.

Para além de o habitante estar a ser monitorizado e acompanhado em tempo real, essa informação está também a ser recolhida, tratada e armazenada.

Propõe-se, então, o desenvolvimento de um sistema que pode ser determinante no aumento da comodidade e monitorização destes casos.

Estima-se que, em 2012/2013, o número de assistências das autoridades locais a questões relacionadas com a assistência de saúde – só em Inglaterra – tenha ultrapassado 1 000 000 de pessoas (Statista 2014a).

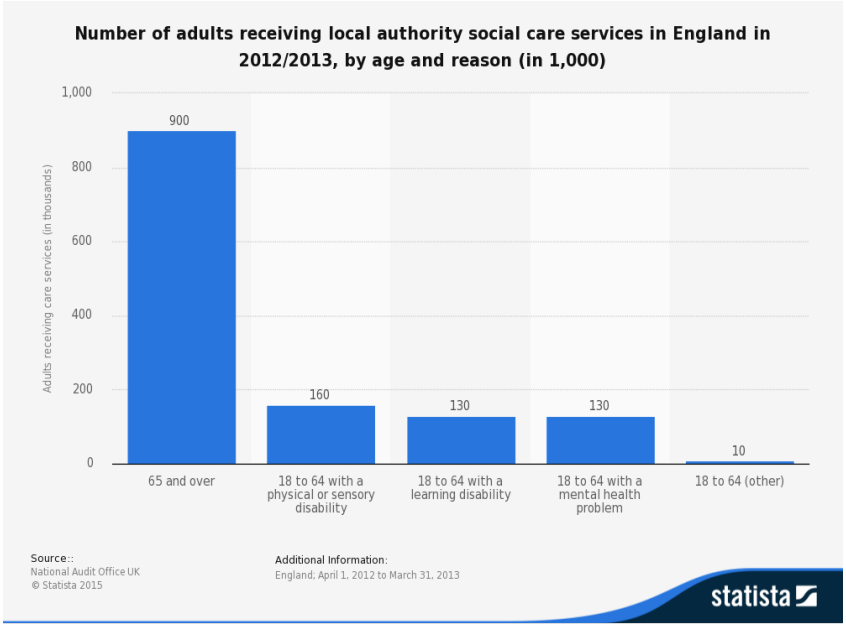


Figura 1 – Número de episódios clínicos em Inglaterra (2012/2013)

Estima-se que os gastos com esses mesmos cuidados tenham chegado aos 4 340 000 de libras (Statista 2014b).

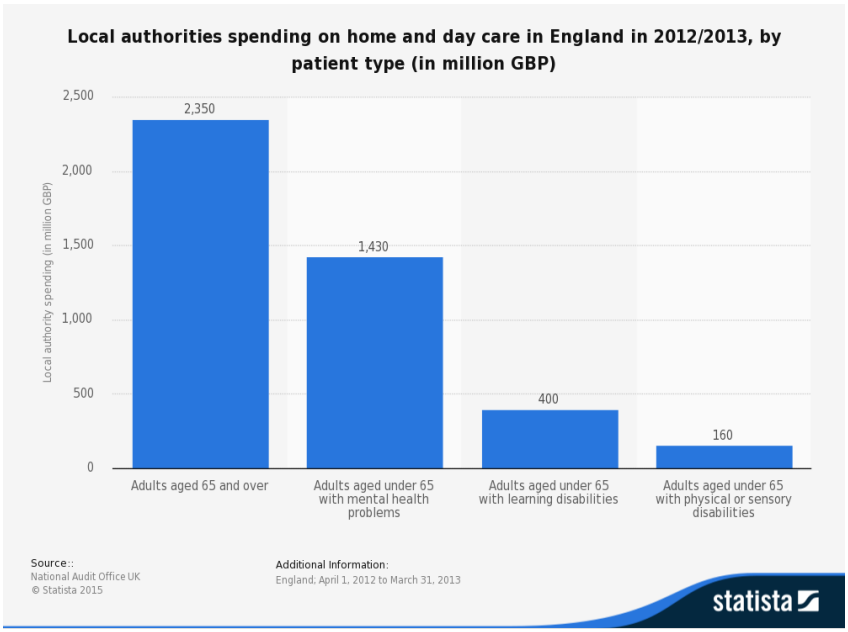


Figura 2 – Custos com assistências em habitações e lares em Inglaterra (2012/2013)

A título de exemplo, um estudo afirma que na Escócia, em 2014, de 739 pessoas que morreram por causa de uma queda, 623 tinham uma idade igual ou superior a 75 anos (Statistia 2016).

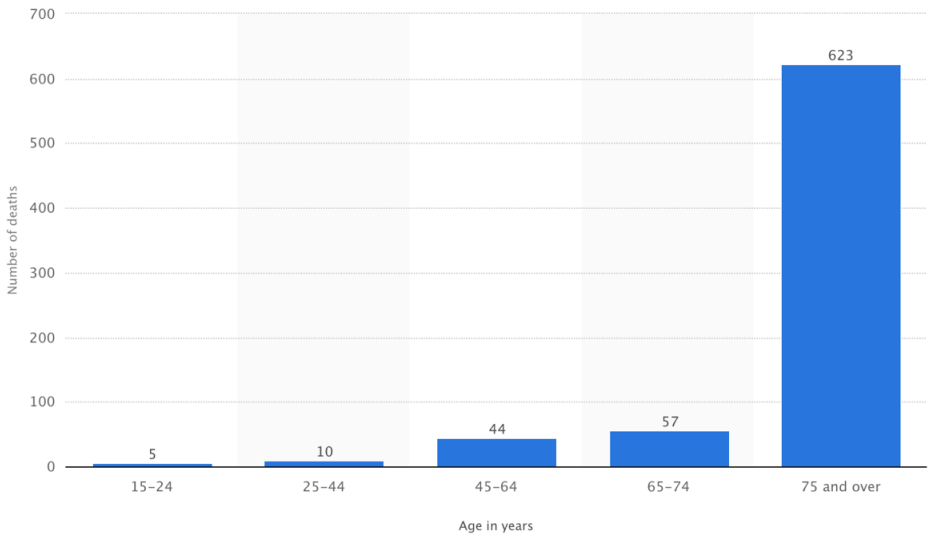


Figura 3 – Número de mortes em quedas na Escócia (2014)

1.3 Análise de Valor

Valor não é intrínseco, não estás nas coisas. Está dentro de nós; valor é a forma como o Homem reage às condições do meio envolvente (Von Mises 1963). Wienhage (Wienhage et al. 2012), definindo uma variante de Análise de Valor designada por Engenharia de Valor, afirma que esta é um método que permite desenvolver novos produtos, melhorar os seus processos de produção, analisar custos e melhorar componentes, visando a redução de custos sem perda de qualidade. A Análise de Valor visa assim proporcionar ao cliente um produto e/ou serviço que ofereça: boa usabilidade, qualidade, flexibilidade, fiabilidade, confiança, etc.

Existem múltiplas interpretações do termo “valor” para o cliente. Valor é a avaliação global da utilidade de um produto baseado no que é recebido e no que é dado. Para um cliente, o que despende para adquirir um produto e/ou serviço, tem de ter uma relação que este considere benéfica para que a sua satisfação perante o mesmo seja favorável (Zeithaml 1988).

Sendo um projeto que acrescenta valor na monitorização de quedas e localização em tempo real de um habitante na casa de banho, apresentando um sistema que se propõe ser dotado de análise de dados, o valor acrescentado está não só na recolha, na disponibilidade, na monitorização, mas também no facto de ser um sistema capaz de despoletar alertas em caso de quedas numa habitação.

1.4 Abordagem

Constitui um desafio implementar um sistema de informação que permita recolher, tratar e armazenar essa mesma informação em tempo real, independentemente do contexto em que se insere, e dos acontecimentos ao longo do tempo. A aplicação *web* tem como finalidade fornecer informação em tempo real de eventuais quedas bem como a configuração do protótipo. Sugere-se assim que toda esta informação seja recolhida com recurso a sensores.

Desta forma, consiste também um desafio a hipótese de que o sistema tenha acoplado a si um sistema de alertas, que se propõem ser despoletados no caso de existirem quedas e acontecimentos que possam comprometer a segurança do habitante.

Dentro do grande tema que é *Ambient Assisted Living* (AAL) (Sousa et al. n.d.) existem duas grandes áreas: o apoio aos utilizadores e o apoio aos cuidadores. A abordagem para o desenvolvimento do sistema foca-se no apoio aos cuidadores com o intuito de lhes prestar um serviço que os deixe mais tranquilizados, sabendo que o habitante está a ser monitorizado.

1.5 Estrutura

Este documento está dividido em sete capítulos principais. São eles:

- Capítulo 1: Contextualiza o projeto e a necessidade do desenvolvimento do mesmo, bem como a interpretação do problema;
- Capítulo 2: Apresenta o Estado da Arte das soluções e tecnologias;
- Capítulo 3: Apresenta os testes efetuados a soluções existentes no mercado;
- Capítulo 4: Apresenta o *design* conceptual da solução;
- Capítulo 5: Apresenta a construção da solução e tecnologias utilizadas;
- Capítulo 6: Apresenta a descrição das experiências e avaliação realizadas à solução desenvolvida;
- Capítulo 7: Apresenta a conclusão bem como o trabalho futuro.

A parte final do relatório contém anexos relevantes para o projeto.

2 FD+ Fall Detection

Neste capítulo são abordados os objetivos propostos, bem como a Análise de Valor e o Estado da Arte. Inicialmente, são abordados os objetivos propostos para a resolução do problema, as premissas assumidas para que o sistema tenha um funcionamento espectável e a explicação do ponto de vista sobre os desenvolvimentos. O Estado da Arte é dividido em duas avaliações: Estado da Arte das soluções e abordagens já existentes no mercado a nível de projetos similares e ainda um Estado da Arte relacionado com as tecnologias de desenvolvimento quer a nível *web*, quer a nível de sistemas móveis (nativos e híbridos). Propõe-se também uma análise à *Internet das Coisas*, que engloba uma apreciação à forma de comunicação entre os dispositivos.

2.1 Objetivos Propostos

O objetivo deste projeto passa por construir uma solução que permita tornar uma habitação mais inteligente, em conjunto com uma plataforma que permita recolher informação, tratá-la e tomar decisões, criando um *environment* próprio com tomadas de decisão para suporte dos habitantes da mesma permitindo identificar quedas num compartimento da habitação que se revela problemático – a casa de banho. Permite identificar onde os habitantes se encontram (dentro desse compartimento) e caso lá estejam, detetar quedas através da recolha de dados que advêm do meio envolvente. Desta forma, impõe-se a necessidade de existirem vários sensores dispersos pela divisão para a captura dessa mesma informação. É possível que o resultado final possa ser incorporado numa habitação convencional ou numa casa inteligente. Propõe-se então criar uma solução que permita verificar e monitorizar a divisão, identificando

quando há uma queda ou quando há uma presença. É fulcral o despoletar de alertas e a existência de tomadas de decisão do próprio sistema em casos de queda em que o habitante não se consiga levantar.

A sociedade tem vindo a assistir a uma preocupação crescente e generalizada com a saúde e bem-estar (Nielsen 2015). Ao mesmo tempo, o conceito *Internet das Coisas* torna-se cada vez mais uma realidade. Desta forma, as oportunidades de mercado na área são mais vastas.

Tendo em conta a importância do termo "habitação", que se apresenta como sinónimo de conforto e segurança, dá-se conta de uma oportunidade de negócio para uma implementação desta natureza, considerando um segmento de mercado relacionado com a área da saúde e bem-estar.

Existindo as casas inteligentes, que não são novidade nos dias que correm, não significa que existam casas inteligentes que contemplem um sistema de deteção de queda. No entanto, não significa também que não hajam projetos diferenciados que possam vir a ser integrados na solução final. Todas as casas inteligentes têm um sistema de recolha de informação e processamento desta e, portanto, esses projetos poderão ser adaptados (em teoria) para a conceção de um projeto com ideias semelhantes. Trata-se, portanto, de um mercado ainda pouco explorado, mas com interesse real generalizado da população mundial (Nielsen 2015).

O desenho e a implementação do sistema devem ser pensados e bem estruturados para o tornar num sistema simples, homogéneo e com uma navegabilidade "natural", o que permite que a solução se afirme como um produto final de qualidade. Assim, uma boa modelação do sistema é um fator decisivo, bem como a preocupação em implementar um sistema capaz de dar resposta às necessidades do utilizador. Estes afirmam-se como fatores chave para o bom funcionamento do sistema, sendo importante a recolha de informação sobre o movimento dos habitantes, para completar e auxiliar a tomada de decisão da solução final.

Pressupõe-se a existência de uma corrente elétrica constante, bem como de uma ligação estável à Internet, para a realização de um projeto desta natureza, em que as falhas são controladas. Estes pressupostos afirmam-se como condições de operação fulcrais para que o projeto possa funcionar em pleno, sendo assim consideradas como um dado adquirido pelo

utilizador do produto. Caso um pressuposto seja alterado ou descartado, propõe-se que projeto tenha uma solução de *backup* para contornar o problema. Neste caso, um sistema de corrente autónoma e uma ligação móvel à Internet para assegurar que em casos extremos seja mantido em pleno funcionamento. Assim, pode existir a necessidade de adaptações e/ou reajustes para que a solução final se mantenha o mais transparente possível para o utilizador.

Propõe-se ainda que os dados sensíveis sejam armazenados numa base de dados utilizando algoritmos de encriptação e de segurança conhecidos, bem como a utilização de boas práticas de programação, vindas dos *guidelines* da documentação das tecnologias utilizadas.

Em resumo, propõe-se que o sistema seja constituído por uma aplicação *web* que permite configurar um compartimento para que possam ser monitorizados todos os movimentos que nele ocorram, bem como a deteção de quedas e o seu registo e ainda um protótipo com vários sensores que permitam recolher a informação pretendida do compartimento em questão.

2.2 Análise de Valor

A Análise de Valor é um ponto crucial para o sucesso de um novo produto e/ou serviço. A necessidade de conseguir entrar no mercado e provar aos possíveis clientes que, de facto, aquele produto e/ou serviço é algo de que necessitam, é uma tarefa exigente e que pode comprometer o sucesso do mesmo. A Análise de Valor é um tema que deve ser eficazmente considerado. Permite, de uma forma direta, apresentar a sua abordagem, pensamento e modo de trabalho, mostrando como este se distingue dos seus concorrentes (diretos e/ou indiretos). O grande foco deve permanecer sempre na explicação do valor do negócio e nunca na tecnologia que tem acoplada a si. Destacar um produto da concorrência é um processo fundamental para o sucesso do mesmo (Zeithaml 1988). A proposição de valor é a oferta de produtos e serviços das empresas que acrescenta valor ao cliente (Embley et al. 2006).

O serviço que se propõe desenvolver permite adicionar a uma habitação (convencional ou inteligente) funcionalidades que possibilitem realizar a monitorização de quedas e movimentos numa casa de banho. Na eventualidade de surgir uma queda, fica registado em sistema e é acionado um alerta para informar que essa mesma situação ocorreu.

Os clientes-alvo do serviço podem assim ser definidos com facilidade, tendo em conta a abrangência do mesmo. Habitações convencionais ou inteligentes, hospitais, lares de idosos, ou até mesmo infantários poderão utilizar o sistema a seu favor.

Com esta tecnologia, para além de os habitantes poderem ser monitorizados em tempo real na casa de banho, existe também a possibilidade de armazenar o histórico destes. Permite assim gerar uma maior sensação de segurança na sua própria habitação, criando um meio envolvente mais controlado.

A posição no mercado de um sistema desta natureza é feita ao longo do tempo, tratando-se de um sistema que pertence a uma grande área que é a Domótica (BOLZANI 2004). Um dos fatores negativos é a privacidade do utilizador, já que embora este seja um sistema que não capta imagens nem se assume como demasiado intrusivo, existe uma abdicação desta, ainda que muito reduzida, para quem utiliza.

Analisando os prós e contras, as necessidades dos utilizadores são mais importantes que a própria privacidade, tratando-se de um sistema de auxílio e monitorização de quedas, podendo inclusive salvar vidas. Assim, a oposição destes é mais baixa comparativamente com outros produtos.

Relativamente aos produtos e/ou empresas concorrentes, o destaque desta plataforma face aos restantes está na interligação e no *environment* criado, podendo ser acoplado a qualquer habitação, seja ela uma habitação convencional ou uma casa inteligente, com benefícios diretos na segurança de um habitante. É ainda um produto que funciona de forma autónoma e sem a necessidade de transportar um objeto para deteção de queda. Estes são fatores que se destacam e acrescentam valor comparativamente aos concorrentes diretos e indiretos.

2.2.1 Modelo Canvas

Tabela 1 – Modelo Canvas

<table>
<tr>
<td rowspan="2">Key Partners
Tecnológico
Marketing
Lares
Hospítais</td>
<td>Key Activities
Qualidade do produto
Suporte ao cliente
Inovação/manutenção das plataformas
Produção</td>
<td rowspan="2">Value Propositions
Inovação
Performance
Risco reduzido
Acessivel
Simples
Prático
Saúde e bem estar
Segurança
Historial
Segmento de pessoas sem riscos de saúde (monitorização preventiva)
Segmento utentes com patologias conhecidas (monitorização ativa)</td>
<td>Customer Relationships
Alto apoio de serviço: suporte com tempo máximo de resposta 24h
Possibilidade de contacto caso haja uma anómalia grave no sistema
Equipa altamente qualificada</td>
<td rowspan="2">Customer Segments
Saúde
Pessoas com particular interesse em Tecnologia
Pessoas com particular interesse em Saúde
Doentes
Lares</td>
</tr>
<tr>
<td>Key Resources
Intelectual
Tecnológico
Saúde</td>
<td>Channels
Internet</td>
</tr>
<tr>
<td colspan="3">Cost Structure
Desenvolvimento das plataformas
Manutenção das plataformas
Custos de matéria prima
Custos fixos (contabilidade, luz, suporte, servidores)
Investimento tecnológico
Licenças
Investigação</td>
<td colspan="3">Revenue Streams
Acesso à plataforma + hardware = 200 € (acresce valor por sensor ultrasónico para abrangir uma maior área)

Formas de pagamento:
Cartão crédito/débito, Dinheiro, …

Contribuição das fontes de rendimento:
Internet - Estimativa de ~90%
Retalhista - Estimativa de ~10%</td>
</tr>
</table>

2.3 Estado da Arte

2.3.1 Soluções e Abordagens Existentes

"Casas Inteligentes" foram motivos de diversos estudos (Harry Wang 2013) (Helal et al. 2009) (Lee & Kwon 2013), no entanto, tanto quanto o autor conhece, estes não se focam na deteção de quedas. Este assunto é por si só relevante pela quantidade de situações críticas que de outra forma não seriam alvo de socorro atempado. A indústria de casas inteligentes representa uma oportunidade para os fornecedores de tecnologia e a plataforma de casas inteligentes providencia uma posição vantajosa sendo flexível, adaptável e escalável. As habitações com monitorização da saúde e bem-estar são vistas como a próxima fronteira de oportunidades (Harry Wang 2013).

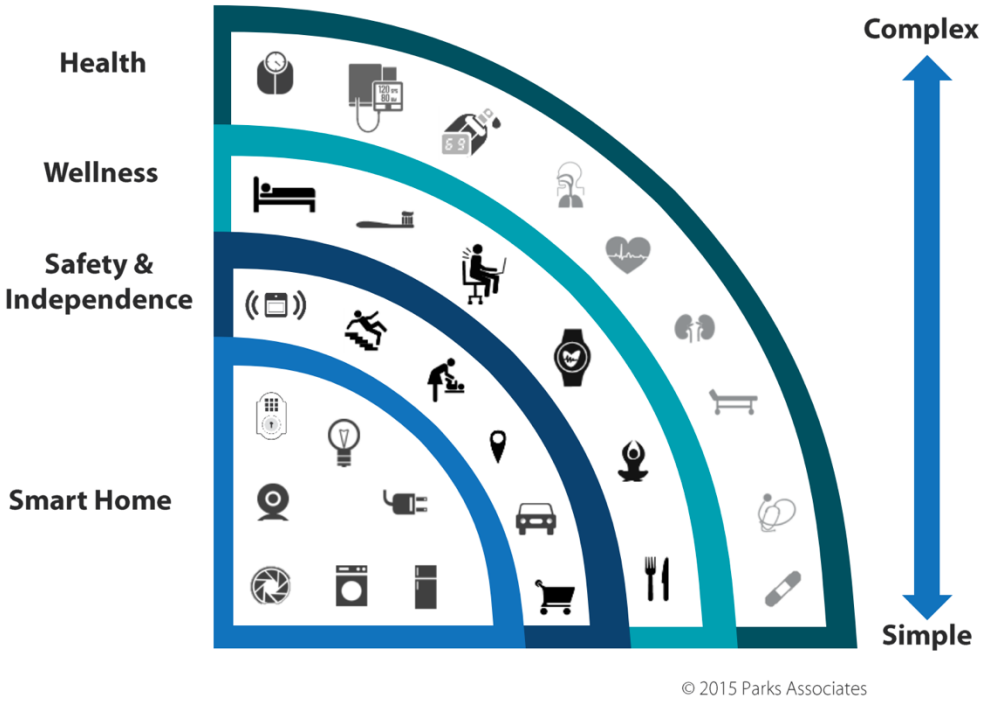


Figura 4 – Serviços de saúde numa casa inteligente

Ao analisar alguns produtos existentes no mercado relacionados com deteção de quedas é possível destacar diferentes tecnologias. Essas tecnologias desenvolvidas podem ser divididas em algumas categorias, onde se destacam os produtos de AAL nacionais e internacionais e ainda duas subcategorias como o apoio aos utilizadores e o apoio aos cuidadores.

2.3.1.1 Projetos Internacionais

- **Sense4care:**

  Sense4care (Sense4Care n.d.) é um produto espanhol desenvolvido para deteção de quedas para pessoas idosas que permite melhorar a sua qualidade de vida, proporcionando assistência em caso de queda. Permite que sejam detetadas quedas dentro e fora da habitação através de um dispositivo que é transportado pelo idoso e que, em caso de deteção de queda, aciona um aviso para o serviço de emergência ou familiares. O sistema permite ainda interligar-se com uma aplicação móvel para quando o idoso se desloque para fora da habitação, seja possível localizá-lo através do recurso do *Global Positioning System* (GPS). Trata-se, portanto, de um dispositivo que deteta quedas, tendo como contra a necessidade de ser transportado para todo o lugar.

Figura 5 – Sense4care

- **Simbad:**

  A Universidade de Manchester, em conjunto com a Universidade de Liverpool desenvolveram um projeto designado por Simbad (Bromiley & Thacker 1999), que permite detetar quedas de um ser humano através de sensores infravermelhos de baixa resolução para monitorizar um compartimento. Assim que um ocupante do compartimento cai no chão é automaticamente feito um pedido de ajuda médica.

- **CAALYX – *Complete Ambient Assisted Living Experiment*:**

  CAALYX (Kamel Boulos et al. 2009) é um projeto que teve o seu início em janeiro de 2007 financiado pela Comissão Europeia, e integrando o esforço de oito participantes e utilizadores finais de seis países europeus, com o objetivo comum de realizar a monitorização do estado de saúde. O principal foco deste projeto passa pela deteção e prevenção de alguma condição adversa de saúde, precavendo eventuais problemas antes de estes se desenvolverem. Caso alguma anomalia seja detetada é acionado o serviço de emergência médica, partilhando a posição geográfica do utente e a condição clínica.

  É composto por:

  - **Sistema de monitorização móvel:** recolhe informações dos sinais vitais, bem como de quedas. Facilita ainda a comunicação entre a pessoa e o responsável de saúde;
  - **Sistema de monitorização em habitações:** providencia o contacto com o responsável de saúde e serviços como televisão, videochamada, etc.;
  - **Sistema de monitorização para o responsável:** providencia a informação sobre quem está a seu encargo e a ser monitorizado.

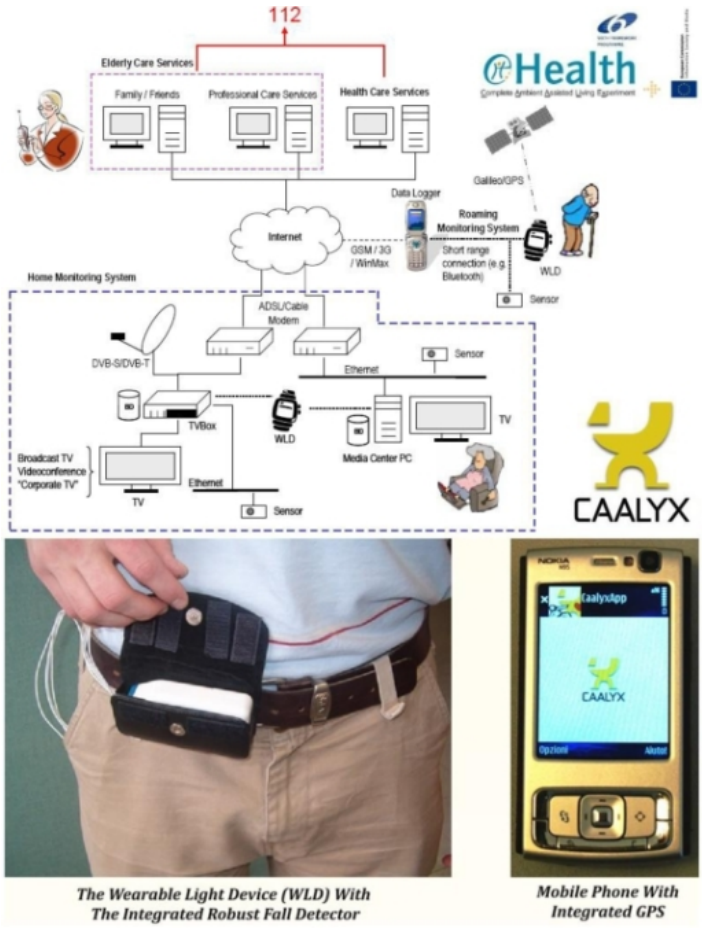


Figura 6 – CAALYX: esquema geral

2.3.1.2 Projetos Nacionais

Relativamente a projetos realizados em Portugal, destacam-se:

- **VirtualECare:** ***intelligent assisted living***

  Projeto desenvolvido na Universidade do Minho na área de AAL pelo Departamento de Informática. O VirtualECare (Costa et al. 2009) é um sistema multiagente inteligente não só para monitorar e interagir com seus clientes (pessoas idosas ou parentes), mas também para se interligar a outros sistemas de computação em execução em diferentes instituições de saúde, centros de lazer, centros de treinos ou lojas. Tem com principal intuito realizar uma monitorização ativa, isto é, uma monitorização constante que permita detetar anomalias e situações que possam comprometer a segurança e bem-estar do utente.

- **aal@home**

  Projeto desenvolvido por três instituições, sendo elas a PLUX – Wireless Biosignals, IT/IST – Instituto de Telecomunicações e a CEFITEC – FCT – Universidade Nova de Lisboa, cujo produto foi desenvolvido com o objetivo de realizar uma monitorização prolongada e contínua dos utentes nas suas habitações. O sistema é baseado em cinco aspetos:

  (a) Sistema *wearable;*
  (b) Integração e sistema modular;
  (c) Comunicação sem fios;
  (d) Monitorização contínua;
  (e) Portabilidade.

aal@home monitoriza três parâmetros fisiológicos: nível de atividade, frequência cardíaca e saturação de oxigénio no sangue. Com o intuito de realizar estas medidas, o sistema integra dois tipos de sensores: um acelerómetro de três eixos e um oxímetro de pulso (Sousa et al, n.d.).

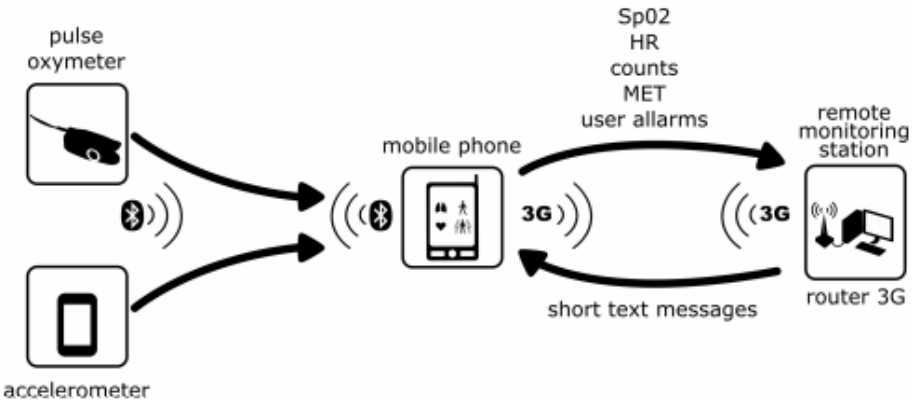


Figura 7 – aal@home: Fluxo de dados entre componentes

2.3.1.3 Aplicações móveis

Existem ainda algumas aplicações disponíveis que permitem detetar quedas através do acelerómetro dos dispositivos móveis. Desta forma, as aplicações podem enviar e-mails e mensagens personalizadas pelos utilizadores para um número de emergência configurado previamente, colmatando e oferecendo opções viáveis para este tipo de acontecimentos. Algumas aplicações permitem ativar o sistema de GPS para uma localização mais precisa. Desta forma, o utilizador tem de ativar a aplicação e realizar o seu percurso. Sendo um sistema que utiliza o acelerómetro, existe a probabilidade de falso-positivos e falso-negativos, visto que a ocorrência da queda do dispositivo móvel pode causar uma aceleração que a aplicação móvel considere ser um valor aceitável para despoletar eventos e consequentemente causar falso-positivos (Silva 2011).

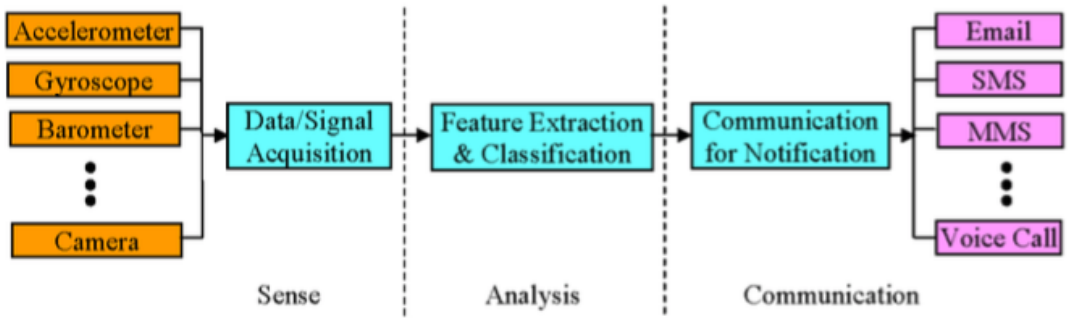


Figura 8 – Arquitetura básica e comum para deteção de quedas (Habib et al. 2014)

- **MOVER: Activity Monitor and Fall Detector for Android Mobile Devices**

  MOVER (Monitor et al. n.d.) é um projeto que foi desenvolvido pela organização Fraunhofer AICOS, sendo esta uma aplicação móvel, mais precisamente para a plataforma Android (Google n.d.), que é dotada da capacidade de detetar quedas. Utilizando os sensores embutidos no telemóvel (acelerómetro, giroscópio, etc.) permite recolher informação e realizar a monitorização do utente. Caso seja detetada uma queda, é acionado um pedido de ajuda. É ainda uma aplicação que tem como finalidade combater o sedentarismo e está dividida em seis estados:

  - *Sleeper* (maior sedentarismo);
  - *Sitter*;
  - *Lagger*;

- *Walker*;
- *Mover*;
- *Hyper* (menor sedentarismo).

Contém ainda uma comunidade que permite fornecer informação e classificar o utilizador dentro de um desses seis estados.

Analisando o Estado da Arte, verificamos que existem diversas soluções, utilizando diversos meios para atingir o resultado mais preciso possível. Desta forma, existem muitas soluções que dependem de dispositivos e da boa vontade do utilizador em os transportar, pelo que, em muitos casos, a ocorrência de acidentes pode não ser detetada.

2.4 Tecnologia Relevante

2.4.1 Aplicação *Web*

As aplicações *web* têm cada vez mais relevância no dia-a-dia dos utilizadores. Isto porque são facilmente acessíveis quer em computadores, quer em dispositivos móveis. Com o surgimento do *Hypertext Markup Language 5* (HTML5) (w3schools 2014) passou a existir uma nova forma de apresentar as aplicações, sendo este definido como a versão mais recente do HTML, onde leva uma marcação simples com o intuito de estruturar um documento para uma plataforma de desenvolvimento de aplicações. Entre outras características, HTLM5 inclui novos elementos e APIs Javascript para melhorar o armazenamento, multimédia e acesso ao *hardware* (Mozilla n.d.). Nesta versão, foram introduzidas novas funcionalidades para ajudar os programadores de aplicações *web*. Foram introduzidos novos elementos baseados na pesquisa sobre as práticas de desenvolvimento, e com especial atenção dada à definição da conformidade clara para os *user agents* com o intuito de melhorar a interoperabilidade (w3schools 2014). Houve então uma melhoria significativa no desenvolvimento destas mesmas aplicações bem como um maior suporte para os novos dispositivos que surgem no mercado (dispositivos móveis). Pode ainda ser adicionado Javascript para programar o comportamento da página para melhor tratamento da mesma, bem como do *Cascading Style*

Sheets 3 (CSS3) que é o último *standard* para CSS (w3schools n.d.) e que permite tornar as aplicações web mais agradáveis visualmente.
Uma linguagem muito utilizada no desenvolvimento de aplicações *web* é o PHP (PHP n.d.). PHP, que significa 'PHP: Hypertext Preprocessor', afirma-se como uma linguagem de programação de ampla utilização, interpretada, que é especialmente interessante para desenvolvimento para a *web* e pode ser combinada dentro do código HTML.

Existem *frameworks* que aceleram e agilizam o processo de desenvolvimento dessas mesmas aplicações. São exemplo:

- **CakePHP** (CakePHP n.d.)**:** *Framework* de desenvolvimento que assenta no padrão de arquitetura de *software* designado por *Modal-View-Controller* (MVC), onde é permitido agilizar o processo de desenvolvimento na linguagem de PHP, utilizando recursos de geração de código. É uma *framework* que está licenciada com a licença MIT, permitindo assim ser utilizada para aplicações com fins comerciais. Permite ainda criar traduções, acessos à base de dados e autenticações de forma simples e rápida. É também uma *framework* que se preocupa com a segurança, tendo cuidado com a validação de *inputs,* proteção *Cross Site Request Forgery* (CSRF), prevenção de injeção *Structured Query Language* (SQL), prevenção de *Cross-Site Scripting* (XSS), etc.;

- **Yii Framework** (Yii PHP n.d.)**:** *Framework* de desenvolvimento na linguagem de PHP, que apresenta como principais vantagens ser extremamente rápida devido ao facto de carregar só as funcionalidades que o programador considere necessárias, tendo ainda uma arquitetura que permite funcionar de forma eficiente com *Asynchronous Javascript and XML* (AJAX). A nível de segurança possui as mesmas preocupações que as referidas anteriormente, tais como: validação de *inputs,* filtro de *output*, injeção de SQL, prevenção de XSS, etc. Permite também ela desenvolver sobre o padrão de arquitetura MVC bem como a separação da lógica de negócio da apresentação;

- **Laravel** (Laravel n.d.)**:** *Framework* de desenvolvimento em PHP que permite desenvolver aplicações *web* de forma simples, com autenticação, permitindo gerir

rotas, sessões e *cache* de forma simples e elegante. É também uma *framework* de fácil manutenção.

- **Grails Framework** (Grails n.d.)**:** *Framework* de desenvolvimento que assenta no padrão de arquitetura de *software* MVC. É uma *framework* de desenvolvimento para *Java Virtual Machine* (JVM), baseado em *Groovy* (Apache Groovy n.d.) e que permite desenvolver aplicações *web*. As suas principais caraterísticas são ter uma curva de aprendizagem menor relativamente a outras e conter vários *plugins* disponíveis para agilizar e acelerar o processo de desenvolvimento. Existe possibilidade de integração com os principais IDEs disponíveis no mercado – Netbeans (Oracle Corporation n.d.), Eclipse (The Eclipse Foundation n.d.), IntelliJ IDEA (JetBrains n.d.), etc.;

2.4.2 Internet das Coisas *(IoT)*

Internet das Coisas, ou *Internet of Things* tem vindo gradualmente a enraizar-se no dia-a-dia. Um bom exemplo disso é o caso das televisões "inteligentes" (*Smart TV)*, que já permitem uma série de funcionalidades muito para além da principal. Este é o próximo passo natural da evolução tecnológica, onde a comodidade e funcionalidades são acopladas às "coisas", dando-lhes "inteligência".

Em 2012, foram estimados 8 700 000 000 de dispositivos ligados à Internet. Estima-se que em 2020 existirão 50 100 000 000 de dispositivos ligados à Internet (Statista 2015b). Assim, é possível concluir que este se afirma como um mercado em constante evolução e crescimento, com numerosos dispositivos a ligarem-se e a trocarem informação entre si.

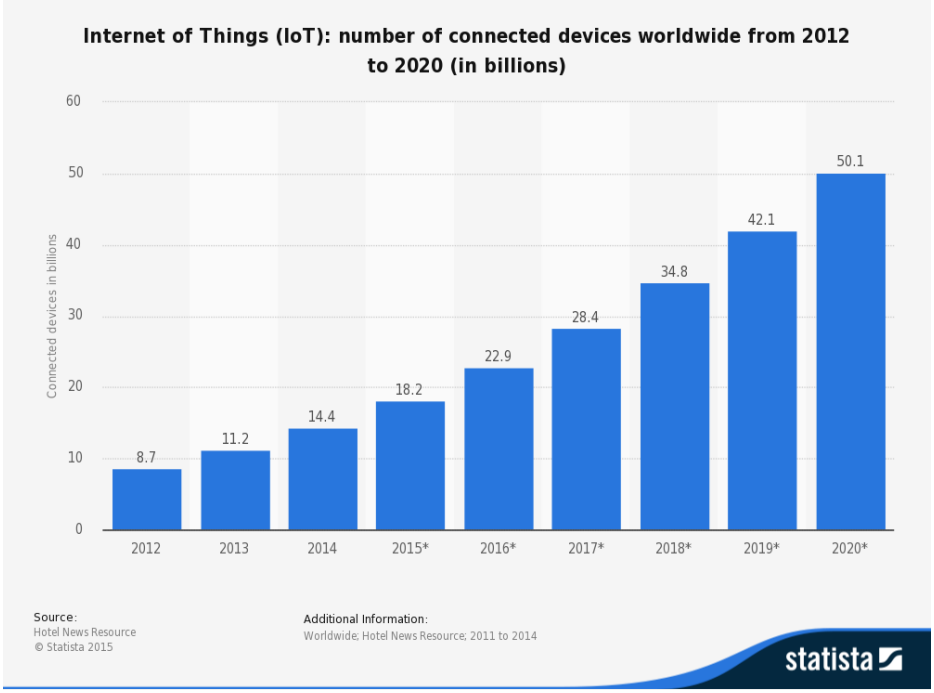


Figura 9 – Número de dispositivos ligados à Internet

É ainda possível verificar que, para além do crescente número de dispositivos ligados à Internet, existe também um consumo desses mesmos dispositivos. Assim, pressupõe-se que haja um gasto de 546 000 000 000 de dólares americanos (USD) para consumidores comuns e de 868 000 000 000 para o mercado empresarial. Em 2020, estima-se que haja um crescimento de 1 534 000 000 000 de USD gastos para os consumidores comuns e um total de 1 477 000 000 000 para o mercado empresarial. Desta forma, é possível detetar um notável crescimento para o consumidor comum (Statista 2015c).

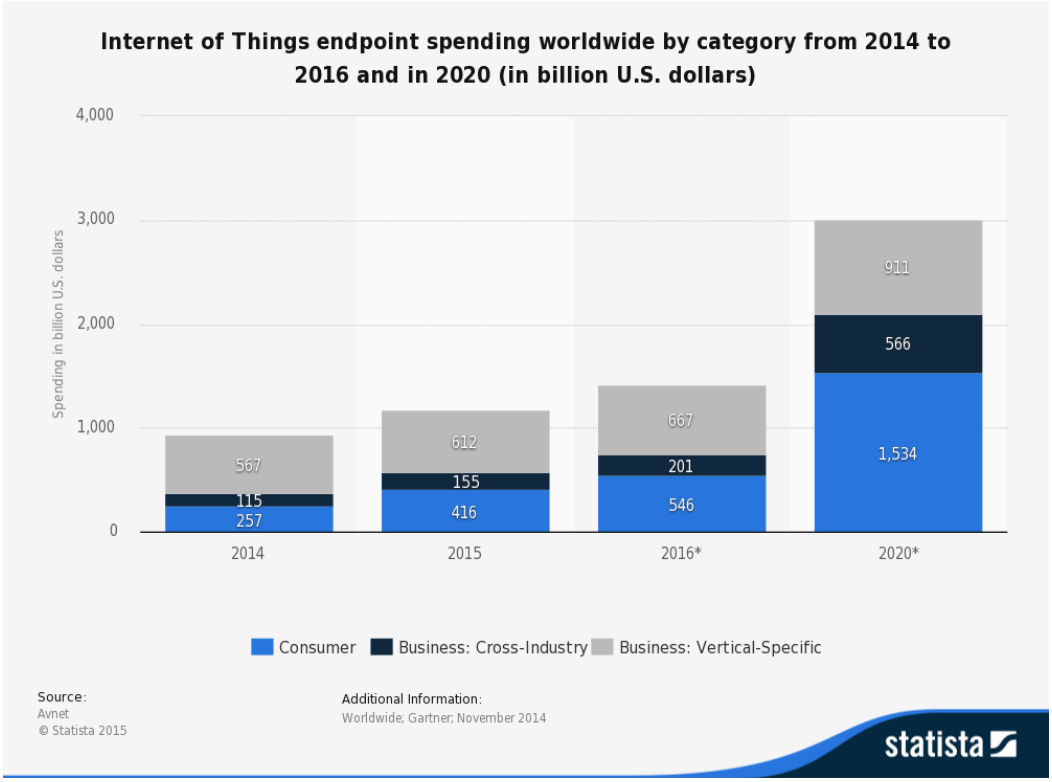


Figura 10 – Valor despendido com a *Internet das Coisas*

O contexto (*context-aware*) é também ele de grande importância para que o comportamento aplicacional possa ser adaptado em conformidade com o contexto em que está a ser lido, através de sensores ou estados. As caraterísticas do contexto podem ser divididas em:

- Espacial;
- Temporal;
- Ambiental;
- Perfil do utilizador;
- Físico e/ou mental;
- Dispositivos e/ou rede (Perera et al. 2013).

Existem algumas *frameworks* de desenvolvimento para aplicações com contexto:

- **Context Toolkit** (Neto et al. 2014)**:** *Framework* que facilita o desenvolvimento de aplicações com contexto. Consiste no desenvolvimento de *widgets* com contexto e uma infraestrutura distribuída que hospeda os mesmos. São componentes que providenciam aplicações com acesso à informação de contexto. Contém serviços que permitem o encapsulamento do sensor, acesso aos dados do contexto, partilha dos dados do contexto através de uma infraestrutura distribuída, bem como o controlo de acesso básico para proteção de privacidade;

- **Java Context-Awareness Framework (JCAF)** (Bardram 2005)**:** O objetivo do JCAF é criar uma utilização generalizada, orientado a serviços, baseado em eventos, criar uma infraestrutura e uma estrutura genérica, expressivo de programação Java (Oracle Corporation n.d.) para implementar e desenvolver aplicações sensíveis ao contexto;

- **Intel Context Sensing SDK** (Intel n.d.)**:** É uma biblioteca disponível para Android e Windows que auxilia a integração de capacidades e serviços sensíveis ao contexto nas aplicações. O SDK é flexível, oferecendo vários métodos para utilizar serviços, de forma independente ou combinada.

Essa informação é recolhida através dos sensores que, conforme a aplicação e finalidade do sistema, podem ser de várias categorias, divididas em:

- Resistiva;
- Indutiva;
- Capacitiva;
- Piezoelétrica;
- Fotoresistor;
- Elástica;
- Térmica.

Já os fatores de avaliação de desempenho podem ser medidos em:

- Alcance;
- Resolução ou discriminação;
- Sensibilidade;
- Resposta;
- Linearidade;
- Precisão;
- Acerto (Storey 2004).

Através dos sensores podemos então capturar informação proveniente do meio envolvente e utilizar essa mesma informação para a realização de ações.

Relativamente à comunicação, existe uma tecnologia muito própria para a transmissão entre pequenos dispositivos. O protocolo IEEE 802.15.4 (IEEE n.d.) é um protocolo com uma taxa de transmissão baixa, baixo consumo energético, onde, tipicamente se adapta às exigências do sensor de rede. O protocolo IEEE 802.15.4 está muito associado com o protocolo ZigBee (Koubâa et al. 2005).

Há também a possibilidade de ser utilizada a tecnologia Bluetooth (Bluetooth SIG n.d.), que é uma tecnologia de rede de baixa capacidade e pequeno alcance. Contudo, Bluetooth tem mais potencial que apenas ser uma tecnologia que substitui os cabos. O Bluetooth mantém a promessa de se tornar a tecnologia de escolha para redes ad hoc. Esta é uma tecnologia de baixo consumo e de baixo custo, o que a torna uma solução atrativa para os típicos

dispositivos móveis que utilizamos nas redes ah hoc (Singh et al. 2011). Assim, através desta tecnologia pode ser criada uma rede ad hoc para a troca e partilha de informação pelos diferentes dispositivos.

Uma outra tecnologia que pode também ela ser utilizada para a partilha de informação é a tecnologia Wi-Fi – protocolo 802.11 (Society 2012b). Segundo a Intel, esta tecnologia transmite um sinal de rádio sobre uma ampla gama de frequências, fazendo com que o sinal seja menos suscetível a interferência e difícil de intercetar (Engineers et al. n.d.). É também uma tecnologia válida para a partilha de informação, não sendo a melhor solução para a rede de sensores como as enunciadas anteriormente (Zigbee e Bluetooth).

No que diz respeito à recolha e tratamento da informação, podem utilizar-se diferentes tecnologias para interagir com os sensores. Existem linguagens que são indicadas para a comunicação com estes. Destaca-se a linguagem Python (Python Software Foundation n.d.).

Python é uma linguagem poderosa e rápida que permite interligar com outras tecnologias. Tem como vantagem correr em qualquer dispositivo, ser de fácil aprendizagem e amigável e ainda, ser uma linguagem *open-source*. Abrange diversas áreas devido aos módulos desenvolvidos pela contribuição da comunidade, que permitem um número vasto de possibilidades, tais como desenvolvimento *web*, acessos a base de dados, aplicações *desktop*, aplicações numéricas e cientificas, aplicações de educação, programação de rede e ainda desenvolvimento de *software* e jogos.

2.4.3 Aplicações Móveis

As aplicações móveis têm vindo a afirmar progressivamente a sua forte presença quer no mercado nacional, quer em mercados internacionais. O crescimento dos dispositivos móveis é o principal impulsionador deste fundamento.

O tráfego global de dados móveis vai multiplicar-se quase por 10 nos próximos cinco anos, alcançando os 292 *exabytes* anuais em 2019 (30 *exabytes* em 2014), o que equivale a um aumento de rácio de 57% entre 2014 e 2019 (Cisco n.d.).

Estatisticamente, o número de aplicações disponíveis nas principais lojas eletrónicas – Apple Store (Apple n.d.) e Google Play (Google n.d.) – tem tido um forte crescimento ao longo do tempo. Desde julho de 2008 até junho de 2015 o número de aplicações na Apple Store cresceu exponencialmente, sendo que até junho de 2015, existiam 1 500 000 de aplicações móveis disponíveis. A Apple reportou que existiram 1 000 000 000 de *downloads* de aplicações no mês de abril de 2009. Estes números têm vindo a aumentar e, em outubro de 2015, havia uma estimativa de 100 000 000 000 de *downloads* feitos em todo o mundo (Statista 2015a).

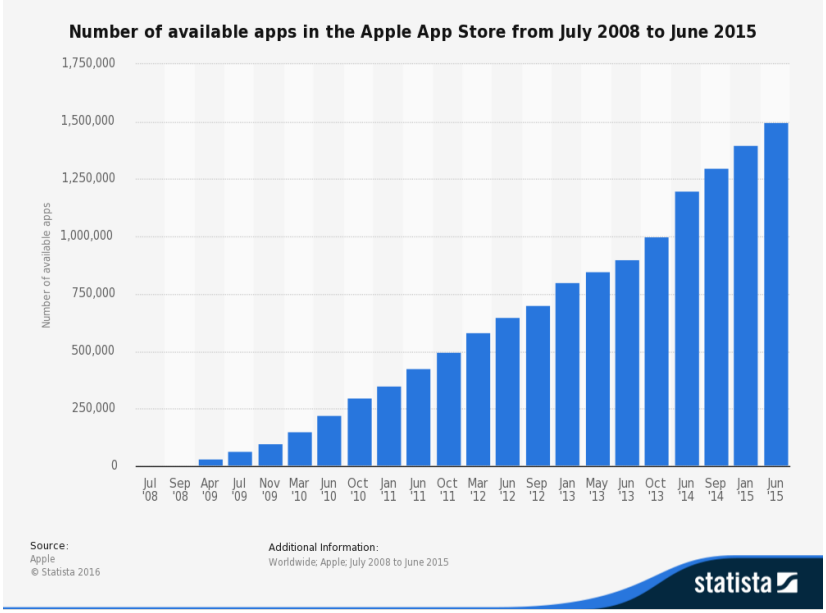


Figura 11 – Número de aplicações disponíveis (Apple Store)

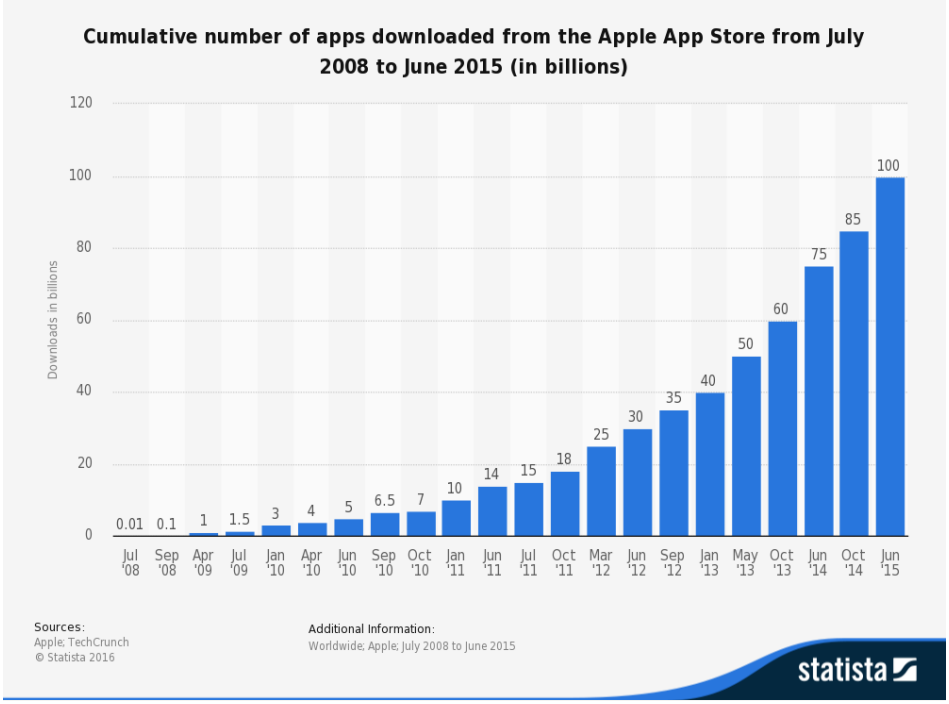


Figura 12 – Número acumulado de *downloads* de aplicações (Apple Store)

Já no mundo do sistema operativo Android para dispositivos móveis, estima-se que desde dezembro de 2009 até novembro de 2015, o crescimento das aplicações móveis tenha atingido o valor de 1 800 000 disponíveis no Google Play.

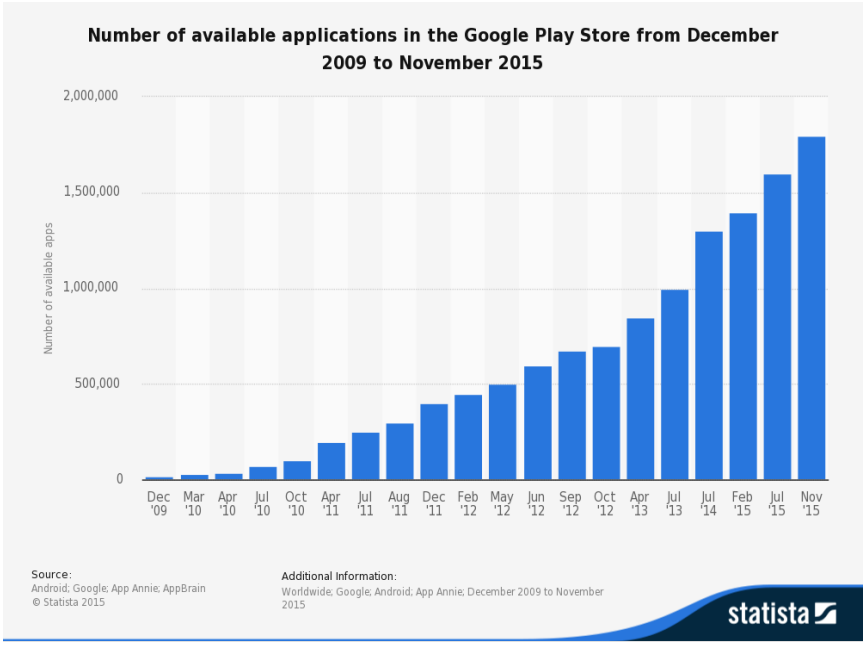


Figura 13 – Número de aplicações disponíveis (Google Play)

Desta forma, comprova-se então que o mercado de dispositivos móveis tem tido uma crescente evolução quer a nível do mercado de Android quer do mercado de iOS. Em julho de 2015, o mundo do Android dispunha de 1 600 000 de aplicações disponíveis para *download,* seguido do mundo do iOS com 1 500 000 de aplicações. Existem ainda outros segmentos de mercado que embora sejam mais reduzidos, oferecem também eles aplicações móveis para as suas plataformas – Windows Mobile (Microsoft n.d.), Firefox OS (Mozilla n.d.), etc.

Aplicações Nativas vs Aplicações Híbridas

Os programadores de aplicações móveis podem desenvolver dois tipos de aplicações: as aplicações nativas e as aplicações híbridas. Aplicações híbridas são aplicações que permitem desenvolver para múltiplas plataformas com apenas um só desenvolvimento. Um só esforço despendido numa aplicação que pode ser utilizada em diferentes plataformas.
As aplicações híbridas contêm alguns tipos de limitações que as nativas não têm. Um dos grandes obstáculos é o seu desempenho, ficando aquém das aplicações nativas bem como outras funcionalidades necessárias para a execução de algumas tarefas. Tarefas essas que são fulcrais para algumas aplicações e que devem ser tidas em consideração na hora da decisão. São exemplo o acesso à câmara fotográfica, acesso à leitura de informações provenientes dos sensores, etc. (Charland & Leroux n.d.). Dependendo da *framework* utilizada, onde cada uma tem as suas vantagens e desvantagens, destacam-se:

- **Ionic Framework** (Ionic Framework n.d.)**:** *Framework open-source* para desenvolver aplicações móveis híbridas com tecnologia *web*. Permite utilizar bibliotecas HTML otimizadas, CSS e Javascript bem como componentes de gestos de CSS e ferramentas para construir aplicações altamente interativas. Construído com *Syntactically Awesome Style Sheets* (SASS) que permite definir valores e utilizar em múltiplos lugares e otimizado para AngularJS (Google n.d.). Têm particular atenção à performance, afirmando que a velocidade é importante e para isso recorrem ao mínimo de manipulação DOM bem como à não utilização de jQuery (The jQuery Foundation n.d.). Utilizam também a aceleração de *hardware* nas transições para que as questões de performance sejam menos notórias face às aplicações nativas (Ionic Framework n.d.);

- **PhoneGap** (PhoneGap n.d.)**:** À semelhança da *framework* apresentada anteriormente, esta utiliza também tecnologias *web* (HTML5, Javascript e CSS) para o desenvolvimento das aplicações. É também ela uma tecnologia *open-source*.

Ambas as *frameworks* apresentadas são bastante similares e têm como base o Apache Cordova (Apache Cordova n.d.) que é também uma *framework* de desenvolvimento.

2.4.4 Infraestruturas de Telecomunicações

As redes de telecomunicações têm vindo a evoluir de forma metódica. Podem ser divididas em duas categorias, isto é, celulares e não celulares.

2.4.4.1 Redes móveis celulares

No que às redes móveis celulares diz respeito, a primeira geração (1G) de redes móveis, desenvolvida nos EUA, foi a *Advanced Mobile Phone System* (AMPS). É baseada em *Frequency Division Multiplexing* (FDM). Um serviço de dados foi adicionado na rede de telefone, que é o *Cellular Digital Packet Data* (CDPD). Utiliza o *Time Division Multiplexing* (TDM). A rede pode oferecer uma taxa de 19.2kbps que explora períodos de inatividade dos canais tradicionais de voz para transportar os dados.

A segunda geração (2G) de redes móveis é principalmente o *Global System for Mobile Communications* (GSM). Foi apresentada inicialmente na Europa e só depois no resto do mundo. Uma outra segunda geração de rede é a *Personal Communications Service* (PCS) ou IS-136 e IS-95; PCS foi desenvolvido nos EUA. O standard IS-136 utiliza *Time Division Multiple Access* (TDMA) com o intuito de partilhar os recursos rádio. O GSM e o PCS IS-136 empregam canais dedicados para transmitir dados.

O *Internacional Telecommunication Union* (ITU) desenvolveu um conjunto de *standards* para a terceira geração (3G) para o sistema de redes móveis sobre o *International Mobile Telecommunication-2000* (IMT-2000) com o intuito de criar uma rede global. Estão programadas para operarem numa frequência perto dos 2GHz e oferecer taxas de transmissão de dados acima dos 2Mbps. Na Europa, o *European Telecommunications*

Standards Institute (ETSI) foi padronizado de *Universal Mobile Telecommunications Systems* (UMTS) assim como a rede 3G (Chaouchi & Luarent-Maknavicius 2009).

Atualmente, a versão mais recente das redes móveis é o 4G. O 4G é definido para atender aos requisitos estabelecidos pelo ITU como parte do *IMT Advanced*. As principais orientações para a evolução da arquitetura do sistema 4G são: reduzir o custo de rede, reduzir a latência de dados e carga de sinalização, mobilidade interna entre outras redes de acesso em 3GPP e não 3GPP, experiência do utilizador com qualidade de serviço (QoS) e capacidade de roaming mundial (Melorose et al. 2015).

2.4.4.2 Redes móveis não celulares

Já nas redes móveis mas não celulares destaca-se a tecnologia IEEE802.11, mais conhecida como Wi-Fi, que é uma rede sem fios pelo que a maior vantagem óbvia é a mobilidade (Gast 2005), tendo a Intel como grande colaborador e ainda a tecnologia ZigBee.

3 Avaliação de Soluções e Abordagens

Neste capítulo são abordadas as grandezas que são utilizadas para avaliar soluções que já existam e que seja pertinente serem utilizadas. São ainda abordados os tipos de testes que a solução desenvolvida deve ultrapassar.

3.1 Grandezas para Avaliação

Num sistema desta natureza é indispensável a existência de testes que permitam tornar a solução final numa solução mais precisa, robusta, com resultados coesos e com um grau de acerto elevado.

Em 2014/2015, depois de excelentes resultados obtidos com um piloto de um sistema de deteção de quedas testado com 200 utentes, a companhia Sense4Care decidiu explorar os direitos comerciais do projeto FATE (FATE n.d.). Este sistema foi desenvolvido e foram lançados vários pilotos, que superaram vários testes com participantes de diferentes parceiros.

Um dos problemas que estas soluções enfrentam são os falso-positivos e os falso-negativos. Por entre os vários testes efetuados, foi possível a obtenção de diversos resultados. Os falso-positivos apresentam-se como um dos fatores preocupantes pelos alertas que possam ativar quando não existe qualquer motivo para isso. O número de falso-positivos é preocupante já que, após originar muitos falso-positivos, poderá levar a que quando ocorra uma queda, este acontecimento seja encarado como mais um falso-positivo e não se preste o auxílio necessário (Moreno et al. 2012).

Já os falso-negativos são igualmente preocupantes, tendo em conta que na ocorrência de uma queda, o sistema pode não ativar o alerta. Este acontecimento (ou falta dele) poderá originar uma falha na prestação de auxílio, deixando o utilizador do sistema em risco de não ser socorrido atempadamente.

Testes ao botão de pânico:

Este sistema tem associado a si uma forma de cancelar eventuais pedidos de ajuda. Este mecanismo deve ser testado, visto que se for um botão "macio" ou de clique com menos pressão pode gerar falso-positivos numa atividade normal do quotidiano (p.e., sentar-se). Já um botão mais "duro" pode levar a que o utilizador não tenha a sensibilidade para sentir se de facto conseguiu acionar o alarme, pelo que pode levar a que um utilizador pressione várias vezes para cancelar um alarme e esteja, ao mesmo tempo, a gerar um novo (Moreno et al. 2012).

	COOSS	FSL	HCPB	NUIG	SEM
Panic button incidences (%)	13,9	19,6	21,1	20,6	11,5

Figura 14 –Total de incidentes para as mensagens geradas pelo botão de pânico por diferentes parceiros

Incidentes na deteção de quedas

A análise aos *logs* armazenado nos dispositivos indicam que a maioria dos alarmes gerados são por uso inapropriado (Moreno et al. 2012).

	COOSS	FSL	HCPB	NUIG	SEM
Fall detection incidences (%)	14,3	35,7	21,1	20,6	11,4

Figura 15 –Total de incidentes na deteção de quedas por diferentes parceiros

Incidentes de alerta na cama

Um dos problemas associados a esta solução deve-se ao facto de os utilizadores não aderirem à utilização do sistema enquanto estão a dormir, não permitindo registar padrões e intervalos regulares de sono. Estes teriam o intuito de recolher mais informação sobre os hábitos de uma pessoa, para ser monitorizado, por exemplo, o número de vezes que a mesma se levanta durante a noite. Caso o utilizador não adira à utilização contínua do sistema, quando o alerta

chegar à central, poderá não conter a informação completa da ocorrência (Moreno et al. 2012).

Falha de transmissão de dados

Um teste de extrema importância que também deve ser realizado é relativo à transmissão de dados, isto é, se os dados que estão a ser difundidos não têm quebras de informação. Uma quebra de transmissão de dados pode levar a que uma ocorrência não seja reportada com sucesso, sendo uma falha grave do sistema.

Testes de usabilidade

Outros testes que um sistema desta natureza enfrenta são os testes de usabilidade, onde é testada a facilidade em transportar o dispositivo e o grau de conforto com o mesmo, que é também tido em consideração. A *User Interface* (UI) são fatores que são testados para se conseguir perceber se as aplicações desenvolvidas são intuitivas e cumprem o propósito da solução final (Moreno et al. 2012).

Em termos qualitativos, estes foram os testes que uma solução presente no mercado efetuou para verificar as falhas e implementar eventuais melhorias para o sistema.

Testes à solução criada

Propõe-se que os testes à solução desenvolvida pelo autor sejam baseados nos testes feitos a outros sistemas semelhantes, sendo que alguns dos casos podem ser adaptados ou descartados, conforme as necessidades da solução apresentada. Em termos técnicos, o sistema não necessita que o utilizador transporte qualquer tipo de objeto. Em consequência, alguns testes, como é o caso da mobilidade e do grau de praticidade do objeto, são testes que não fazem sentido na solução implementada, pelo que devem ser descartados. Desta forma, é fulcral avaliar a solução e perceber quais são os testes que fazem sentido serem aplicados.

Como se trata de um protótipo, propõe-se que o mesmo seja testado com dados simulados, sendo que estes sejam o mais aproximado à realidade, para tornar os testes mais fiáveis.
Testes como a avaliação dos falso-positivos e dos falso-negativos são testes que devem de ser efetuados com bastante rigor e com um elevado número de amostras. Propõe-se contabilizar a quantidade de falsos alarmes que o sistema gera, com o intuito de perceber qual o grau de acerto da solução desenvolvida.

A transmissão de dados é também um fator preponderante ao sistema e como tal deve igualmente ser alvo de testes. Existe a necessidade de o sistema conter uma fila de mensagens para que sejam entregues ao sistema e este as trate convenientemente. Nos testes realizados devem ser aplicados e contabilizadas com sucesso o número de mensagens enviadas e o número de mensagens recebidas, com o intuito de fazer uma percentagem de sucesso da entrega de mensagens.

A solução necessita de sensores para recolher informações do meio envolvente pelo que esses mesmos sensores devem ser alvo de testes. Devem, também eles, ser testados relativamente à sua precisão e ao grau de acerto. Devem também ser capazes de capturar medições em tempo real e com o utilizador em movimento. Testes como o grau de acerto ou acurácia, a repetibilidade e a resposta no ambiente são parâmetros que devem ser tidos em consideração.

A necessidade de recolherem as medidas o mais precisas possível leva a que o grau de acerto do algoritmo seja mais elevado, pelo que é necessário efetuar vários testes para verificar se os sensores estão a recolher as medidas corretas. O ângulo de incidência do sensor ou campo de visão são um fator que deve ser testado, visto que podem existir zonas em que o sensor não consiga abranger, criando um ponto em que não sejam detetadas as ocorrências (*blindspot)*. A resposta ao ambiente deve também ela ser testada, uma vez que os sensores podem encontrar uma condição de humidade (vapor de água do banho).

Estes são fatores preponderantes ao sistema, pelo que os seus testes são fulcrais para o avaliar.

4 Solução

Este capítulo aborda o *design* conceptual da solução para a aplicação *web*, as tecnologias que se propõe serem utilizadas e a abordagem à modelação da base de dados. É ainda apresentada uma abordagem minimalista ao design da aplicação *web* e à forma como está pensada a sua implementação. É também contemplada neste capítulo a lógica de implementação para o desenvolvimento, bem como a modelação da base de dados.

4.1 *Design* Conceptual da Solução

Propõe-se um sistema que seja capaz de detetar pessoas, monitorizar os seus movimentos e detetar quedas. Todo o sistema deve estar interligado, isto é, o protótipo desenvolvido que recolhe informações do meio envolvente deve ele próprio ser capaz de recolher informação, tratá-la e proceder ao envio da mesma para ser armazenada e apresentada na aplicação *web*. Desta forma permite apresentar a informação dos acontecimentos. O grande foco inicial passa por criar e pensar num ecossistema global e próprio, que permita o funcionamento e a interligação dos diferentes mundos. É fulcral desenhar o sistema pensando em várias premissas para diferentes tipos de sistemas, de forma a que este permita uma maior escalabilidade, caso seja necessário. Numa fase inicial, é importante definir dois focos da solução final:

- Aplicação *web*: apresenta-se como a interface de monitorização para o cliente final;
- Protótipo: apresenta-se como a solução que fica colocada no compartimento.

Com a aplicação *web* e um protótipo a capturar informação proveniente do meio envolvente, surge então a necessidade da existência de uma base de dados modulada a pensar nestes cenários, bem como de uma forma que permita a consulta de estados e troca de informação.

No que diz respeito ao protótipo, é necessária a leitura de dados vindos dos sensores e o seu respetivo processamento e tratamento. Propõe-se que os dados sejam lidos através dos sensores existentes para esse efeito – sensor de ultrassom (Indoware 2013) e sensor infravermelho passivo (mais conhecido por PIR - *Passive Infrared Sensor*) (System 2009) – sejam reunidos num ponto "central" e posteriormente enviados para o sistema armazenar essas leituras. Esta solução não contempla apenas a recolha de informação, já que são também executadas ações (gerar alertas) pelo próprio sistema, que efetua as mesmas de forma automática.

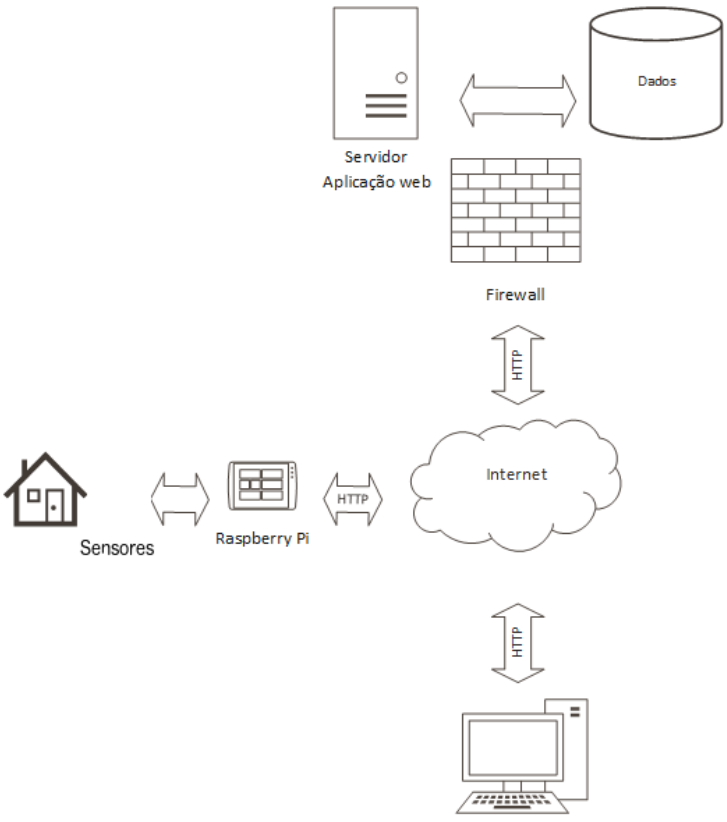


Figura 16 – Esquema geral da solução proposta

Ao aceder à aplicação *web*, propõe-se que os utilizadores possam visualizar uma página inicial com a apresentação do produto, assim como as formas de entrar em contacto com a administração, para esclarecimento de eventuais dúvidas.

É também proposto que os utilizadores validem as suas credenciais e tenham acesso à plataforma. Após *login* efetuado com sucesso, o utilizador tem acesso ao seu perfil e a um painel de controlo (*dashboard)* com informações relevantes sobre a habitação e mais concretamente sobre o compartimento (casa de banho) que está a ser monitorizado bem como os valores obtidos deste.

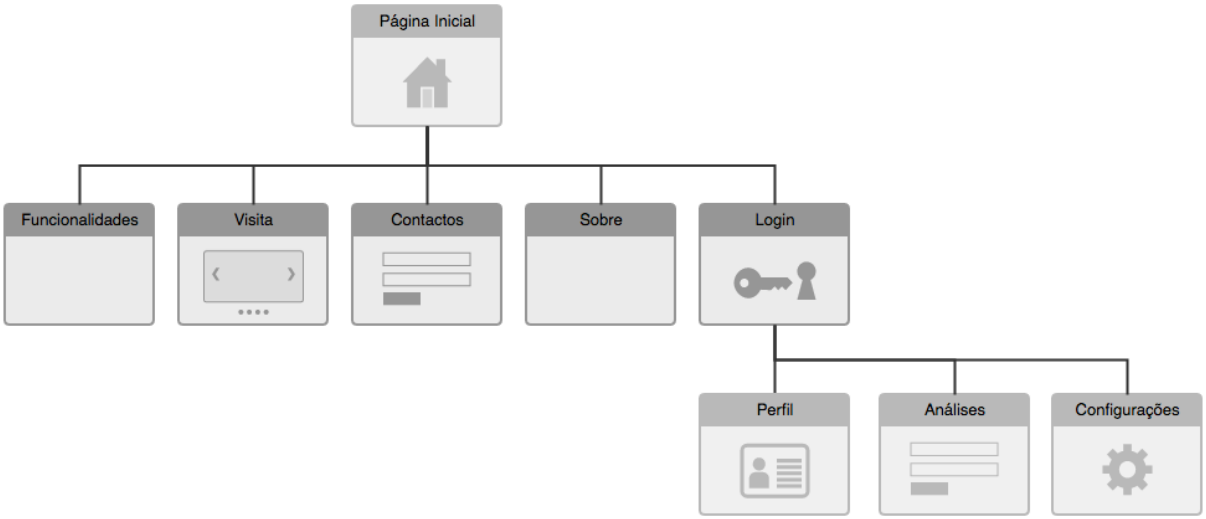


Figura 17 – *Design* minimalista da aplicação *web*

No que diz respeito ao *reporting*, propõe-se que o sistema realize análises automaticamente, de uma forma independente, recolhendo informações sobre o compartimento e do habitante. Propõe-se que a informação do habitante a ser recolhida passe pela presença deste no compartimento e o respetivo estado (em pé, agachado ou queda).

No que diz respeito à lista das funcionalidades do sistema, é proposta para implementação a possibilidade de tomada de decisão sobre acontecimentos que ocorram no compartimento. No caso da ocorrência de uma queda, esta deve ser detetada e ser realizado o pedido de auxílio de forma autónoma pelo sistema através do envio de e-mails e notificações para o(s) utilizador(es) identificados(s) na aplicação como sendo o(s) contacto(s) de emergência em

caso de ocorrência de uma queda. Esses contactos de emergência são denominados por responsáveis. É proposta a existência de um ou mais deste tipo de contactos, podendo ser feita a respetiva manutenção (adicionar, editar e remover) na aplicação *web*.

Para a deteção de uma queda é medida a velocidade que permite verificar que tipo de movimento está a ser feito. Caso exista uma velocidade demasiado alta e a diferença de alturas (capturada na configuração e a atual lida) seja um valor superior ao previamente configurado em sistema, estamos perante uma queda.

O cálculo de velocidade é uma grandeza que tem duas variáveis: distância e tempo. Esta grandeza é calculada pela seguinte fórmula:

$$V = \frac{\Delta D}{\Delta T}$$

, onde V representa a velocidade, a variável D corresponde à distância percorrida (deslocamento) e T ao intervalo de tempo (Pérez 2003).

A implementação da comunicação baseia-se numa fila de mensagens que permite o seu armazenamento e receção no destinatário (protótipo e servidor). Existe uma tecnologia designada por *RabbitMQ* (Pivotal Software n.d.) que funciona como um intermediário que permite à aplicação enviar e receber mensagens. É um *software open-source* e com suporte comercial, onde se destacam como principais características as mensagens robustas, a sua fácil utilização, o facto de correr na maioria dos sistemas operativos e de suportar múltiplas plataformas de desenvolvimento. Após a sua instalação e configuração, todas as mensagens trocadas são feitas de forma assíncrona, separando o enviar e o receber das mensagens.

É fundamental a aplicação de boas práticas no desenvolvimento da solução empregando um desenvolvimento iterativo (Rational Software 2004). Impõe-se ainda a necessidade de ser realizada uma análise às funcionalidades implementadas, com o intuito de verificar se o *software* está de acordo com as especificações, de forma a validar se o sistema corresponde às expetativas. No desenvolvimento de *software*, Verificação e Validação (V&V) é um conceito bastante abordado. A verificação do *software* procura pela consistência, integridade e exatidão do *software* e pela sua documentação de apoio, uma vez que está em desenvolvimento e fornece suporte para a conclusão de que o *software* é válido. Permite confirmar se o resultado do desenvolvimento de *software* corresponde às exigências iniciais para este.

Já a validação de *software* é a confirmação fornecendo evidências objetivas de que as especificações do *software* estão em conformidade com as necessidades dos utilizadores e as funcionalidades previstas, e que os requisitos específicos utilizados através do *software* são cumpridos de forma consistente (Food et al. 2002). Em termos práticos, a verificação permite perceber se estamos a construir certo o produto, enquanto que a validação permite perceber se estamos a construir o produto certo, sendo necessário ter em consideração este conceito (V&V) para o desenvolvimento da solução final.

O diagrama de decisão seguinte (figura 18) permite perceber, de uma forma global, como é definida a solução, qual o fluxo da informação e ainda como funciona e decide as várias etapas através dos cálculos e resultados recolhidos através do meio envolvente.

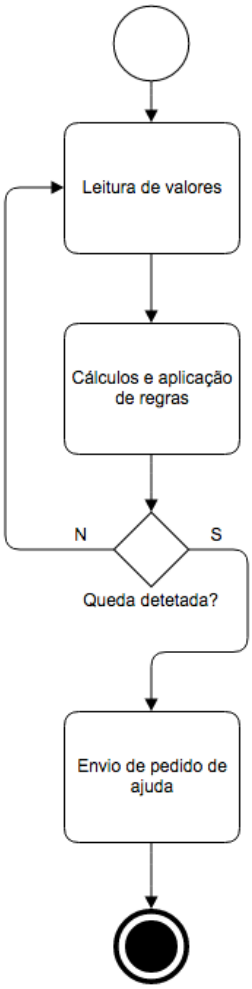


Figura 18 – Diagrama de decisões do sistema

4.2 Arquitetura da Solução

As propostas das tecnologias para a solução passam por uma *framework* de desenvolvimento para a aplicação *web*, sendo que neste caso a escolha recai sobre a *framework* Grails, por ter uma base que assenta no padrão de arquitetura de *software* MVC e permite acelerar o processo de desenvolvimento. Tem como vantagem o facto de poder ser feita a modelação da base de dados do lado da *framework* e de permitir o desenvolvimento de serviços externos para serem consumidos, apresentando a possibilidade da interligação com outras aplicações (p.e. móvel). Permite também proteger o código, ficando assim encapsulado e protegido num ficheiro designado por WAR, tornando a solução numa aplicação de código fechado. É dotada ainda da gestão de segurança necessária para um sistema desta natureza, entendendo-se como sendo uma abordagem adequada para o desenvolvimento da aplicação *web*, comunicação com a base de dados e ainda dos serviços para comunicação externa. Propõe-se que esses serviços sejam desenvolvidos em RestFul (Richardson & Ruby 2007) por ser um protocolo cliente/servidor e por permitir um conjunto de operações fulcrais para a proposta de solução, sendo elas pedidos de informação (GET), adição e atualização da informação (*POST* e *PUT)* e remoção da informação (*DELETE)*. Uma outra vantagem desta tecnologia é a interligação com o servidor de mensagens RabbitMQ necessária para o desenvolvimento do projeto.

Relativamente à base de dados, a escolha recai sobre a tecnologia MySQL (Oracle Corporation n.d.) por ser uma base de dados relacional, *open-source*, bastante popular e com grande utilização no mercado profissional.

Em relação à forma de captura de informação, propõe-se que esta seja recolhida utilizando sensores e um Raspberry Pi (Raspberry Pi Foundation n.d.) para o tratamento dos dados e respetivo envio para o servidor. Uma outra possibilidade está na utilização de um Arduino (Arduino n.d.), ponderando os custos/benefícios da utilização deste face ao Raspberry Pi, no entanto, as limitações do mesmo (funcionalidades e características) podem conduzir a uma solução menos expansível para novas funcionalidades que possam vir a ser adicionadas. Optando por um Raspberry Pi, é proporcionada uma maior capacidade de processamento,

com mais memória disponível e comunicação Ethernet (Society 2012a), o que simplifica a comunicação com o servidor.

No que diz respeito ao desenvolvimento do *software* do protótipo, a linguagem utilizada para o desenvolvimento da solução é Python. Esta escolha tem em conta a simplicidade da linguagem, o facto de apresentar uma comunicação com os sensores mais simples e prática, existindo também uma extensa documentação e bibliotecas que auxiliam o desenvolvimento.

4.3 Base de Dados

Devido ao facto de existir armazenamento, surge a necessidade de utilizar algumas normas de segurança relativamente à base de dados. Desde logo, impõe-se a existência de um controlo no acesso à informação, bem como no fluxo da mesma, de forma a que os dados não sejam acessíveis por canais ocultos. A encriptação de dados é também uma medida que se propõe utilizar, tendo em conta que os mesmos são pessoais e com informações que não deverão ser visíveis por quem não tem autorização para tal, caso exista um desvio no canal de transmissão. Desta forma, propõe-se a utilização de uma *hash* de encriptação designada por SHA-256 (Mendel et al. 2006) pelo facto de ter sido inventada pela *National Security Agency* (NSA), sendo que devolve um maior número de bits relativamente ao SHA-1. É também uma das *hash* mais populares (Chaouchi & Luarent-Maknavicius 2009).

A modelação da base de dados (figura 19) é então fundamental para que esta consiga armazenar a informação de forma coerente e com uma relação lógica. Assim, uma habitação tem vários compartimentos e um compartimento pode ter um ou vários sensores, onde, cada sensor recolhe várias medidas. Após essas medidas serem analisadas, existem duas tabelas capazes de armazenar os eventos e o tipo de eventos que são detetados ao longo da recolha de informação. No que diz respeito ao acesso à aplicação *web*, existe a necessidade de o utilizador ter um local para armazenar informação do seu perfil, uma tabela que o inclua dentro de um grupo de utilizadores e as respetivas permissões de acessos para cada grupo. Existem ainda tabelas para armazenamento de informação de funções e permissões em casos particulares para determinados utilizadores, uma tabela de registo (*log*) de ações e transações

da aplicação e ainda uma tabela que permite associar responsáveis para contactos de emergência. Todas as tabelas, campos e relações estão pormenorizadas nos anexos.

Solução

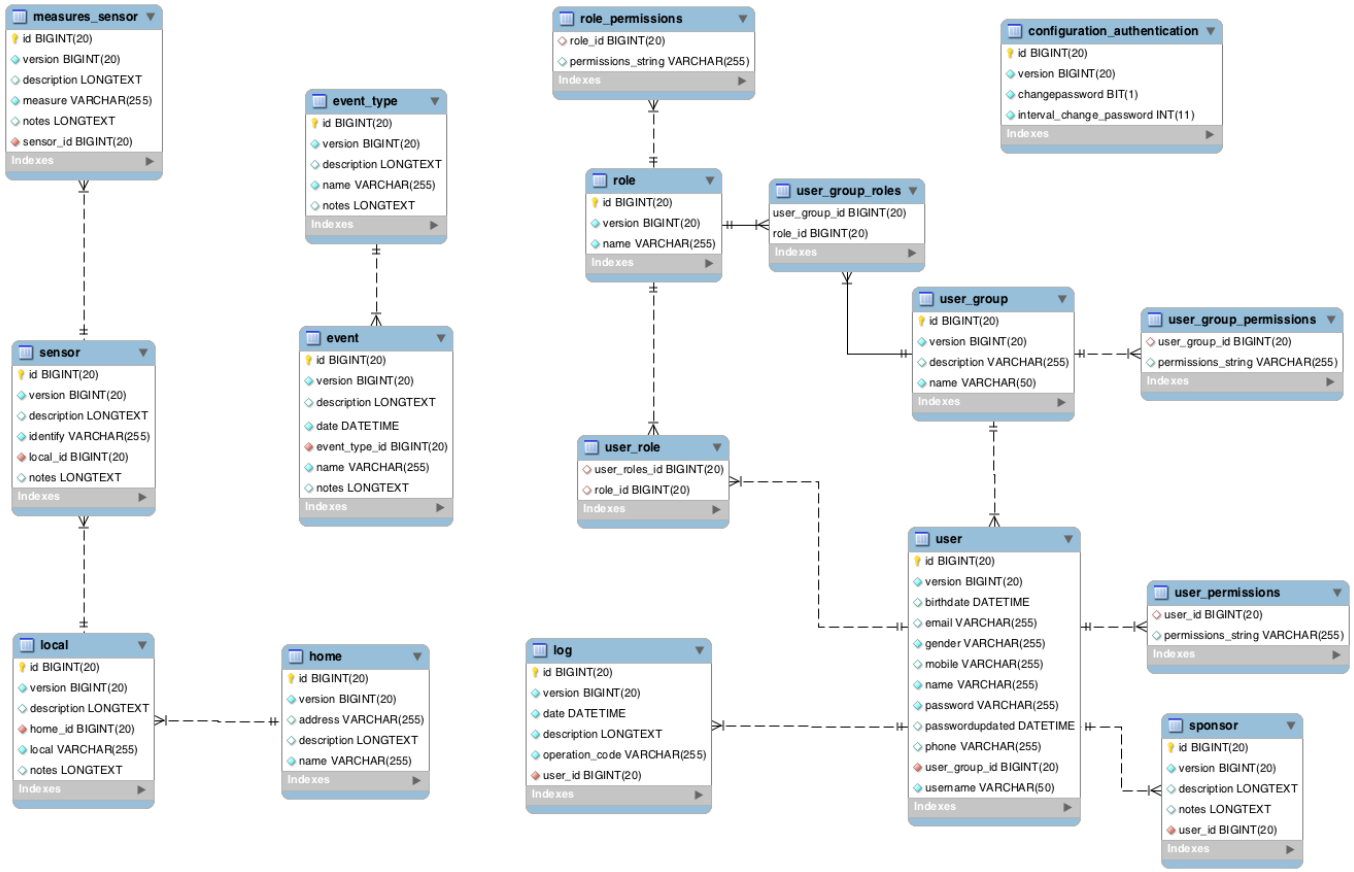


Figura 19 – Modelação da Base de Dados

Solução

5 Implementação

Neste capítulo é apresentada a construção da solução, sendo explicado detalhadamente o desenvolvimento da aplicação *web*, toda a lógica de funcionamento e a respetiva solução desenvolvida no protótipo que recolhe a informação do meio envolvente e as regras de negócio aplicadas.

5.1 Protótipo

A solução para a recolha de informação proveniente do meio envolvente passa pela utilização de um computador de pequenas dimensões em que o todo *hardware* está integrado numa placa. Esta integra pins, que são a interface ou o elo de ligação com o mundo exterior designados por *General Purpose Input/Output* (GPIO) (Raspberry Pi Foundation n.d.). São eles que permitem que a recolha do meio envolvente seja possível.

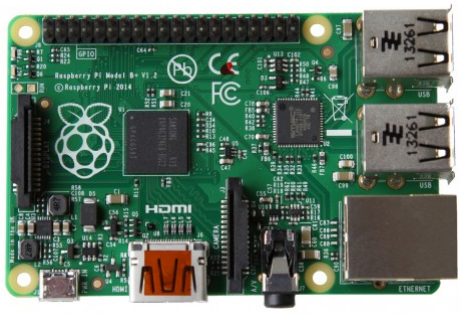


Figura 20 – Exemplo de um Raspberry Pi

Através dos sensores ligados aos respetivos GPIO, os dados do meio envolvente são recolhidos, analisados e tratados utilizando a linguagem de Phyton. Tendo em conta esta linguagem, é possível ler toda a informação proveniente do sensor e tratá-la.

O sensor PIR (figura 21) utilizado no sistema permite detetar as passagens quando um corpo quente (ser humano/animal) interceta a sua área. Em primeiro lugar, interceta uma metade do sensor de PIR, o que provoca uma mudança diferencial positiva entre as duas metades. Quando o corpo quente deixa a área de deteção, o inverso acontece, sendo que o sensor gera uma alteração diferencial negativa (figura 22). Estes pulsos são as alterações que são detetadas (System 2009).

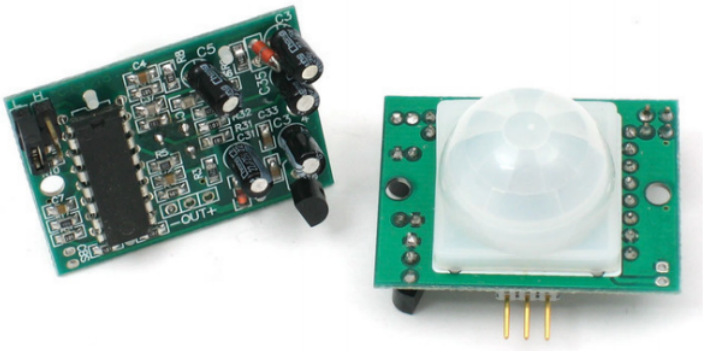


Figura 21 – Sensor PIR

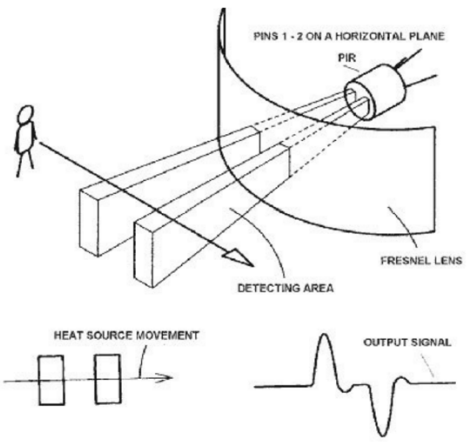


Figura 22 – Funcionamento do sensor PIR

Desta forma, em termos práticos, o sensor devolve um valor booleano, onde o valor 0 representa a ausência de deteção de movimento e o valor 1 corresponde à ocorrência de um movimento.

Após a não existência de qualquer movimento, analisado pelo PIR, os sensores de ultrassom estão aptos a recolher informações do meio envolvente. O sensor de ultrassom utilizado na solução foi o sensor HC-SR04 (Indoware 2013) (figura 23), que dispõe de entre 2 e 400cm de medição sem contacto, em que a precisão pode chegar aos 3mm. Este módulo contém um transmissor ultrassónico, um recetor e um circuito de controlo. A forma de funcionamento do sensor passa pelo envio de forma automática de um sinal (impulso) de 40kHz detetando quando o sinal regressa. Desta forma, é possível calcular a distância utilizando a seguinte fórmula:

$$D = T * \frac{V}{2}$$

, onde D representa a distância e V a velocidade do som (340m/s).

Figura 23 – Sensor de ultrassom HC-SR04

Depois da captação dos sinais vindos do meio envolvente, os algoritmos devem processar e aplicar a sua lógica de negócio, o que permite analisar toda a informação e assim tomar a decisão através dos valores recolhidos.

O protótipo tem ainda acoplado um sensor de temperatura e de humidade para recolha de informação do ambiente que permite recolher valores sobre estas duas variáveis. Desta forma, foi utilizado o sensor DHT 11 (D-robotics UK 2010) semelhante à figura 24.

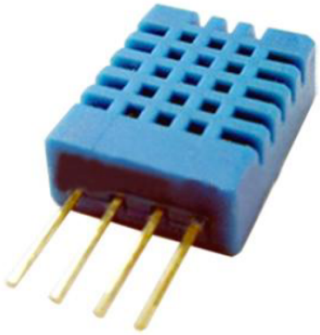


Figura 24 – Sensor de deteção de temperatura e humidade DHT 11

5.1.1 Lógica de Negócio

O protótipo (figura 25) está dotado de um algoritmo para que este seja autónomo na sua configuração. No momento em que este está a realizar essa ação (configuração), no caso de detetar que existe movimento dentro do compartimento, através do recurso ao sensor de movimento, este não permite que sejam recolhidas as leituras que servem como base para todos os cálculos com o intuito de não afetar a recolha de valores que servem de suporte para todo o funcionamento do sistema.

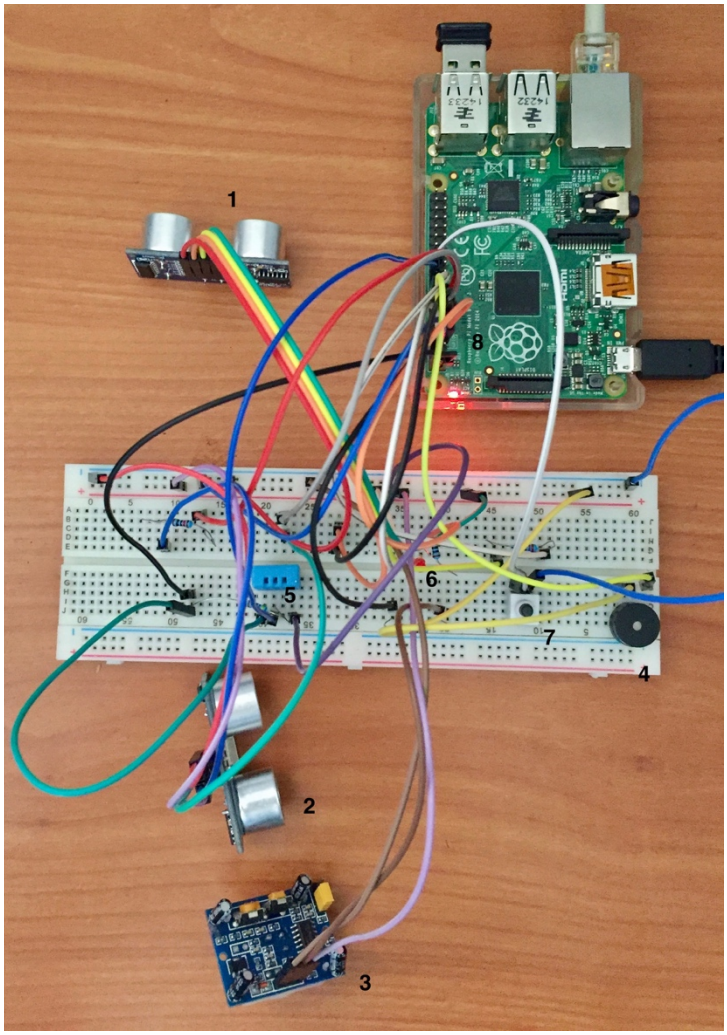


Figura 25 – Protótipo desenvolvido: 1. HC-SR04: Deteção de queda; 2. HC-SR04: Deteção de passagem na porta; 3. PIR: Sensor de movimento; 4. Buzzer; 5. DHT 11: Temperatura e Humidade; 6. LED; 7. Botão de cancelamento; 8. Raspberry-Pi

No caso de não existir movimento dentro do compartimento, o protótipo começa a autoconfigurar-se recorrendo ao sensor de ultrassom. Inicia uma recolha de medidas entre o teto e o chão onde, através do cálculo da mediana, é obtido um valor o mais aproximado possível da sua distância real ao solo.

Após esta configuração inicial, o sistema encontra-se pronto para recolher informações e detetar eventuais quedas ou agachamentos (figura 26).

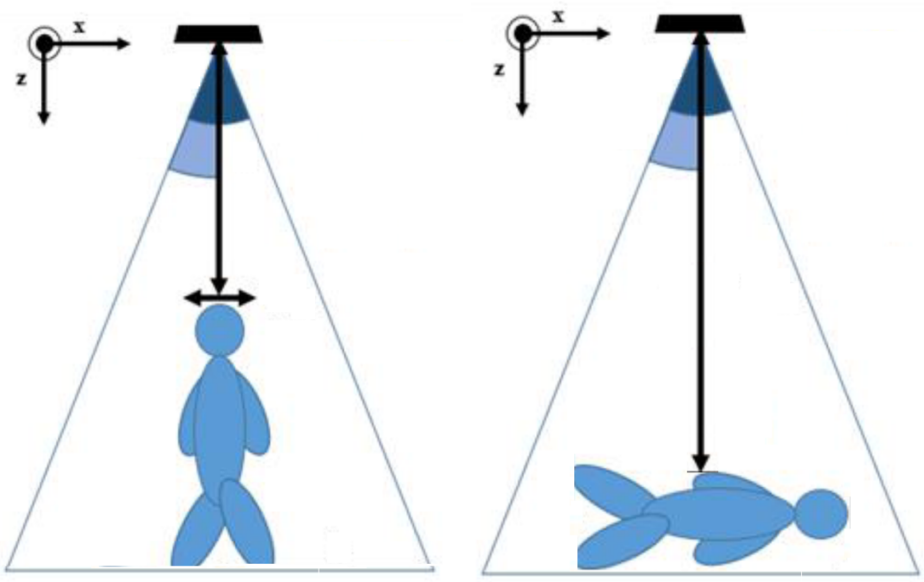


Figura 26 – Sensor de ultrassom em funcionamento (Gasparrini et al. 2014)

Com o recurso da medida do teto ao chão e com a diferença de altura que está a ser recolhida, auxiliado pelos valores do cálculo da velocidade num instante de tempo, é possível verificar se se trata de uma queda ou um agachamento. No caso de uma queda, a velocidade é acentuada, bem como a diferença de altura recolhida. No caso de um agachamento, é possível verificar a velocidade num instante de tempo (menor que numa queda) bem como a diferença de altura recolhida (também menor que numa queda). Toda a lógica, bem como as decisões implementadas no protótipo, podem ser verificadas no diagrama de decisões seguinte (figura 27).

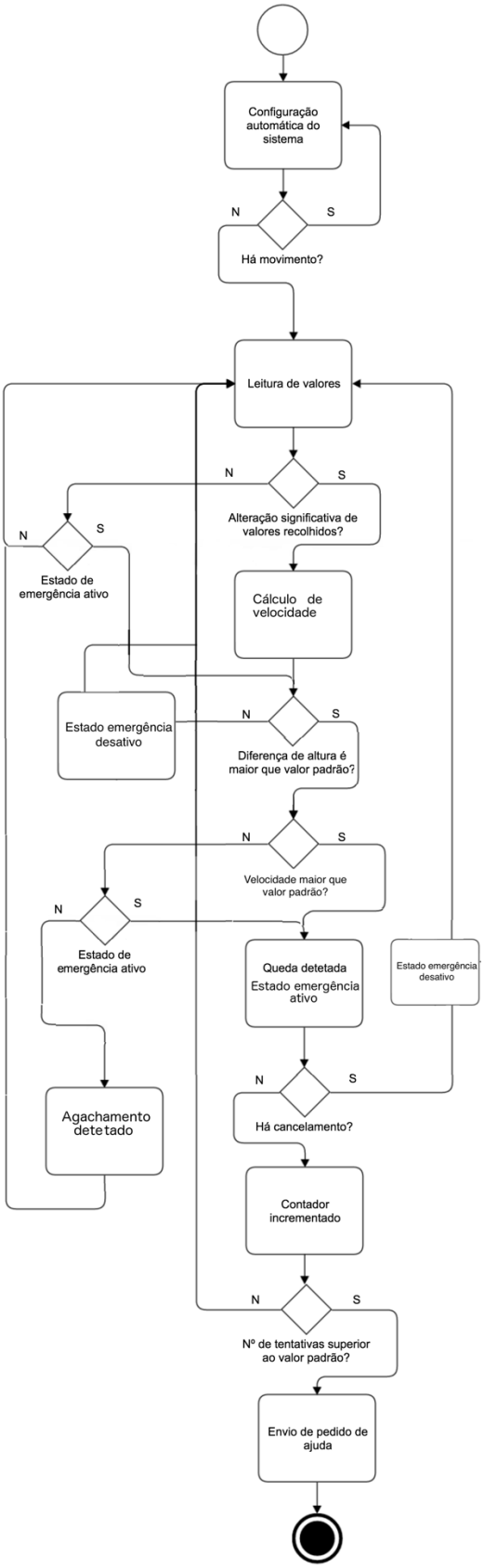


Figura 27 – Diagrama de decisões

O protótipo desenvolvido tem ainda um controlo de entradas e saídas do compartimento que permite recolher mais informações para a solução. O sensor deve ser aplicado a uma altura superior à de possíveis animais domésticos que possam existir, permitindo ao protótipo controlar a entrada de habitantes, mas ignorando animais domésticos. No caso de não existir um acesso, mas sendo detetado movimento, estamos perante um animal doméstico no compartimento. É colocado um sensor de ultrassom na entrada do compartimento e assim que existir uma redução de comprimento medido, estamos perante uma passagem na porta. Esta informação é complementada com o sensor de movimento para recolher também ele informação relevante para a tomada de decisão.

Em termos de comunicação, a mesma é implementada através da tecnologia RabbitMQ, onde a versão do RabbitMQ Server utilizada é a 3.6.1. O envio da informação é feito através da declaração da ligação ao servidor da tecnologia (*Connection String* ao servidor) indicando o endereço IP do mesmo (código 1). A comunicação é possível através da importação da biblioteca designada por *pika* (Garnock-Jones & M. Roy n.d.) que é uma implementação pura do protocolo *Advanced Message Queuing Protocol* (AMQP) (OASIS n.d.) para Python, e que permite simplificar toda a comunicação entre a linguagem e o servidor de mensagens (RabbitMQ Server).

```
import pika
import RPi.GPIO as GPIO
import time
import statistics
import Adafruit_DHT

connection =
pika.BlockingConnection(pika.ConnectionParameters(host='192.168.1.3'))
channel = connection.channel()
```

Código 1 – Ligação do protótipo ao servidor de mensagens (RabbitMQ)

As mensagens provenientes do protótipo desenvolvido em Pyhton passam pelo servidor intermediário que corre o serviço de mensagens (RabbitMQ Server) e são envidas para o servidor com a lógica de negócio e com a aplicação *web*.

Com o canal em aberto, apenas é necessário enviar as mensagens com o formato mais apropriado.

```
channel.basic_publish(exchange='', routing_key='queueMessage',body= msg)
channel.basic_publish(exchange='', routing_key='queueMessage',body= 'HELP')
channel.basic_publish(exchange='', routing_key='queueMessage',body= 'SQUAT')
```

Código 2 – Exemplo do envio da mensagem com a medida recolhida e pedido de ajuda

No que diz respeito ao servidor que contém todo o armazenamento da informação e a aplicação *web*, é declarada a ligação à fila que recebe as mensagens e as trata, onde, sempre que este servidor é inicializado, um serviço fica responsável para receção das mensagens provenientes do RabbitMQ Server. Neste serviço (código 3) é solicitado o envio de e-mails caso exista uma mensagem com o valor "*HELP*". No caso de receção do valor "*SQUAT*" verifica-se que existe um agachamento e é adicionado esse mesmo valor à tabela de eventos ocorridos. Pode ainda receber as medidas recolhidas pelos sensores, isto é, recebe do sensor de ultrassom a distância do teto ao solo quando se configurou e recebe leituras de temperatura e humidade ao longo do tempo.

```
@Transactional
class MessageService {
    static rabbitQueue = 'queueMessage'

    void handleMessage(byte[] message) {
        String value = new String(message, "UTF-8");

        if(value == "HELP"){
            Event event = new Event()
            def eventType = EventType.findById(1)

            def currentdate = new Date().format('yyyy-MM-dd HH:mm:ss')

            event.name = "Queda"
            event.date = new Date().parse('yyyy-MM-dd HH:mm:ss', currentdate)
            event.eventType = eventType

            event.save flush:true

            def sponsor = Sponsor.findAll()
            for(int i = 0; i < sponsor.size(); i++){
                def email = sponsor[i].user.email
                def sub = "FallDetection+: Queda detetada!"

                sendMail {
                    multipart true
                    to email
                    subject sub
                    html (view:"/alertEmail/alert")
```

```
                    inline 'fallDetection', 'image/jpg', new File('./web-
app/homepage/images/logo-black.png')
                }
            }
        }else if(value == "SQUAT"){
            Event event = new Event()
            def eventType = EventType.findById(2)

            def currentdate = new Date().format('yyyy-MM-dd HH:mm:ss')

            event.name = "Agachamento"
            event.date = new Date().parse('yyyy-MM-dd HH:mm:ss', currentdate)
            event.eventType = eventType

            event.save flush:true
        }else{
            def map = value.split("-");
            for (int i = 0; i < map.size()-1 ; i++) {
                def sensor = Sensor.findByIdentify(Long.parseLong(map[i]))

                if(sensor != null){
                    MeasuresSensor measures = new MeasuresSensor()
                    measures.sensor = sensor
                    measures.measure = map[i+1]

                    measures.save flush:true
                }

                i++
            }
        }
    }
}
```

Código 3 – Serviço para tratamento das mensagens e envio de e-mail

No caso do valor absoluto na diferença entre o valor que foi retirado aquando foi feita a calibração e a distância lida nesse instante ser maior que a diferença calculada do chão ao teto, então significa que a pessoa está em pé ou se levantou e, consequentemente, o alarme pode ser desativado. No caso desse mesmo valor ser maior que o valor mínimo de altura pré-definido em sistema, significa que existiu um movimento que pode ser considerado uma queda e é necessário verificar outras condições que permitam validar esse mesmo acontecimento. No caso do valor mínimo pré-definido em sistema, este permite que, caso um utilizador esteja no chão, seja verificada essa mesma condição. Assim, o valor pré-definido em sistema é de 80cm, o que representa que um utilizador se encontra caído, estando o sensor a detetar essa mesma altura mínima. É verificado se o valor absoluto da velocidade é superior à velocidade máxima considerável para deteção de queda. Esse valor é pré-definido em sistema ou caso já tenha sido detetada uma queda anteriormente, indica que existe então, de facto, alguém caído no chão, e como tal, necessita de um pedido de ajuda, incrementando o

contador do número de tentativas com o intuito de verificar se o utilizador se consegue levantar e recompor, antes de existir um pedido de ajuda. Uma queda é considerada como um acontecimento com uma força igual ou inferior a 2.5G, sob o pressuposto de que os habitantes são incapazes de executar uma ação acima desse limiar (Abbate et al. 2010). Desta forma, é possível calcular a velocidade utilizando o seguinte cálculo:

$$V = \sqrt{2 * G * H}$$

, onde V representa a velocidade, G a Força-G (9.81 m/s^2) e H representa a altura (em metros).

Desta forma, uma queda pode ser detetada, mas pode não ser necessário acionar um alarme sobre a mesma. Caso se verifique que o número de tentativas até acionar o alerta seja ultrapassado, estamos perante uma queda em que o habitante não se conseguiu levantar e são acionadas medidas que permitam sinalizar o acontecimento, isto é, uma sirene (*buzzer*) emite um som intermitente e uma lâmpada (*led)* piscará para alertar eventuais pessoas por perto.

```
if abs(calibration - distance) >= minHeight and abs(calibration - distance) <=
maxHeight:
    if abs(velocity) > maxVelocity or someBodyOnFloor == True:
        if pirValue == 0: # alguém no chão pela altura e não se mexe;
            someBodyOnFloor = True
            countFalls+=1
        elif pirValue == 1:
            countFalls+=1

        GPIO.setmode(GPIO.BCM)
        GPIO.setup(buzzer, GPIO.OUT)
        GPIO.output(buzzer, True)
        time.sleep(0.5)
        GPIO.output(buzzer, False)
        GPIO.setup(4, GPIO.OUT)
        GPIO.output(4,GPIO.HIGH)
        time.sleep(0.2)
        GPIO.output(4,GPIO.LOW)

        if countFalls == attempts:
            GPIO.output(buzzer, True)
            time.sleep(0.1)
            GPIO.output(buzzer, False)
            time.sleep(0.1)
            GPIO.output(buzzer, True)
            time.sleep(0.1)
            GPIO.output(buzzer, False)
            GPIO.output(buzzer, True)
            time.sleep(0.1)
            GPIO.output(buzzer, False)
            time.sleep(0.1)
```

```
            GPIO.output(buzzer, True)
            time.sleep(0.1)
            GPIO.output(buzzer, False)

        channel.basic_publish(exchange='', routing_key='queueMessage',body=
'HELP')
        countFalls = 0

    else:
        if countSquad == attemptsSquad:
            channel.basic_publish(exchange='', routing_key='queueMessage',body=
'SQUAT')
            countSquad = 0
    countSquad+=1
    someBodyOnFloor = False
else:
    someBodyOnFloor = False
    countFalls = 0
```

Código 4 – Lógica de negócio no pedido de ajuda

O sistema está ainda dotado de uma função (código 5) que ao longo do tempo vai recolhendo informações de temperatura e humidade, com o intuito de armazenar informação sobre o estado do meio envolvente aquando a leitura dos sensores. Tratando-se de uma casa de banho, temperatura e humidade são dois fatores que se alteram com maior frequência, principalmente nos momentos do banho.

```
def tempsensor():
    umid, temp = Adafruit_DHT.read_retry(sensor, pino_sensor)

    if umid is not None and temp is not None:
        print ("Temperatura = {0:0.1f}  Humidade = {1:0.1f}\n").format(temp,
umid)
        msg = str(pino_sensor) + "-T: " + str(temp) + " H: " + str(umid)
        channel.basic_publish(exchange='', routing_key='queueMessage',body=
msg)
    else:
        print ("Erro a ler os dados do sensor de temperatura e humidade")
```

Código 5 – Função de leitura de temperatura e humidade

Esta função utiliza uma biblioteca *open-source* designada de Adafruit_DHT (Adafruit Industries n.d.) desenvolvida para a linguagem Python, que permite recolher a informação de uma forma mais simples.

5.2 Grupo de Utilizadores

Os utilizadores estão agrupados em diferentes tipos e cada um deles tem associado a si diferentes permissões de acesso. Assim, dois tipos de grupos estão contemplados no sistema: o grupo de administração e o grupo de utilizadores.

As restrições de acesso do grupo de administração são nulas levando a que este grupo tenha permissões para adicionar novos tipos de ações, bem como fazer a própria manutenção e gestão do sistema via interface *web* ou RestFul. Pode ainda configurar um novo sistema, adicionar novos compartimentos a serem monitorizados, realizar a associação de sensores aos compartimentos, gerir tipos de eventos e registar novos responsáveis e respetivos contactos para serem notificados de eventuais problemas detetados (quedas).

O grupo de utilizadores é então um grupo com menores privilégios. O sistema adota a filosofia de redução dos privilégios, sendo que, cada grupo tem os privilégios necessários para executar as tarefas supostas e nada mais. Desta forma, um utilizador que não tenha permissões de acesso para realizar determinada ação na aplicação *web,* não tem também permissões de acesso a essa informação via serviços RestFul. Assim, as permissões de acesso são transversais, independentemente da forma como se tenta aceder, criar, editar ou remover informação da aplicação.

5.3 Serviços RestFul

Os serviços RestFul encontram-se implementados com o objetivo de trocar mensagens com a utilização de *JavaScript Object Notation* (JSON) (w3schools n.d.) ou *eXtensible Markup Language* (XML) (w3schools n.d.), sendo que a solução contempla ambos os formatos para esse efeito. Este serviço tem como base fornecer a informação necessária para uma aplicação móvel e outros serviços relevantes para futuras melhorias e/ou expansões do sistema. Só é possível fazer pedidos de informação via RestFul caso o utilizador faça o pedido com as suas credenciais de acesso, o que permite uma maior segurança sobre os dados armazenados. Toda a aplicação está dotada destes serviços para todas as ações, isto é, é possível ler (*GET*), adicionar (*POST*), editar (*PUT*) e remover (*DELETE*) toda a informação. Desta forma, qualquer

aplicação, seja ela *web*, móvel ou outra pode tratar e aceder à informação (com as respetivas permissões associadas a cada grupo de utilizador), dando uma maior escalabilidade ao projeto.

Para efetuar o respetivo *login* basta utilizar as credenciais de acesso, como exemplificado na figura seguinte.

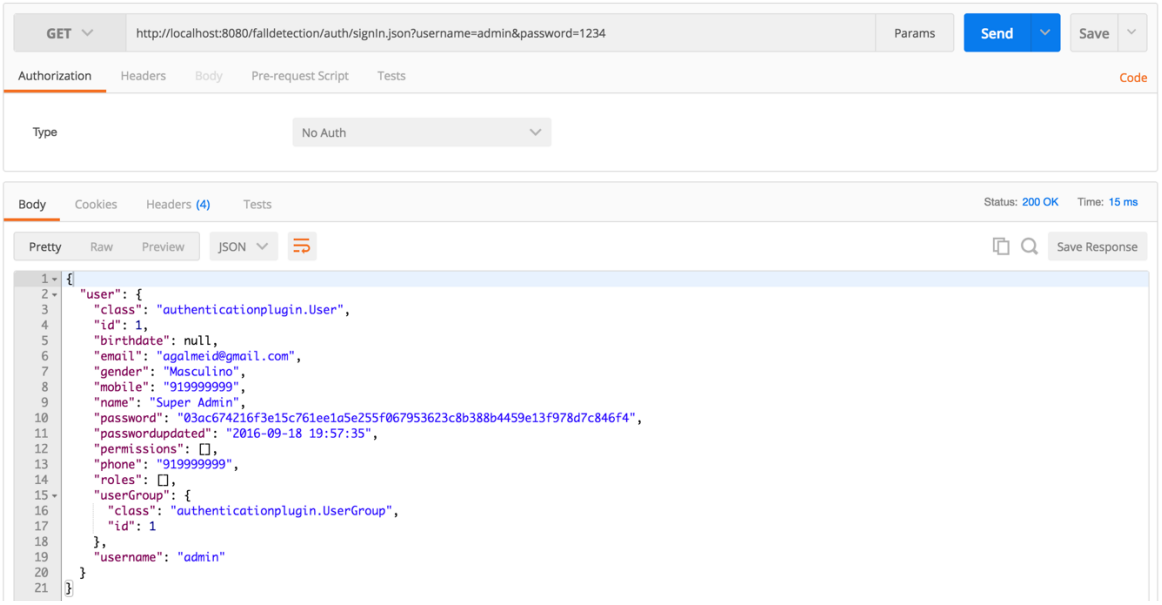


Figura 28 – Login com sucesso via *webservice*: resposta em formato JSON

No caso da autenticação falhada, o servidor não retorna qualquer informação referente ao utilizador:

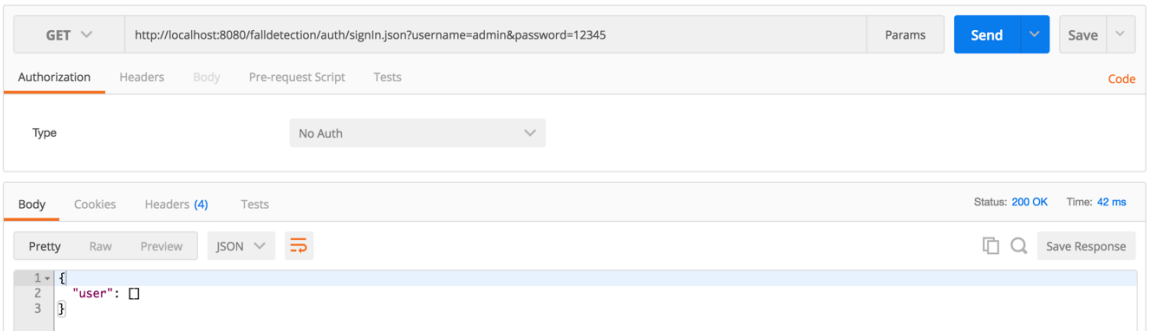


Figura 29 – Login sem sucesso via *webservice*: resposta em formato JSON

No que toca às restantes funcionalidades (ler, adicionar, editar e remover) para cada tipo de modelo (*model* ou *domain*) (sensor, medidas de sensor, habitação, compartimentos, eventos, tipos de eventos, responsáveis, etc.), o utilizador poderá executar a ação pretendida caso esteja autenticado previamente e tenha permissões para as funcionalidades. A título de exemplo, em seguida são apresentadas as funcionalidades para uma habitação, sendo que o processo é similar em cada tipo de modelo.

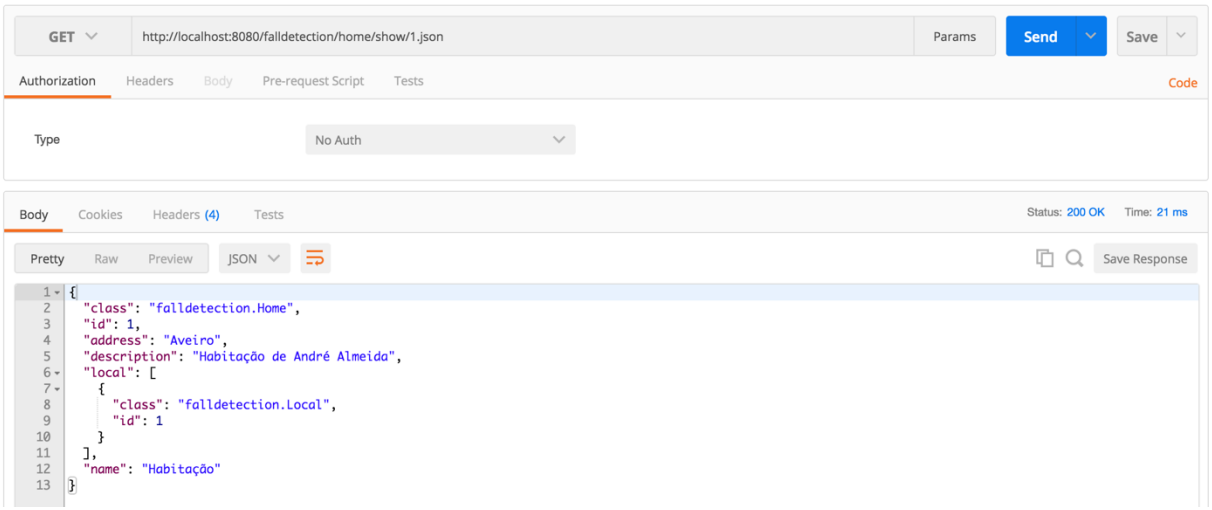


Figura 30 – Leitura de dados da habitação: resposta em formato JSON

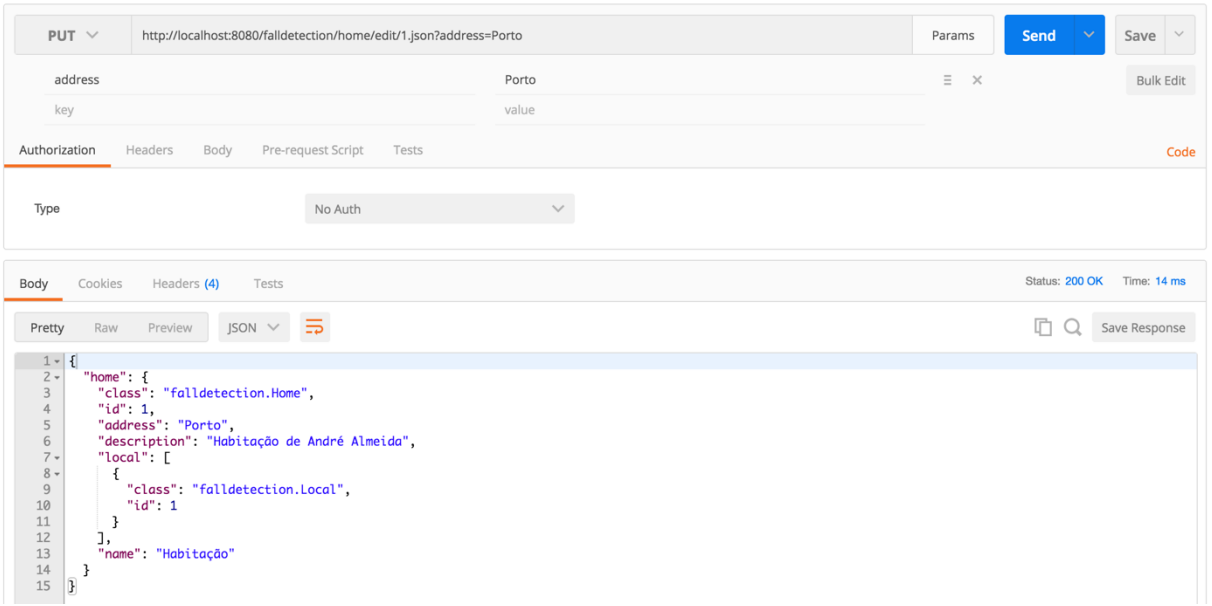


Figura 31 – Edição de dados da habitação: resposta em formato JSON

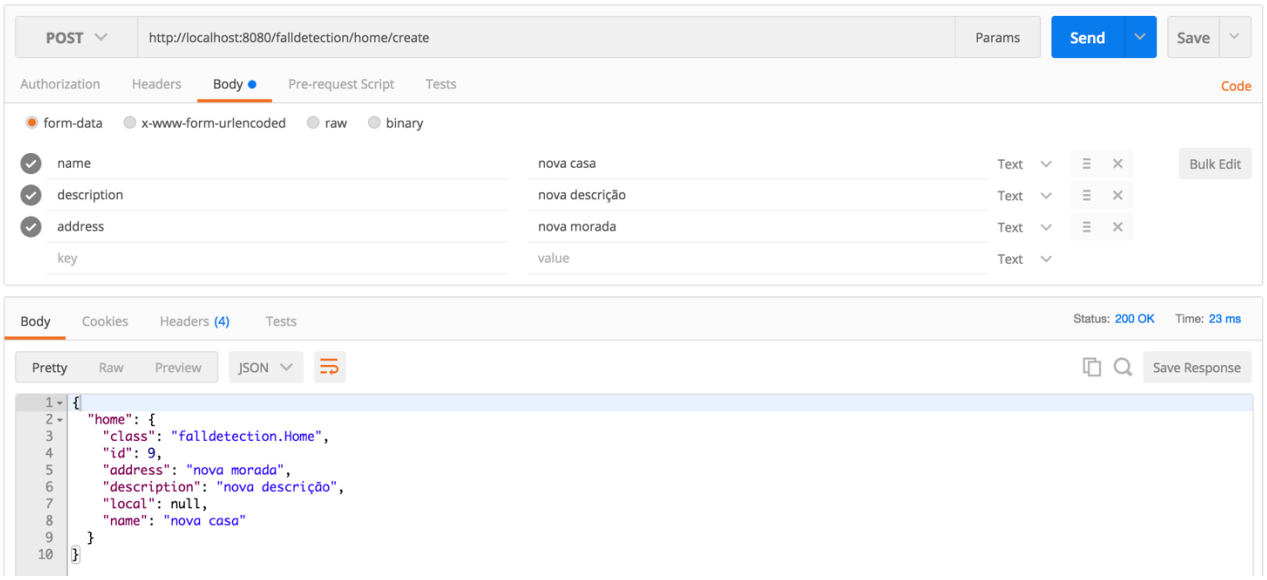


Figura 32 – Criação de uma habitação: resposta em formato JSON

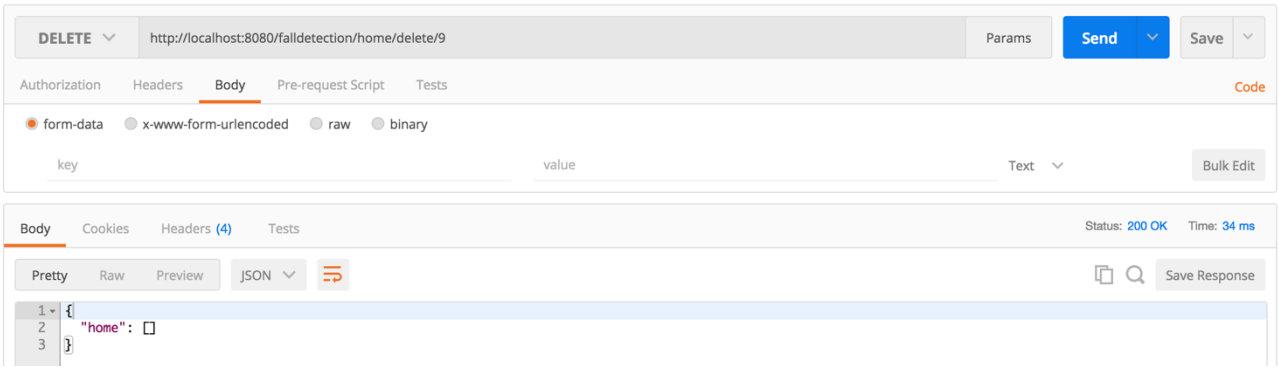


Figura 33 – Remoção de uma habitação: resposta em formato JSON

Toda a interação com os *webservices* da aplicação encontra-se em anexo, para consulta mais aprofundada.

6 Avaliação à Solução

Neste capítulo são apresentados os testes efetuados à solução com o intuito de avaliar e analisar eventuais problemas desta face a outros sistemas que se possam equiparar.

6.1 Grandezas para Avaliação

Na recolha da informação destaca-se a necessidade da qualidade dos dados para que exista uma amostra representativa e com precisão elevada. Numa primeira fase, é indispensável definir os objetivos das hipóteses a testar. Deve também ser definido o universo, seguido da seleção da técnica de amostragem que mais se adeque aos dados e tipo de amostra que está a ser recolhida. O método de recolha de dados utilizado consiste na simulação de eventos e recolha do grau de acerto do mesmo.

Um teste deve recair no número e grau de acerto sobre os falso-positivos, isto é, deve ser feito um estudo em que seja analisado o número de falso-positivos reportado pelo sistema, com o objetivo de definir o quão precisos são os algoritmos criados para analisar essas mesmas ocorrências. Desta forma, oito grandes testes foram efetuados, sendo eles:

1. Grau de acerto na deteção de queda;
2. Grau de acerto na deteção de agachamento;
3. Grau de acerto na deteção de queda num ambiente com humidade;
4. Grau de acerto na deteção de um agachamento num ambiente com humidade;

5. Grau de acerto na deteção de movimento;
6. Grau de acerto na deteção de movimento num ambiente com humidade;
7. Grau de acerto na entrega de mensagens;
8. Grau de acerto na entrega de mensagens num ambiente com humidade.

Com estes testes, é possível concluir a precisão dos sensores e dos algoritmos desenvolvidos para cada tipo de evento em diferentes tipos de ambiente (com e sem humidade). Existe a necessidade de criar dois testes com um grau de humidade mais elevado devido ao facto de a solução ser projetada para detetar quedas em casas de banho (embora funcione noutros compartimentos), onde, durante e após um banho, é gerada uma maior quantidade de vapor de água e humidade, levando a que as mesmas possam afetar o grau de acerto e de precisão dos sensores. Assim, estes dois testes com humidade foram adicionados para obter resultados com condições menos favoráveis para os sensores.

6.2 Resultados dos Testes Aplicados

A avaliação é dividida em oito grandes testes, como referidos anteriormente. Segundo a fórmula de cálculo do número de amostras:

$$N = \frac{\frac{Z\alpha}{2} * \sqrt{p * q}}{E}$$

, onde N representa o tamanho da amostra, $\frac{Z\alpha}{2}$ o valor crítico para o grau de confiança (95% corresponde ao valor 1.96), *p* a proporção de resultados favoráveis da variável na população, *q* a proporção de resultados desfavoráveis na população (*q*=1-*p*) e *E* o erro padrão, normalmente 5% (Kerns 2010) (Miot 2011).

Assim, é necessário que o universo seja composto por 385 amostras (leituras) para cada tipo de teste, onde é definido o grau de acerto para cada tipo de evento. Com 385 amostras, consegue-se atingir um nível de confiança de 95% e uma margem de erro de 5%. Os dados são classificados em três tipos, sendo eles, queda, agachamento ou não detetado para o caso dos testes às condições de queda e agachamento. No caso do movimento, são classificados em

dois tipos, sendo eles detetado e não detetado e no caso das mensagens dividido em dois tipos, entregue e não entregue.

6.2.1 Deteção de queda – sem humidade

No teste de deteção de queda sem humidade, verificou-se que em 385 amostras, 367 foram consideradas quedas efetivas, 16 foram consideradas como agachamento e 2 não foram detetadas como se pode verificar na figura 34:

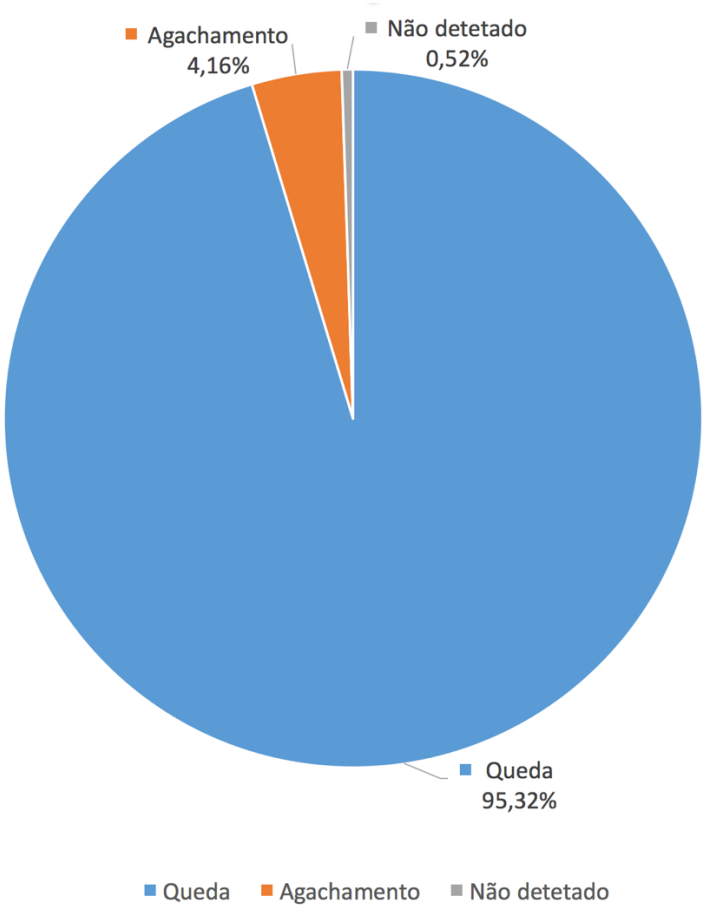


Figura 34 – Conclusão dos testes na deteção de queda

Pode-se então concluir que no teste de deteção de queda em ambiente sem humidade, em termos percentuais, num universo de 385 amostras, o grau de acerto é de 95.32%.

6.2.2 Deteção de agachamento – sem humidade

No teste de deteção de agachamento sem humidade, verificou-se que em 385 amostras, 339 foram consideradas agachamentos efetivos e 46 foram consideradas quedas, conforme se pode verificar na figura 35:

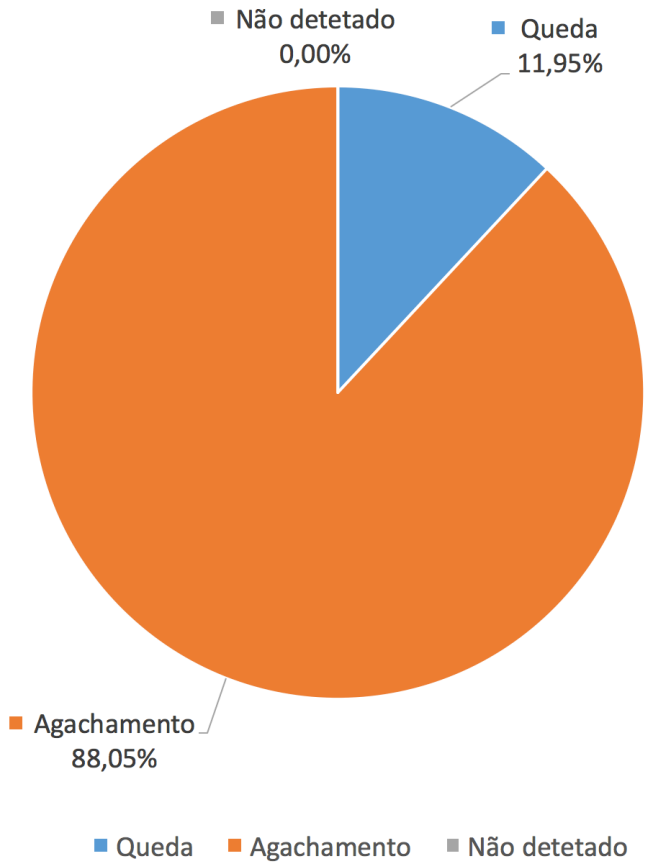


Figura 35 – Conclusão dos testes na deteção de agachamento

Pode-se então concluir que no teste de deteção de agachamento em ambiente sem humidade, em termos percentuais, num universo de 385 amostras, o grau de acerto é de 88.05 %.

6.2.3 Deteção de queda – com humidade

No teste de deteção de agachamento com 90% de humidade, verificou-se que em 385 amostras, 363 foram consideradas quedas efetivas, 20 foram considerados agachamentos e 2 amostras não foram detetadas, como se pode verificar na figura 36:

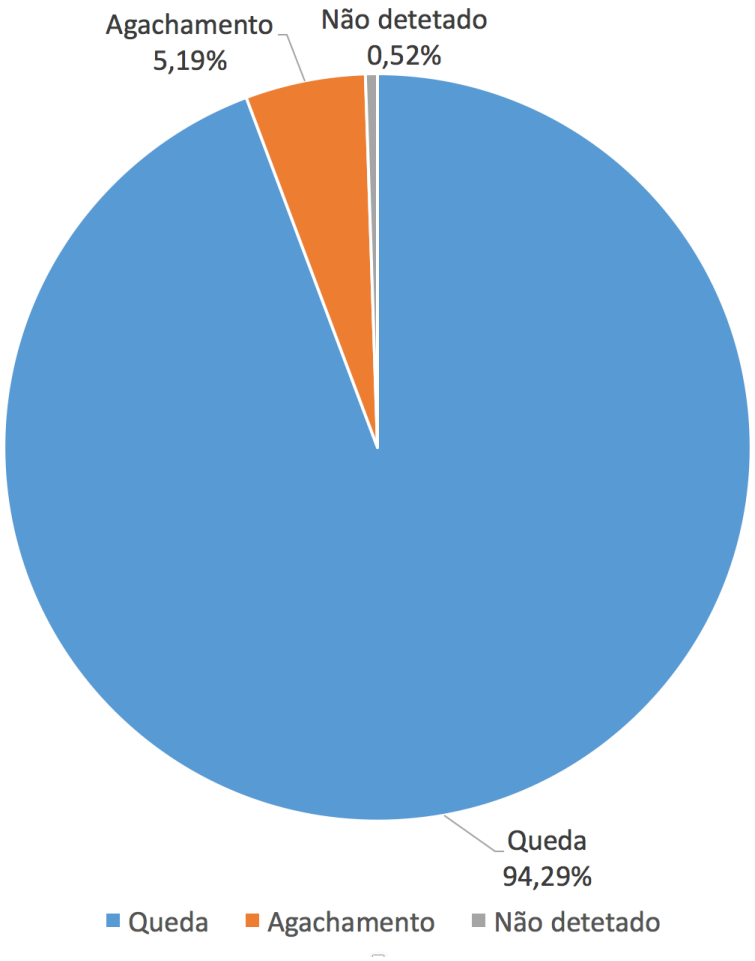


Figura 36 – Conclusão dos testes na deteção de queda (com humidade a 90%)

Pode-se então concluir que no teste de deteção de quedas em ambiente com humidade, em termos percentuais, num universo de 385 amostras, o grau de acerto é de 94.29%. Comparativamente com os testes efetuados sem humidade considerável, verifica-se que o grau de acerto ronda valores muito próximos (94.29% comparativamente a 95.32%).

6.2.4 Deteção de agachamento – com humidade

No teste de deteção de agachamento com 90% de humidade, verificou-se que em 385 amostras, 319 foram consideradas agachamentos efetivos, 60 foram consideradas como queda e 6 amostras não foram detetadas, como se pode verificar na figura 37:

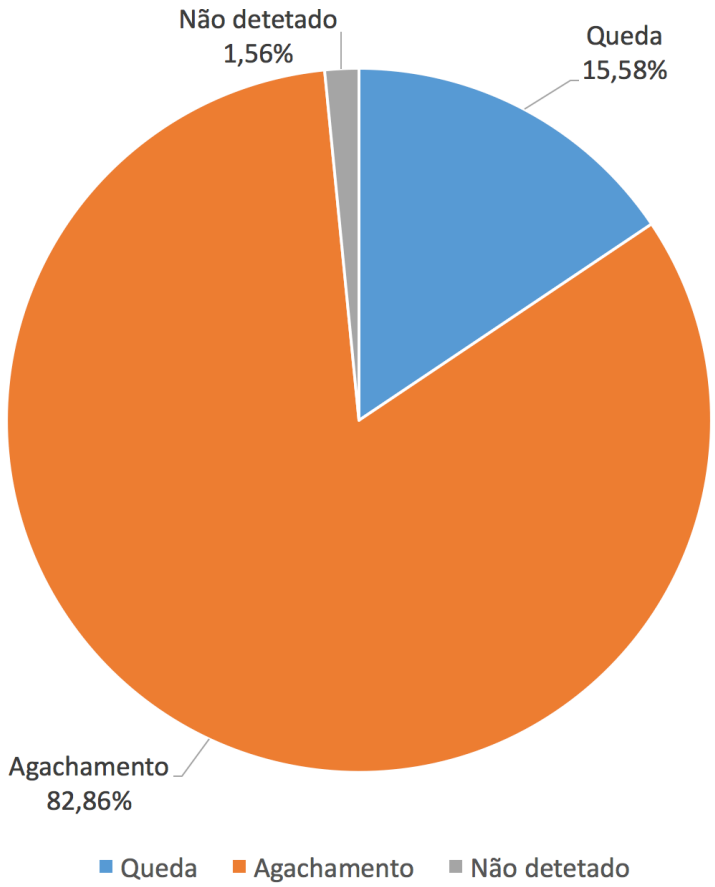


Figura 37 – Conclusão dos testes na deteção de agachamento (com humidade a 90%)

Pode-se então concluir que no teste de deteção de agachamento em ambiente com humidade, em termos percentuais, num universo de 385 amostras, o grau de acerto é de 82.86%. Comparativamente com os testes efetuados sem humidade considerável, verifica-se que o grau de acerto ronda valores com uma diferença percentual de 5.19% (82.86% comparativamente a 88.05%).

6.2.5 Deteção de movimento – sem humidade

No teste de deteção de movimento sem humidade verificou-se que em 385 amostras, 354 detetaram movimento e 31 não detetaram qualquer movimento, conforme se pode verificar na figura 38:

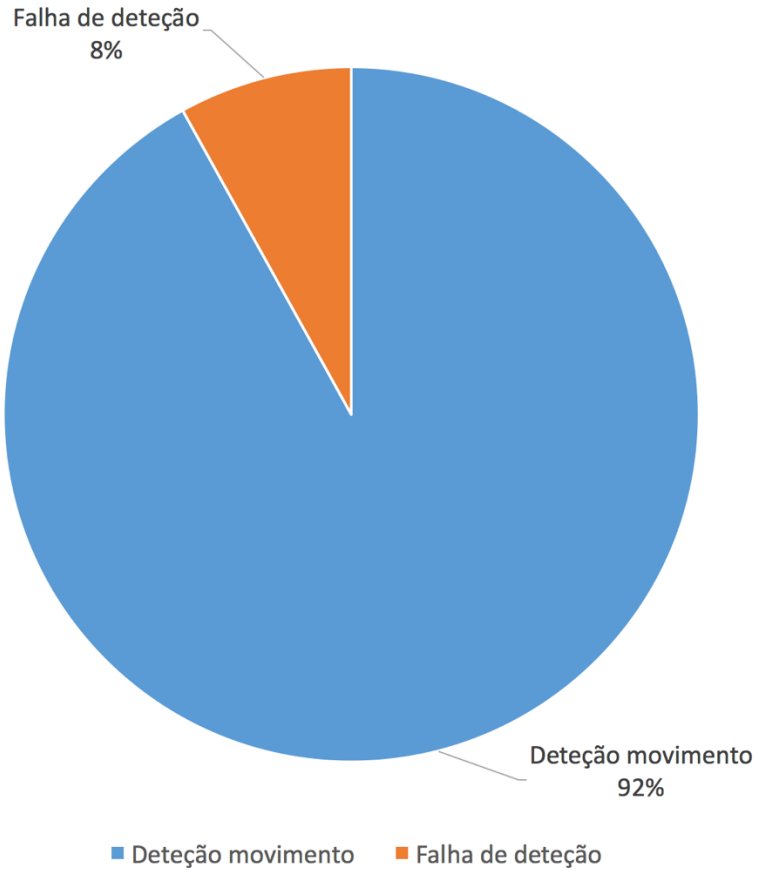


Figura 38 – Conclusão dos testes na deteção de movimento

Pode-se então concluir que no teste de deteção de movimento em ambiente sem humidade, em termos percentuais, num universo de 385 amostras, o grau de acerto é de 92%.

6.2.6 Deteção de movimento – com humidade

No teste de deteção de movimento com 90% de humidade, verificou-se que em 385 amostras, 312 foram considerados movimentos e 73 não detetaram qualquer movimento, como se pode verificar na figura 39:

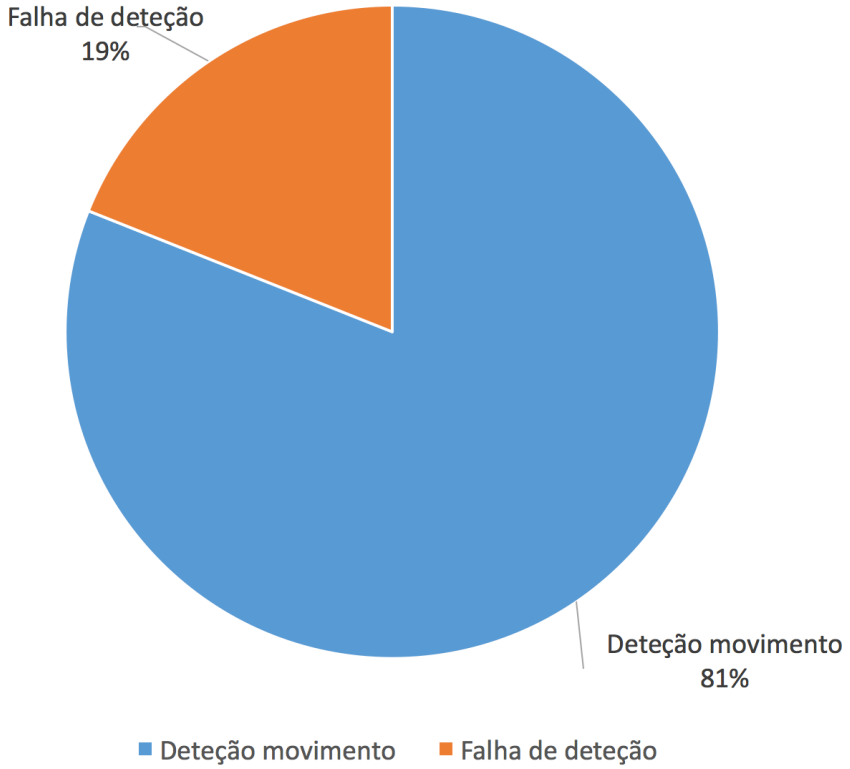


Figura 39 – Conclusão dos testes na deteção de movimento (com humidade a 90%)

Pode-se então concluir que no teste de deteção de movimento em ambiente com humidade, em termos percentuais, num universo de 385 amostras, o grau de acerto é de 81%. Comparativamente com os testes efetuados sem humidade considerável, verifica-se que o grau de acerto ronda valores com uma diferença percentual de 11% (81% comparativamente a 92%).

6.2.7 Mensagens

Nos testes de envio e receção de mensagens por parte do protótipo até ao servidor – também ele testado – num universo constituído por 385 amostras, a totalidade foi corretamente entregue (100%).

Os testes foram realizados em rede cablada – protocolo 802.3 (Society 2012a), obtendo a entrega total das mensagens, como se pode verificar na figura 40:

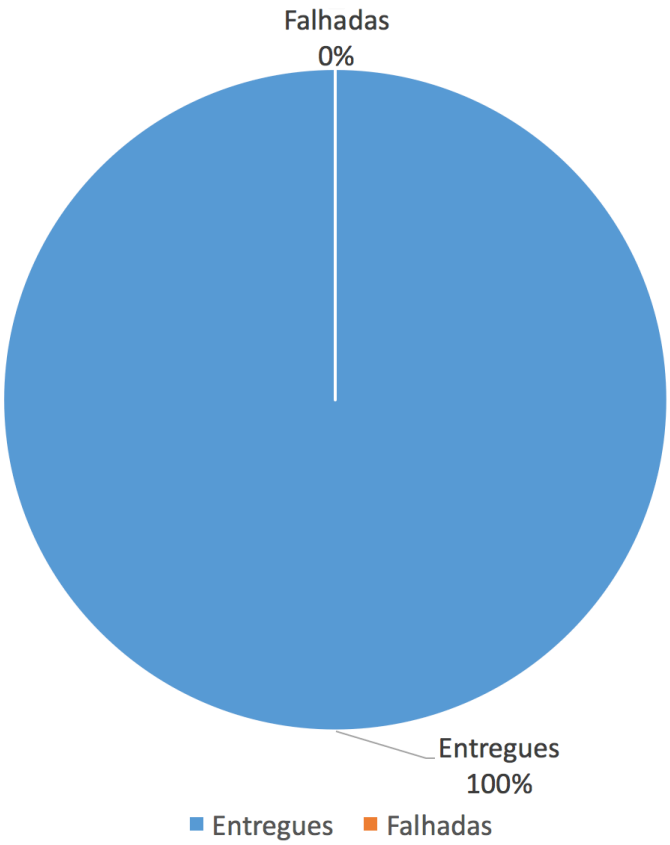


Figura 40 – Grau de acerto na troca de mensagens

7 Conclusões

7.1 Síntese do Trabalho

Com o desenvolvimento desta solução é possível analisar e compreender a grande área em que esta se encontra inserida, neste caso a domótica, bem como todas as dificuldades inerentes à mesma. O facto de a solução ser autónoma, não dependendo de objetos que o utilizador deve transportar para ser detetada a queda, acrescenta valor, não correndo o risco de perda do objeto ou esquecimento do mesmo.

Uma das grandes dificuldades sentidas no desenvolvimento da solução passa pela deteção de queda, onde, através da leitura de informações provenientes do meio envolvente, nem sempre é simples chegar à conclusão de que de facto existe uma queda. Esta pode ser interpretada de diferentes formas, sendo que nem todas as quedas ocorrem de igual modo, o que aumenta a complexidade na deteção das mesmas.

Pode afirmar-se que uma queda e um agachamento são situações similares, com variáveis que podem levar a diferentes conclusões. Esta solução permite detetar ambos os casos, dentro dos parâmetros lidos pelos sensores. No entanto, como referido anteriormente, uma queda pode ocorrer em diferentes circunstâncias e com diferentes formas, pelo que a solução pode não conseguir detetar a mesma. Um utilizador do sistema pode ter uma queda com uma velocidade mais lenta comparativamente a outras, pelo simples facto de, por exemplo, se apoiar em algum objeto em seu redor, reduzindo a velocidade da mesma, levando a que o sistema possa não a conseguir detetar.

Numa análise aos testes efetuados, verifica-se que a humidade é um fator preponderante na recolha dos dados, mas não causando uma diferença elevada, em termos percentuais. Já no que ao sensor de movimento diz respeito, a diferença percentual é significativa. Com esta solução, quanto maior for o compartimento em que se encontra instalada, maior será o número de sensores de ultrassom necessários para abranger toda a área. Com o seu aumento, existe a necessidade de o algoritmo ser capaz de lidar com mais sensores e de os mesmos não enviarem os sinais em simultâneo, com o intuito de não gerar más leituras por parte dos sensores de ultrassom (HC-SR04).

Em termos de objetivos inicialmente estabelecidos, os mesmos foram atingidos com sucesso, levando a que a solução final seja uma solução que deteta quedas e agachamentos, podendo auxiliar os utilizadores a terem uma melhoria na qualidade de vida e proporcionar uma maior segurança para os mesmos.

7.2 Melhorias Futuras

No que diz respeito a melhorias futuras, esta solução pode ser aprimorada com a adição de mais sensores, possibilitando a otimização da deteção de quedas e permitindo que as informações possam ser recolhidas com maior precisão, visto que existem mais sensores a recolherem a mesma do meio envolvente. Um exemplo dessa melhoria passa pela inserção de um detetor térmico como o Omron D6T (Corporation n.d.), permitindo que se consiga aprimorar ainda mais a solução com a deteção de calor. Desta forma, será possível perceber-se com maior detalhe se de facto estamos perante um ser vivo (pessoa ou animal doméstico) ou a mudança da disposição de mobiliário. Permite então que os algoritmos se possam reajustar sempre que não detetem pessoas no compartimento, levando o sistema a se autoconfigurar e recolher informações pertinentes para melhorar a solução final.

Em termos de interação com o utilizador, pode ainda ser desenvolvida uma aplicação móvel que permita comunicar a informação para uma aplicação, visto que a solução atual já está dotada de *webservices* que permitem a troca de informação para outras aplicações.

8 Bibliografia

Abbate, S., Avvenuti, M. & Corsini, P., 2010. Monitoring of human movements for fall detection and activities recognition in elderly care using wireless sensor network: a survey. *Wireless Sensor Networks ApplicationCentric Design*, pp.1–20. Available at: http://media.csee.ltu.se/courses/M7012E_2011_LP2_Pervasive_Computing/hotpapers/R25 Avvenuti - Monitoring of human movements for fall detection.pdf.

Adafruit Industries, Adafruit_Python_DHT. Available at: https://github.com/adafruit/Adafruit_Python_DHT [Accessed September 28, 2016].

Apache Cordova, Apache Cordova. Available at: http://cordova.apache.org/ [Accessed February 20, 2016].

Apache Groovy, The Groovy programming language. Available at: http://www.groovy-lang.org/ [Accessed October 10, 2016].

Apple, Apple Store na App Store. Available at: https://itunes.apple.com/pt/app/apple-store/id375380948?mt=8 [Accessed October 10, 2016].

Arduino, Arduino - Home. Available at: https://www.arduino.cc/ [Accessed September 8, 2016].

Bardram, J.E., 2005. The Java Context Awareness Framework (JCAF)–a service infrastructure and programming framework for context-aware applications. *Pervasive Computing*, (April), pp.98–115. Available at: http://www.springerlink.com/index/YL2FEN8CLQQWQ2TB.pdf.

Bluetooth SIG, Bluetooth Technology Website. Available at: https://www.bluetooth.com/ [Accessed October 10, 2016].

BOLZANI, C.A.M., 2004. *Residências Inteligentes*, Editora Livraria da Fisica. Available at: https://books.google.com/books?id=tgTlPE10u68C&pgis=1 [Accessed February 20, 2016].

Bromiley, P. & Thacker, N., 1999. SIMBAD. Available at: http://www.tina-vision.net/projects/simbad.php [Accessed June 6, 2016].

CakePHP, CakePHP v3.2 - the rapid development php framework. Available at: http://cakephp.org/ [Accessed February 9, 2016].

Chaouchi, H. & Luarent-Maknavicius, M., 2009. *Wireless and Mobile Network Security* 1ª. H. Chaouchi & M. Luarent-Maknavicius, eds.,

Charland, B.A. & Leroux, B., Mobile Application Development : Web vs . Native. , pp.0–4.

Cisco, Tráfego global de dados móveis crescerá quase 10 vezes entre 2014 e 2019. Available at: http://www.cisco.com/web/PT/press/articles/2015/20150203.html [Accessed February 12, 2016].

Corporation, O., of Stationary Human Presence • OMRON ' s unique MEMS and ASIC technology achieve a high. , pp.1–4.

Costa, R. et al., 2009. VirtualECare: Intelligent assisted living. *Lecture Notes of the Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering*, 1 LNICST, pp.138–144.

D-robotics UK, 2010. Temperature Sensor DHT 11 Humidity & Temperature Sensor. *DHT11 Datasheet*,

p.9.
Embley, D.W., Olivé, A. & Ram, S., 2006. *Conceptual Modeling - ER 2006: 25th International Conference on Conceptual Modeling, Tucson, AZ, USA, November 6-9, 2006, Proceedings*, Springer Science & Business Media. Available at: https://books.google.com/books?id=XfdAu8ZdY0cC&pgis=1 [Accessed February 20, 2016].
Engineers, E. et al., Helping Define IEEE 802 . 11 and other Wireless LAN Standards. , pp.1–9.
FATE, New Fall Detector arrives this September 2015. Available at: http://fate.upc.edu/index.php [Accessed August 19, 2016].
Ferguson, S., 2014. "Ridesourcing" Offers Promise, Peril for Cities. Available at: http://www.ubmfuturecities.com/author.asp?section_id=459&doc_id=526764) [Accessed February 9, 2016].
Food, Administration, D. & Others, 2002. General principles of software validation; final guidance for industry and FDA staff. *Rockville (MD): FDA*.
Garnock-Jones, T. & M. Roy, G., Introduction to Pika — pika 0.10.0 documentation. Available at: https://pika.readthedocs.io/en/0.10.0/index.html [Accessed September 11, 2016].
Gasparrini, S. et al., 2014. A Depth-Based Fall Detection System Using a Kinect® Sensor. *Sensors*, 14(2), pp.2756–2775. Available at: http://www.mdpi.com/1424-8220/14/2/2756/ [Accessed September 19, 2016].
Gast, M., 2005. *802.11 Wireless Networks: The Definitive Guide*, Available at: http://books.google.com/books?id=9rHnRzzMHLIC&pgis=1.
Google, Android. Available at: https://www.android.com/intl/pt_pt/.
Google, AngularJS — Superheroic JavaScript MVW Framework. Available at: https://angularjs.org/ [Accessed October 10, 2016b].
Google, Google Play. Available at: https://play.google.com/store?hl=pt_PT [Accessed October 10, 2016c].
Grails, The Grails Framework. Available at: https://grails.org/ [Accessed February 9, 2016].
Habib, M.A. shfak et al., 2014. Smartphone-based solutions for fall detection and prevention: challenges and open issues. *Sensors (Basel, Switzerland)*, 14(4), pp.7181–7208.
Harry Wang, 2013. Smart Home Platforms for Health. *Journal of Chemical Information and Modeling*, 53(9), pp.1689–1699.
Helal, A., Cook, D.J. & Schmalz, M., 2009. Smart home-based health platform for behavioral monitoring and alteration of diabetes patients. *Journal of diabetes science and technology (Online)*, 3(1), pp.141–148.
IEEE, IEEE SA - 802.15.4-2011 - IEEE Standard for Local and metropolitan area networks--Part 15.4: Low-Rate Wireless Personal Area Networks (LR-WPANs). Available at: https://standards.ieee.org/findstds/standard/802.15.4-2011.html [Accessed October 10, 2016].
Indoware, 2013. Ultrasonic Ranging Module HC - SR04. *Datasheet*, pp.1–4. Available at: http://www.micropik.com/PDF/HCSR04.pdf.
Intel, Intel® Context Sensing SDK | Intel® Developer Zone. Available at: https://software.intel.com/en-us/context-sensing-sdk [Accessed February 12, 2016].
Ionic Framework, Ionic: Advanced HTML5 Hybrid Mobile App Framework. Available at: http://ionicframework.com/ [Accessed February 20, 2016].
JetBrains, IntelliJ IDEA the Java IDE. Available at: https://www.jetbrains.com/idea/ [Accessed October 10, 2016].
Kamel Boulos, M.N. et al., 2009. Connectivity for healthcare and well-being management: examples from six European projects. *International journal of environmental research and public health*, 6(7), pp.1947–71. Available at: http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=2738891&tool=pmcentrez&rendertype=abstract [Accessed January 6, 2016].
Kerns, G.J., 2010. *Introduction to Probability and Statistics Using R*, Available at: papers2://publication/uuid/7AC444D3-E151-41DE-8521-51948F21A8EE.
Koubâa, A., Alves, M. & Tovar, E., 2005. IEEE 802.15.4 for Wireless Sensor Networks: A Technical Overview. *Architecture*, (July).
Laravel, Laravel - The PHP Framework For Web Artisans. Available at: https://laravel.com/ [Accessed February 9, 2016].

Lee, H. & Kwon, J., 2013. The Smart Home Service System Architecture for Healthy and Safe Human Life. *Proceedings, The 3rd International Conference on Circuits, Control, Communication, Electricity, Electronics, Energy, System, Signal and Simulation*, 25, pp.13–16.

Melorose, J., Perroy, R. & Careas, S., 2015. *DESIGN, DEPLOYMENT AND PERFORMANCE OF 4G-LTE NETWORKS*,

Mendel, F. et al., 2006. Analysis of Step-Reduced SHA-256. , pp.126–143.

Microsoft, Windows Dev Center. Available at: https://developer.microsoft.com/en-us/windows [Accessed October 10, 2016].

Miot, H.A., 2011. Tamanho da amostra em estudos clínicos e experimentais. *Jornal Vascular Brasileiro*, 10(4), pp.275–278.

Von Mises, L., 1963. *Human Action: A Treatise on Economics.* 3rd ed., Available at: http://www.jstor.org/stable/2145453?origin=crossref.

Monitor, A., Detector, F. & Android, F.O.R., MOVER. , pp.2–3.

Moreno, J.M. et al., 2012. Deliverable D3 . 3 – Pilot report for the second round of tests.

Mozilla, Firefox OS. Available at: https://www.mozilla.org/en-US/firefox/os/ [Accessed October 10, 2016a].

Mozilla, HTML5 - Glossary | MDN. Available at: https://developer.mozilla.org/en-US/docs/Glossary/HTML5 [Accessed February 7, 2016b].

Neto, B.G.D.L. et al., 2014. Implementação de uma aplicação ubíqua no Context Toolkit : problemas e uma proposta de extensão.

Nielsen, 2015. We are what we eat: Healthy eating trends around the world. , (January), pp.1–27. Available at: http://www.nielsen.com/us/en/insights/reports/2015/we-are-what-we-eat.html.

OASIS, Home | AMQP. Available at: https://www.amqp.org/ [Accessed October 10, 2016].

Oracle Corporation, java.com. Available at: https://www.java.com/pt_BR/ [Accessed October 10, 2016a].

Oracle Corporation, MySQL. Available at: https://www.mysql.com/ [Accessed September 8, 2016b].

Oracle Corporation, Welcome to NetBeans. Available at: https://netbeans.org/ [Accessed October 10, 2016c].

Perera, C. et al., 2013. Context Aware Computing for The Internet of Things: A Survey. , X(X), pp.1–41. Available at: http://arxiv.org/abs/1305.0982.

Pérez, M., 2003. *Conceptual Physics*,

PhoneGap, PhoneGap. Available at: http://phonegap.com/ [Accessed February 20, 2016].

PHP, PHP: PHP Usage Stats. Available at: http://php.net/usage.php [Accessed February 8, 2016].

Pivotal Software, RabbitMQ - Messaging that just works. Available at: https://www.rabbitmq.com/ [Accessed August 16, 2016].

Python Software Foundation, About Python™ | Python.org. Available at: https://www.python.org/about/.

Raspberry Pi Foundation, GPIO: Raspberry Pi Models A and B - Raspberry Pi Documentation. Available at: https://www.raspberrypi.org/documentation/usage/gpio/.

Raspberry Pi Foundation, New product launch! Introducing Raspberry Pi Model B+ - Raspberry Pi. Available at: https://www.raspberrypi.org/blog/introducing-raspberry-pi-model-b-plus/.

Rational Software, 2004. Rational Unified Process Best Practices for Software. *Development*, pp.1–21. Available at: http://scholar.google.com/scholar?hl=en&btnG=Search&q=intitle:Rational+Unified+Process+Best+Practices+for+Software#4.

Richardson, L. & Ruby, S., 2007. *RESTful Web Services*,

Sense4Care, Introducing the first personal Fall Detector - Sense4care. Available at: http://www.sense4care.com/en [Accessed March 15, 2016].

Silva, H.M., 2011. HomeSense Dissertação para obtenção do Grau de Mestre em Engenharia de Redes de Comunicações.

Singh, P., Sharma, D. & Agrawal, S., 2011. A Modern Study of Bluetooth Wireless Technology. *International Journal of Computer Science, Engineering anf Information Technology (IJCSEIT)*, 1(3), pp.55–63.

Society, I.C., 2012a. *IEEE Standard for Ethernet - Section Two*,

Society, I.C., 2012b. *IEEE Standard for Information technology--Telecommunications and information*

exchange between systems Local and metropolitan area networks--Specific requirements Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications, Available at: http://ieeexplore.ieee.org/servlet/opac?punumber=6178209.

Sousa, J. et al., aal @ home : a New Home Care Wireless Biosignal Monitoring Tool for Ambient Assisted Living. *Plux*.

Statista, 2014a. Adults receiving social care services in England 2012/2013, by age/reason. Available at: http://www.statista.com/statistics/297235/adults-in-receipt-of-social-care-services-in-england/ [Accessed February 4, 2016].

Statista, 2015a. Apple App Store: number of available apps 2015 | Statistic. Available at: http://www.statista.com/statistics/263795/number-of-available-apps-in-the-apple-app-store/ [Accessed February 12, 2016].

Statista, 2014b. Home and day care: Total spend by local authorities in England 2013, by patient type. Available at: http://www.statista.com/statistics/297299/total-spend-by-local-authorities-on-home-and-day-care-in-england/ [Accessed February 4, 2016].

Statista, 2015b. IoT: number of connected devices worldwide 2012-2020 | Statistic. Available at: http://www.statista.com/statistics/471264/iot-number-of-connected-devices-worldwide/ [Accessed February 10, 2016].

Statista, 2015c. IoT endpoint spending by category worldwide 2014-2016 and 2020 | Statistic. Available at: http://www.statista.com/statistics/485252/iot-endpoint-spending-by-category-worldwide/ [Accessed February 11, 2016].

Statistia, 2016. Deaths from falls in Scotland 2014. Available at: https://www.statista.com/statistics/537376/death-as-a-result-of-a-fall-by-age-in-scotland/ [Accessed October 10, 2016].

Storey, N., 2004. *Electrical and Electronic Systems by Neil Storey* Pearson, ed., Available at: http://pt.scribd.com/doc/180002401/Book-Electrical-and-Electronic-Systems-by-Neil-Storey#scribd [Accessed February 12, 2016].

System, A.L., 2009. PIR motion sensor. *US Patent 7,579,595*. Available at: http://www.google.com/patents/US7579595.

The Eclipse Foundation, Eclipse. Available at: https://eclipse.org/ [Accessed October 10, 2016].

The jQuery Foundation, jQuery. Available at: https://jquery.com/ [Accessed October 10, 2016].

Vermesan, O. & Friess, P., 2014. *Internet of Things Applications - From Research and Innovation to Market Deployment*, Available at: https://books.google.com.br/books?id=kw2doAEACAAJ.

w3schools, CSS3 Introduction. Available at: http://www.w3schools.com/css/css3_intro.asp [Accessed February 7, 2016a].

w3schools, 2014. HTML5. Available at: https://www.w3.org/TR/html/ [Accessed February 7, 2016].

w3schools, JSON Introduction. Available at: http://www.w3schools.com/js/js_json_intro.asp [Accessed October 10, 2016b].

w3schools, XML Tutorial. Available at: http://www.w3schools.com/xml/ [Accessed October 10, 2016c].

Wienhage, P., Rocha, I. & Scarpin, J.E., 2012. Aplicação do Target Costing e Engenharia do Valor na Precificação de Curso de Pós-Graduação. *ABCustos Associação Brasileira de Custos*, 7(1), pp.85–109.

Yii PHP, Yii PHP Framework: Best for Web 2.0 Development. Available at: http://www.yiiframework.com/ [Accessed February 9, 2016].

Zeithaml, V.A., 1988. Consumer perceptions of price, quality, and value: a means-end model and synthesis of evidence. *The Journal of Marketing*, 52(3), pp.2–22.

9 Anexos

Capítulo que complementa a informação necessária para o desenvolvimento da solução.

9.1 Descrição da base de dados

Home

A tabela *Home* é a tabela que tem associada a si toda a informação relacionada com a casa.

Tabela 2 – Base de dados: *Home*

Campo	Tipo	Designação
Id	*BIGINT*	Chave primária que relaciona com outras tabelas.
Version	*BIGINT*	Controlo de versão dos dados.
Address	*VARCHAR*	Morada física da habitação.
Description	*LONGTEXT*	Descrição da habitação e complemento de informação.
Name	*VARCHAR*	Nome associado à habitação.

Local

A tabela *Local* é uma tabela que armazena todas as informações dos compartimentos. É possível configurar um compartimento e associá-lo à habitação correspondente. Uma habitação pode ter vários compartimentos e cada compartimento está associado apenas a uma habitação.

Tabela 3 – Base de dados: *Local*

Campo	Tipo	Designação
Id	*BIGINT*	Chave primária que relaciona com outras tabelas.
Version	*BIGINT*	Controlo de versão dos dados.
Description	*LONGTEXT*	Descrição do local e complemento de informação.
Home_id	*BIGINT*	Chave primária que correlaciona uma habitação a um local (uma habitação tem associada a si vários locais/compartimentos).
Local	*VARCHAR*	Compartimento da habitação (casa de banho).
Notes	*LONGTEXT*	Notas para adição de informação pertinente.

Sensor

Um sensor está associado a um compartimento, sendo que este pode ter vários sensores, mas um sensor apenas pode ter um compartimento associado.

Tabela 4 – Base de dados: *Sensor*

Campo	Tipo	Designação
Id	*BIGINT*	Chave primária que relaciona com outras tabelas.
Version	*BIGINT*	Controlo de versão dos dados.
Description	*LONGTEXT*	Descrição do sensor e complemento de informação.
Identify	*VARCHAR*	Identificador associado a um sensor.
Local_id	*BIGINT*	Chave primária que correlaciona um sensor a um local (um local/compartimento tem associada a si vários sensores).
Notes	*LONGTEXT*	Notas para adição de informação pertinente.

MeasuresSensor

Um sensor recolhe informação e essa informação é armazenada na tabela *MeasuresSensor*, onde todas as medidas são guardadas e associadas a um sensor. Um sensor capta várias medidas, mas uma medida apenas pertence a um sensor.

Tabela 5 – Base de dados: *MeasuresSensor*

Campo	Tipo	Designação
Id	*BIGINT*	Chave primária que relaciona com outras tabelas.
Version	*BIGINT*	Controlo de versão dos dados.
Description	*LONGTEXT*	Descrição da medida do sensor recolhida e complemento de informação.
Measure	*VARCHAR*	Medida recolhida pelo sensor
Notes	*LONGTEXT*	Notas para adição de informação pertinente.
Sensor_id	*BIGINT*	Chave primária que correlaciona um sensor a uma medida (um sensor tem associada a si várias medidas recolhidas).

User

Tabela onde é armazenada toda a informação do utilizador do sistema.

Tabela 6 – Base de dados: *User*

Campo	Tipo	Designação
Id	*BIGINT*	Chave primária que relaciona com outras tabelas.
Version	*BIGINT*	Controlo de versão dos dados.
birthdate	*DATETIME*	Data de nascimento do utilizador.
Email	*VARCHAR*	E-mail do utilizador do sistema.
Gender	*VARCHAR*	Género do utilizador (masculino ou feminino).
Mobile	*VARCHAR*	Contacto telefónico (telemóvel) do utilizador.
Name	*VARCHAR*	Nome do utilizador.
Password	*VARCHAR*	*Password* do utilizador (encriptada pelo sistema).
Passwordupdated	*DATETIME*	Data da atualização de *password* (permite ao sistema forçar uma atualização de uma nova *password*).
Phone	*VARCHAR*	Contacto telefónico (telefone) do utilizador.

User_group_id	*BIGINT*	Chave primária que correlaciona um utilizador a um grupo de utilizadores (um grupo de utilizadores tem associada a si vários utilizadores).
Username	*VARCHAR*	Nome de utilizador para efetuar *login* no site.

Sponsor

Tabela que herda todos os campos do utilizador, uma vez que um responsável é também um utilizador, com alguns campos extra.

Tabela 7 – Base de dados: *Sponsor*

Campo	Tipo	Designação
Id	*BIGINT*	Chave primária que relaciona com outras tabelas.
Version	*BIGINT*	Controlo de versão dos dados.
Description	*LONGTEXT*	Descrição do responsável e complemento de informação.
Notes	*LONGTEXT*	Notas para adição de informação pertinente.
User_id	*BIGINT*	Chave primária que correlaciona um responsável a um utilizador (um responsável é também ele um utilizador).

UserPermissions

Tabela que permite armazenar as permissões dadas em especial a um utilizador.

Tabela 8 – Base de dados: *UserPermissions*

Campo	Tipo	Designação
User_id	*BIGINT*	Chave primária que relaciona com outras tabelas.
Permissions_string	*VARCHAR*	Permissões associadas a uma regra em específico.

Role

Tabela que permite armazenar as regras ou funções.

Tabela 9 – Base de dados: *Role*

Campo	Tipo	Designação
Id	*BIGINT*	Chave primária que relaciona com outras tabelas.
Version	*BIGINT*	Controlo de versão dos dados.
Name	*VARCHAR*	Nome da regra.

RolePermissions

Tabela que permite armazenar as permissões dadas em especial a uma regra ou a uma função.

Tabela 10 – Base de dados: *RolePermissions*

Campo	Tipo	Designação
Role_Id	*BIGINT*	Chave primária que relaciona com outras tabelas.
Permissions_string	*VARCHAR*	Controlo de versão dos dados.

UserRole

Tabela que permite armazenar informação do utilizador referente ao papel que tem no sistema.

Tabela 11 – Base de dados: *UserRole*

Campo	Tipo	Designação
User_roles_id	*BIGINT*	Chave primária que relaciona com outras tabelas.
Role_id	*BIGINT*	Chave primária da tabela Role que correlaciona uma regra a um utilizador (um utilizador tem associada a si várias regras).

UserGroup

Tabela que permite armazenar informação do utilizador e de um grupo (grupo de administração, grupo de utilizadores, etc.).

Tabela 12 – Base de dados: *UserGroup*

Campo	Tipo	Designação
Id	*BIGINT*	Chave primária que relaciona com outras tabelas.
Version	*BIGINT*	Controlo de versão dos dados.
Description	*VARCHAR*	Descrição do grupo de utilizadores e complemento de informação.
Name	*VARCHAR*	Nome o grupo de utilizadores.

UserGroupPermissions

Tabela que armazena a informação das permissões que são dadas a um grupo de utilizadores.

Tabela 13 – Base de dados: *UserGroupPermissions*

Campo	Tipo	Designação
User_group_id	*BIGINT*	Chave primária que relaciona com outras tabelas.
Permissions_string	*VARCHAR*	Permissões associadas a um grupo em específico.

UserGroupRoles

Tabela que armazena a informação dos papéis que são atribuídos a um grupo de utilizadores.

Tabela 14 – Base de dados: *UserGroupRoles*

Campo	Tipo	Designação
User_group_id	*BIGINT*	Chave primária da tabela de grupo de utilizadores que relaciona com as regras.
Role_id	*BIGINT*	Chave primária da tabela Role que correlaciona uma regra a um grupo de utilizadores (um grupo de utilizadores tem associada a si várias regras).

EventType

Tabela que permite adicionar diferentes tipos de eventos que um sensor pode detetar (queda ou agachamento).

Tabela 15 – Base de dados: *EventType*

Campo	Tipo	Designação
Id	*BIGINT*	Chave primária que relaciona com outras tabelas.
Version	*BIGINT*	Controlo de versão dos dados.

Description	*LONGTEXT*	Descrição da medida do sensor recolhida e complemento de informação.
Name	*VARCHAR*	Nome associado ao tipo de evento.
Notes	*LONGTEXT*	Notas para adição de informação pertinente.

Event

Tabela que armazena toda a informação relacionada com os eventos recolhida ao longo das diferentes leituras efetuadas pelos sensores. Um evento tem apenas um tipo de evento.

Tabela 16 – Base de dados: *Event*

Campo	Tipo	Designação
Id	*BIGINT*	Chave primária que relaciona com outras tabelas.
Version	*BIGINT*	Controlo de versão dos dados.
Description	*LONGTEXT*	Descrição da medida do sensor recolhida e complemento de informação.
Date	*DATETIME*	Data da ocorrência do evento.
Event_type	*BIGINT*	Chave primária que correlaciona um tipo de evento a um evento (um tipo de evento tem associada a si

		vários eventos).
Name	*VARCHAR*	Nome associado ao evento.
Notes	*LONGTEXT*	Notas para adição de informação pertinente.

Log

Tabela que armazena toda a informação relacionada com transações e ações do utilizador para registo.

Tabela 17 – Base de dados: *Log*

Campo	Tipo	Designação
Id	*BIGINT*	Chave primária que relaciona com outras tabelas.
Version	*BIGINT*	Controlo de versão dos dados.
Date	*DATETIME*	Data da ocorrência do evento.
Description	*LONGTEXT*	Descrição da medida do sensor recolhida e complemento de informação.
Operation_code	*VARCHAR*	Código de operação da ação tomada.
User_id	*BIGINT*	ID do utilizador que executou determinada operação.

ConfigurationAuthentication

Tabela que armazena informação relacionada com a atualização da *password* num determinado intervalo.

Tabela 18 – Base de dados: *ConfigurationAuthentication*

Campo	Tipo	Designação
Id	*BIGINT*	Chave primária que relaciona com outras tabelas.
Version	*BIGINT*	Controlo de versão dos dados.
Changepassword	*BIT*	Ativar ou desativar a alteração da *password*.
Interval_change_password	*INT*	Número de dias que um utilizador tem de atualizar a sua *password*.

9.2 RestFul

Application Programming Interface (API) para acesso à informação via RestFul.

9.2.1 Autenticação

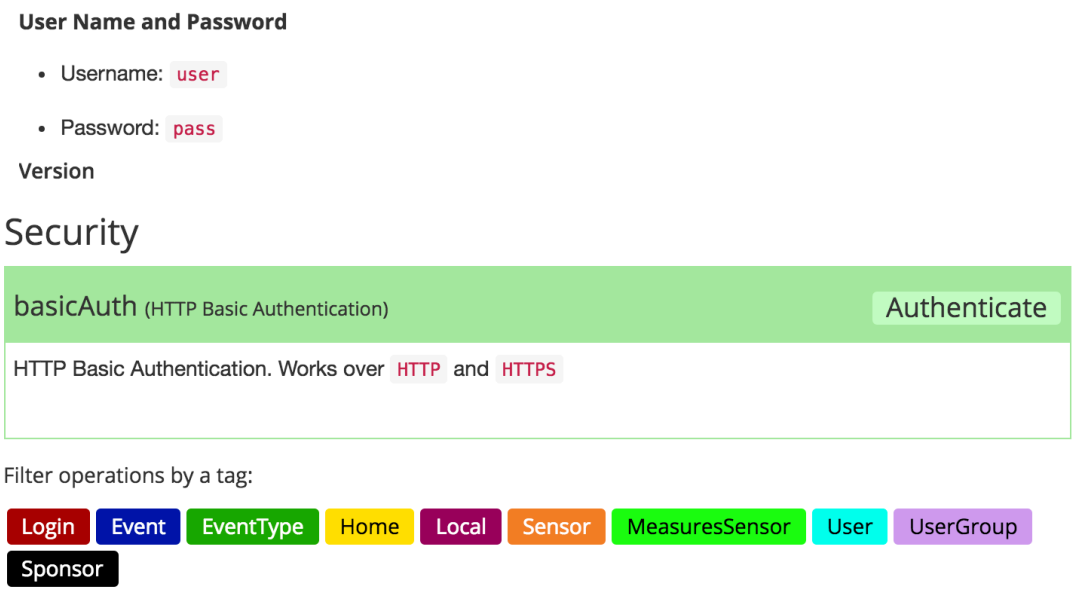


Figura 41 – RestFul API: autenticação básica

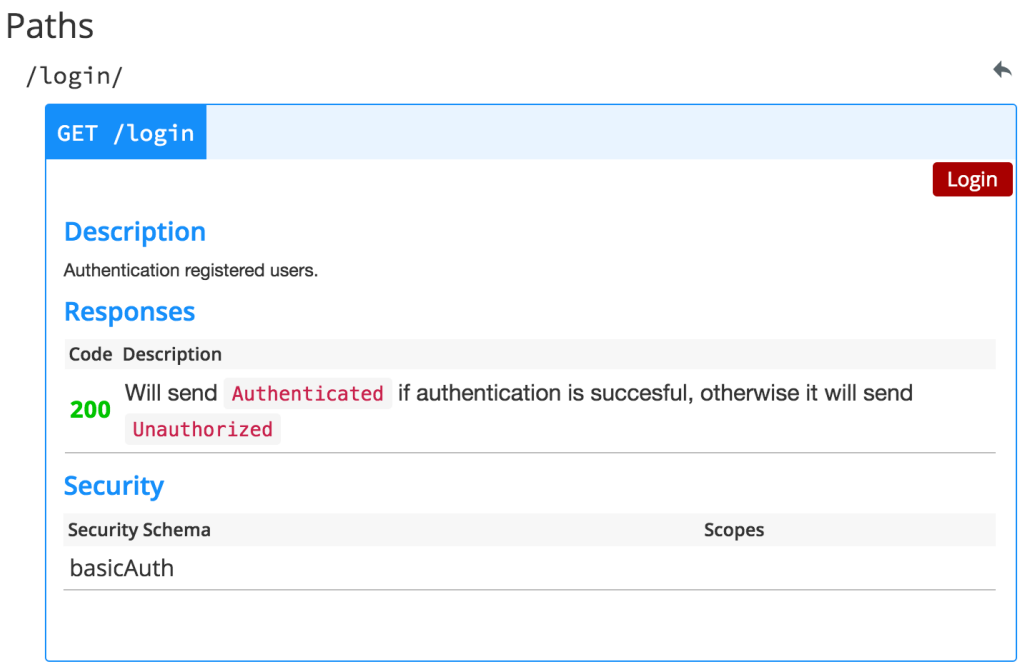


Figura 42 – RestFul API: *login*

9.2.2 Evento

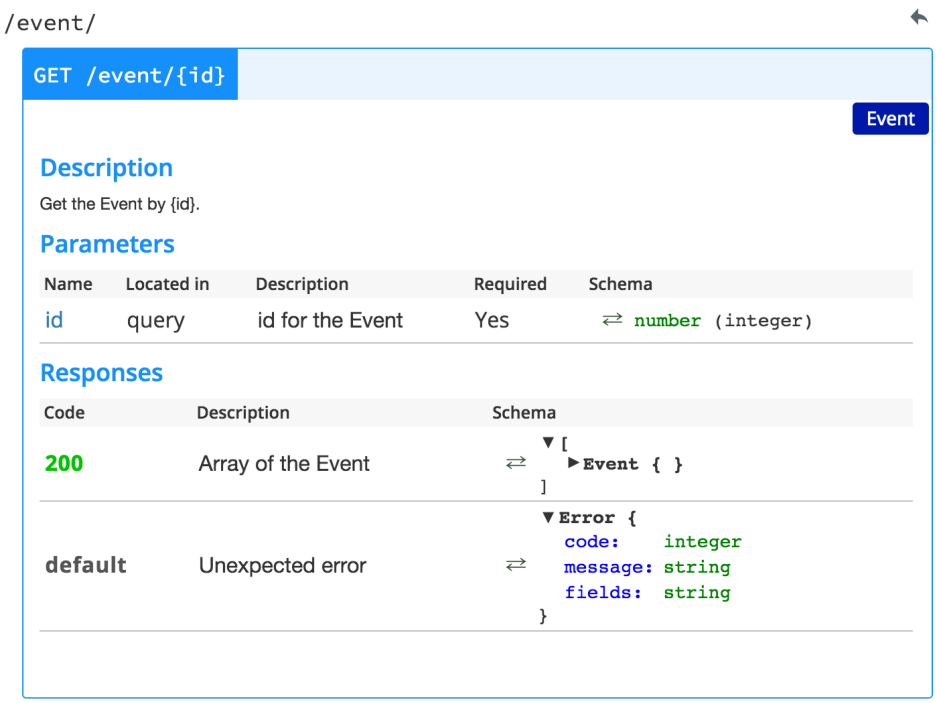


Figura 43 – RestFul API: leitura de um evento

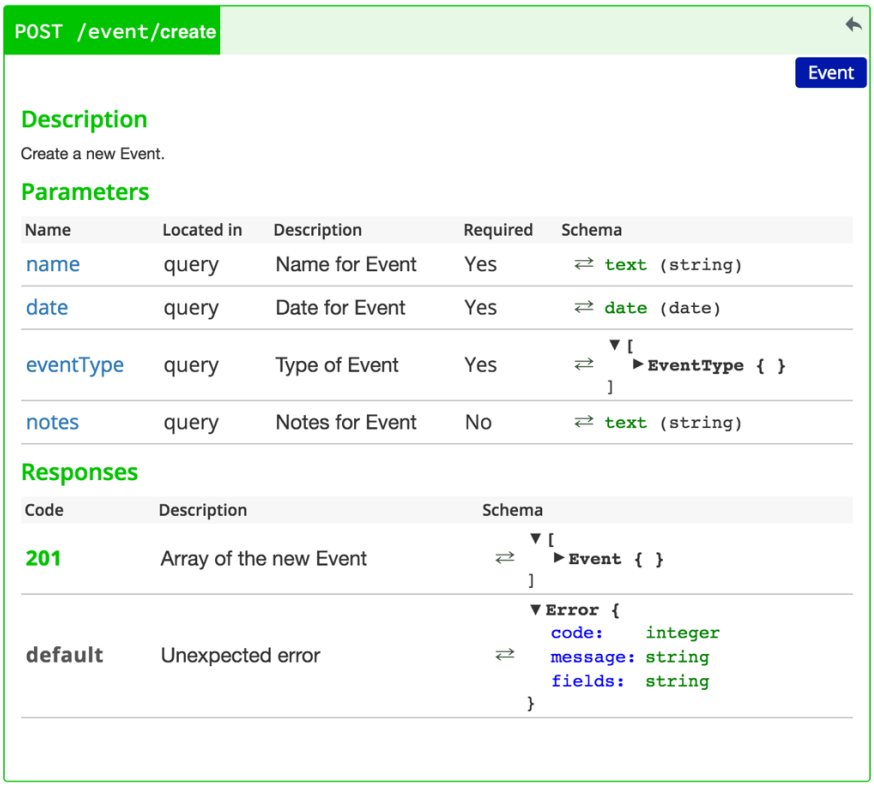
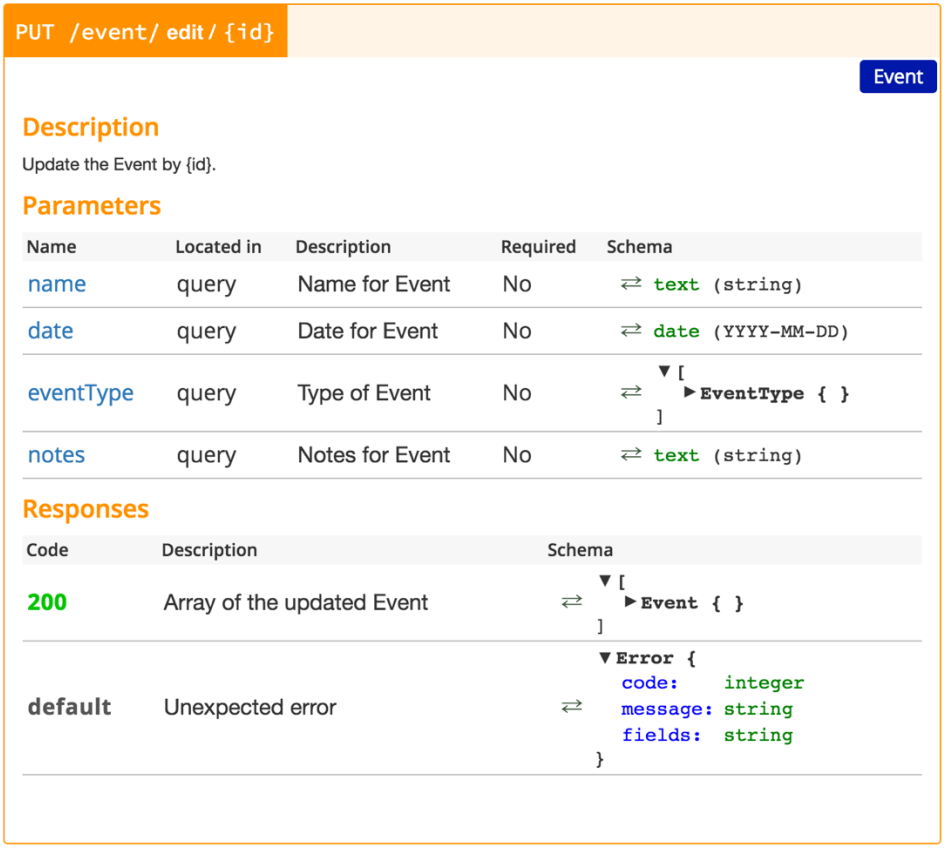


Figura 44 – RestFul API: criar um evento



PUT /event/edit/{id}

Event

Description

Update the Event by {id}.

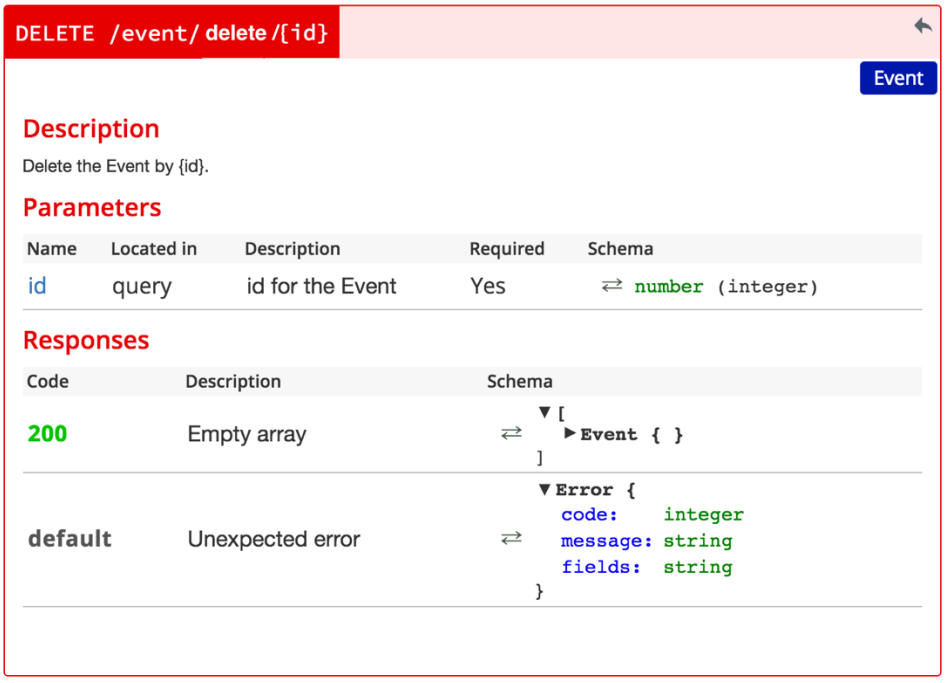
Parameters

Name	Located in	Description	Required	Schema
name	query	Name for Event	No	⇄ text (string)
date	query	Date for Event	No	⇄ date (YYYY-MM-DD)
eventType	query	Type of Event	No	⇄ ▼[▶EventType { }]
notes	query	Notes for Event	No	⇄ text (string)

Responses

Code	Description	Schema
200	Array of the updated Event	⇄ ▼[▶Event { }]
default	Unexpected error	⇄ ▼Error { code: integer message: string fields: string }

Figura 45 – RestFul API: editar um evento



DELETE /event/delete/{id}

Event

Description

Delete the Event by {id}.

Parameters

Name	Located in	Description	Required	Schema
id	query	id for the Event	Yes	⇄ number (integer)

Responses

Code	Description	Schema
200	Empty array	⇄ ▼[▶Event { }]
default	Unexpected error	⇄ ▼Error { code: integer message: string fields: string }

Figura 46 – RestFul API: apagar um evento

9.2.3 Tipos de Evento

/eventType/

GET /eventType/{id}

EventType

Description

Get the EventType by {id}.

Parameters

Name	Located in	Description	Required	Schema
id	query	id for the EventType	Yes	⇄ number (integer)

Responses

Code	Description	Schema
200	Array of the EventType	⇄ ▼ [▶ EventType { }]
default	Unexpected error	⇄ ▼ Error { code: integer, message: string, fields: string }

Figura 47 – RestFul API: leitura um tipo de evento

POST /eventType/create

EventType

Description

Create a new EventType.

Parameters

Name	Located in	Description	Required	Schema
name	query	Name for EventType	Yes	⇄ text (string)
description	query	Description for EventType	No	⇄ text (string)
notes	query	Notes for EventType	No	⇄ text (string)

Responses

Code	Description	Schema
201	Array of the new EventType	⇄ ▼ [▶ EventType { }]
default	Unexpected error	⇄ ▼ Error { code: integer, message: string, fields: string }

Figura 48 – RestFul API: criar um tipo de evento

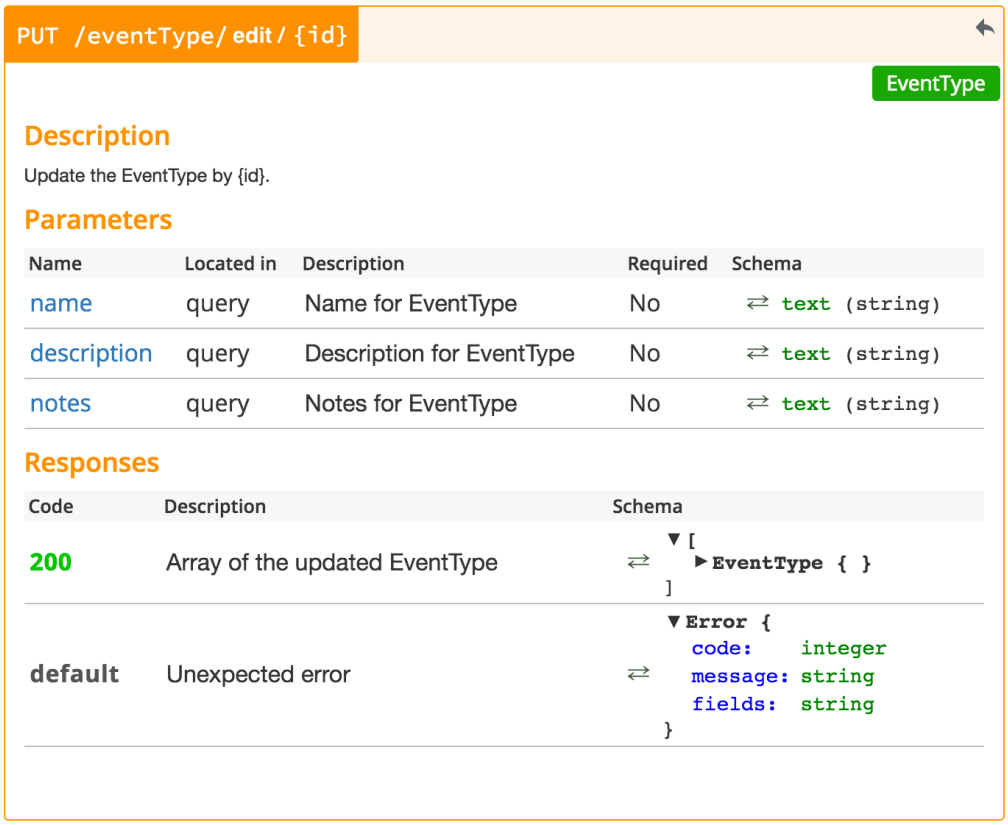


Figura 49 – RestFul API: editar um tipo de evento

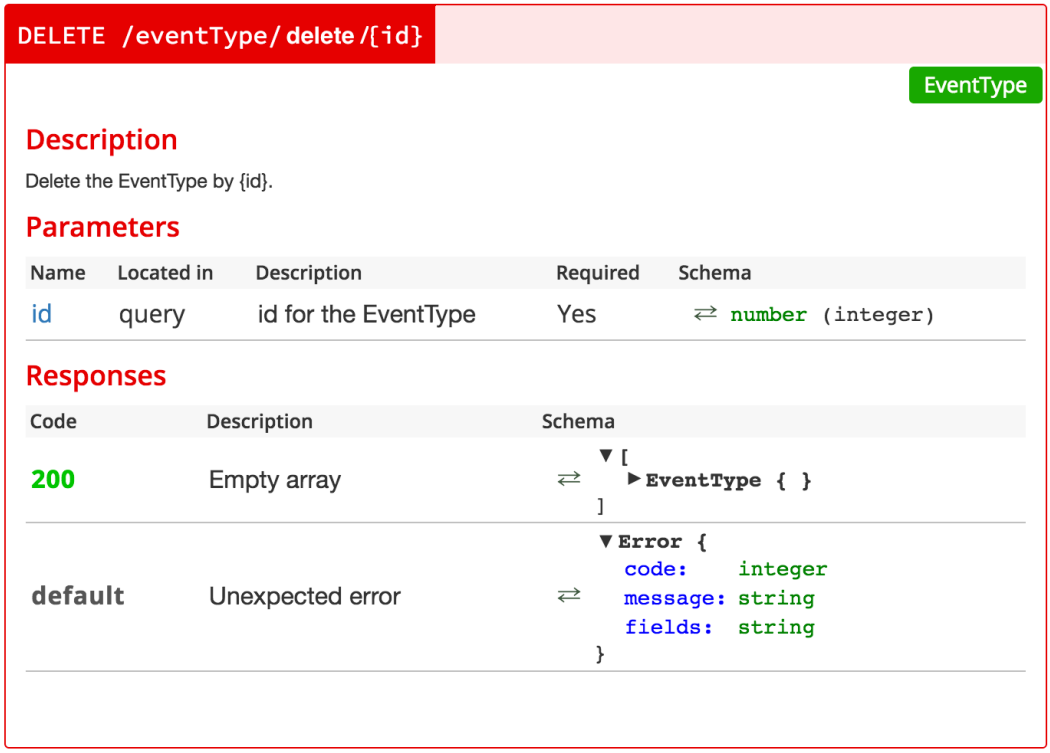


Figura 50 – RestFul API: apagar um tipo de evento

9.2.4 Habitação

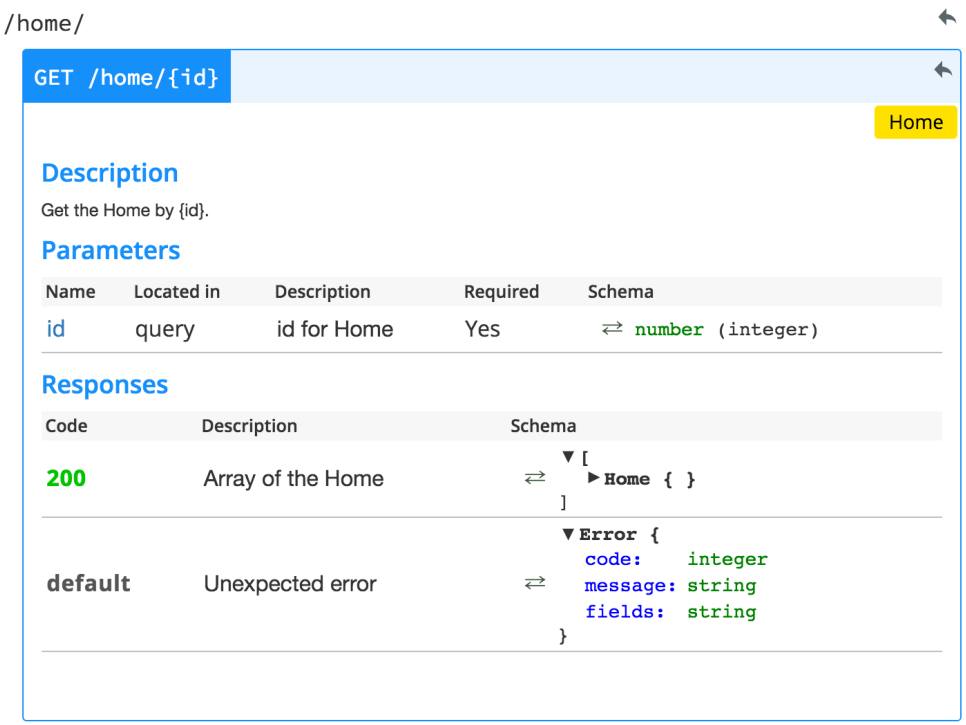


Figura 51 – RestFul API: leitura de uma habitação

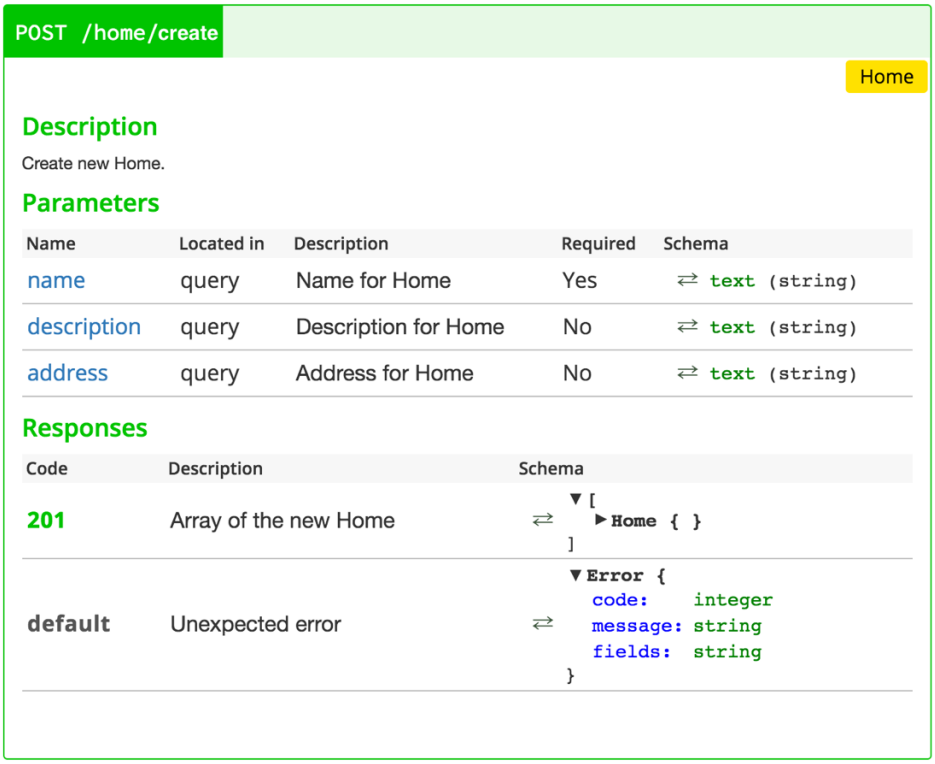


Figura 52 – RestFul API: criar uma habitação

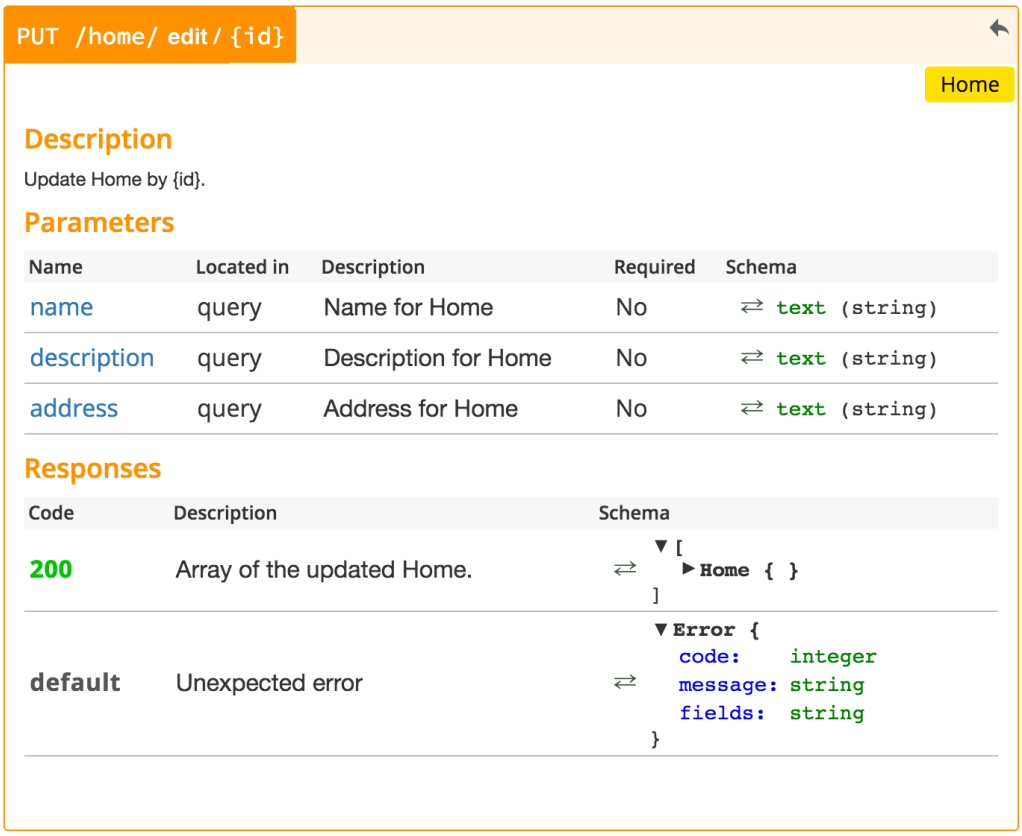


Figura 53 – RestFul API: editar uma habitação

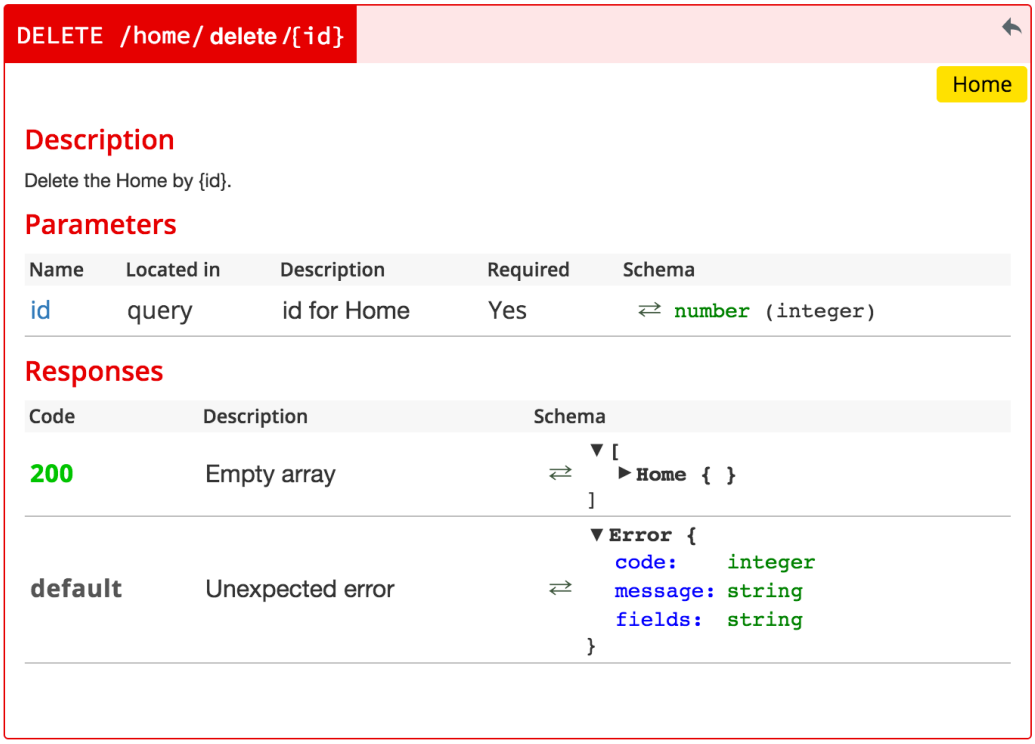


Figura 54 – RestFul API: apagar uma habitação

9.2.5 Compartimento

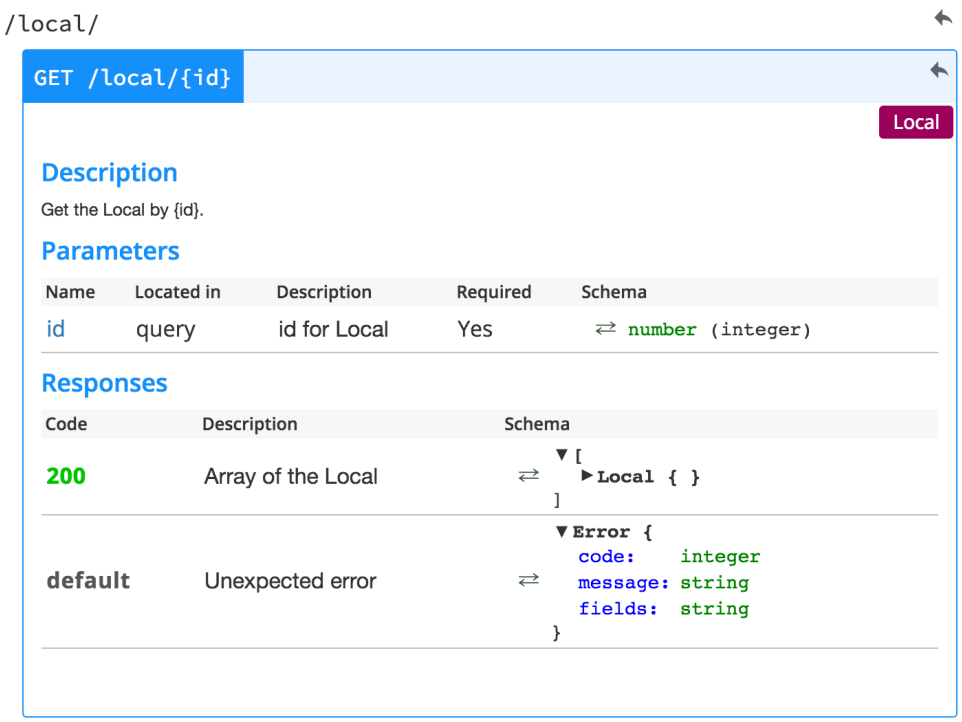


Figura 55 – RestFul API: leitura de um compartimento

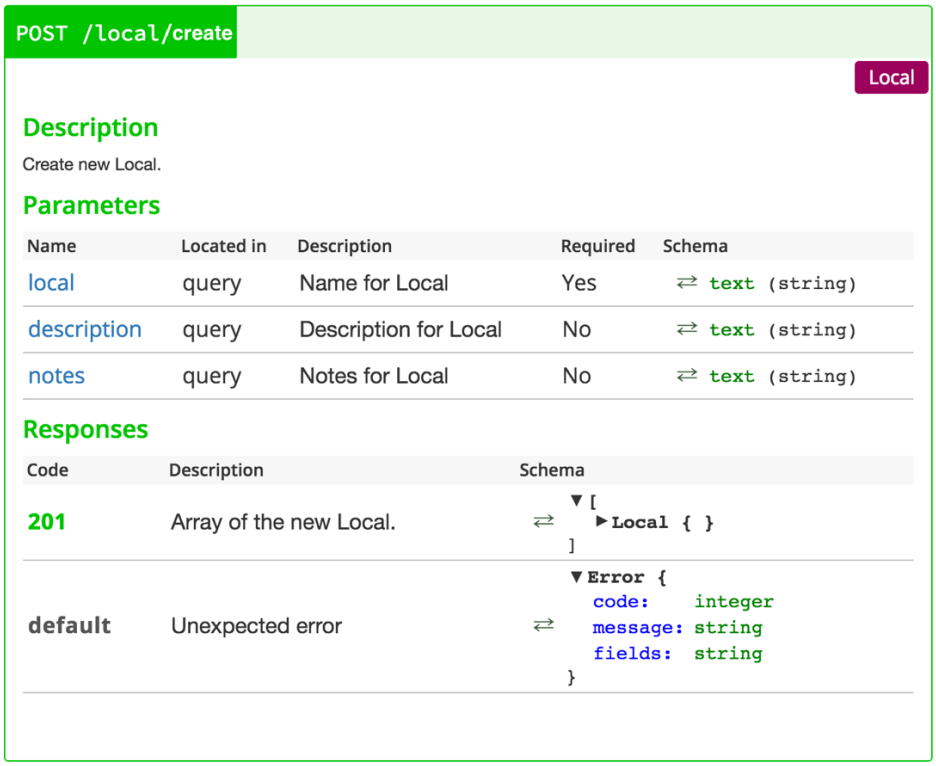
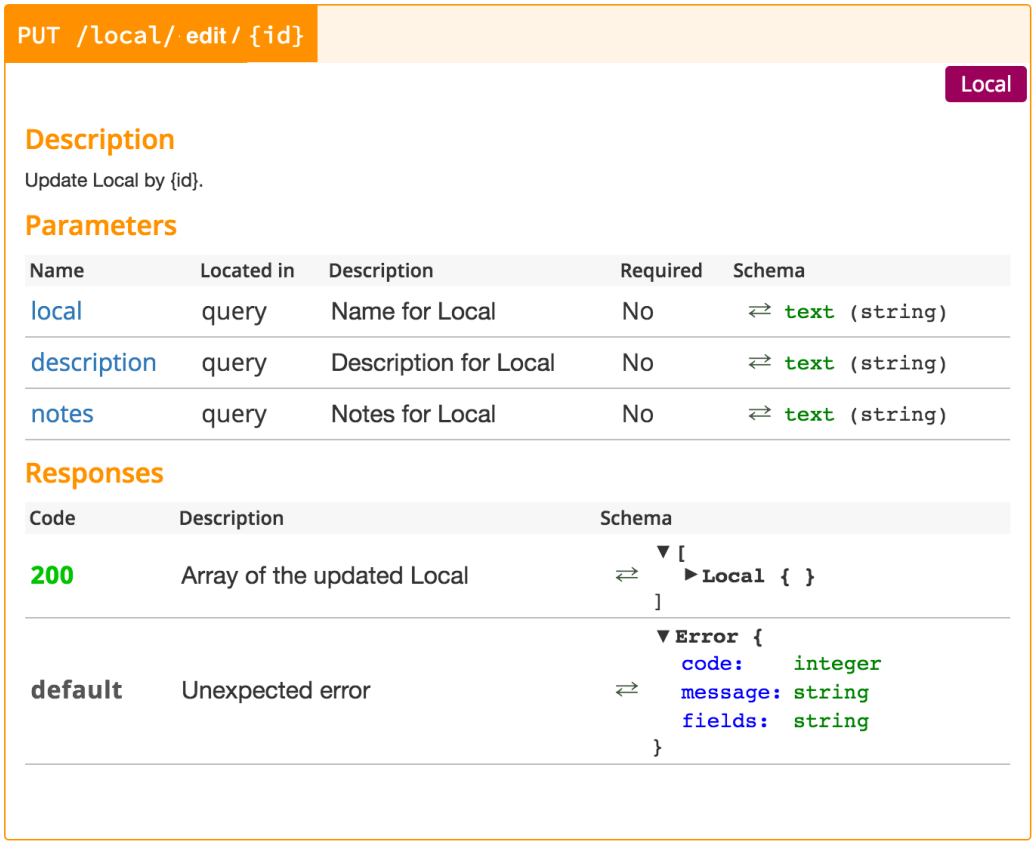


Figura 56 – RestFul API: criar um compartimento



PUT /local/ edit / {id}

Local

Description

Update Local by {id}.

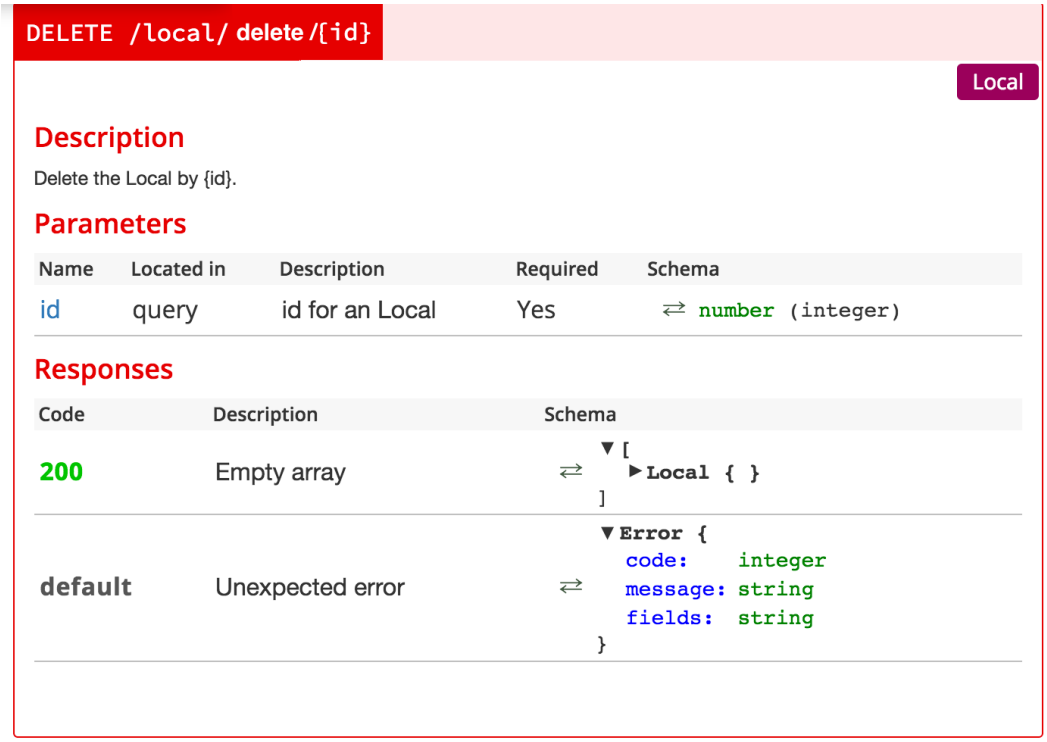
Parameters

Name	Located in	Description	Required	Schema
local	query	Name for Local	No	⇄ text (string)
description	query	Description for Local	No	⇄ text (string)
notes	query	Notes for Local	No	⇄ text (string)

Responses

Code	Description	Schema
200	Array of the updated Local	⇄ ▼ [▶ Local { }]
default	Unexpected error	⇄ ▼ Error { code: integer message: string fields: string }

Figura 57 – RestFul API: editar um compartimento



DELETE /local/ delete /{id}

Local

Description

Delete the Local by {id}.

Parameters

Name	Located in	Description	Required	Schema
id	query	id for an Local	Yes	⇄ number (integer)

Responses

Code	Description	Schema
200	Empty array	⇄ ▼ [▶ Local { }]
default	Unexpected error	⇄ ▼ Error { code: integer message: string fields: string }

Figura 58 – RestFul API: apagar um compartimento

9.2.6 Sensor

/sensor/

GET /sensor/{id}

Sensor

Description

Get the Sensor by {id}.

Parameters

Name	Located in	Description	Required	Schema
id	query	id for Sensor	Yes	⇄ number (integer)

Responses

Code	Description	Schema
200	Array of the Sensor	⇄ ▼ [▶ Sensor { }]
default	Unexpected error	⇄ ▼ Error { code: integer message: string fields: string }

Figura 59 – RestFul API: leitura de um sensor

POST /sensor/create

Sensor

Description

Create new Sensor.

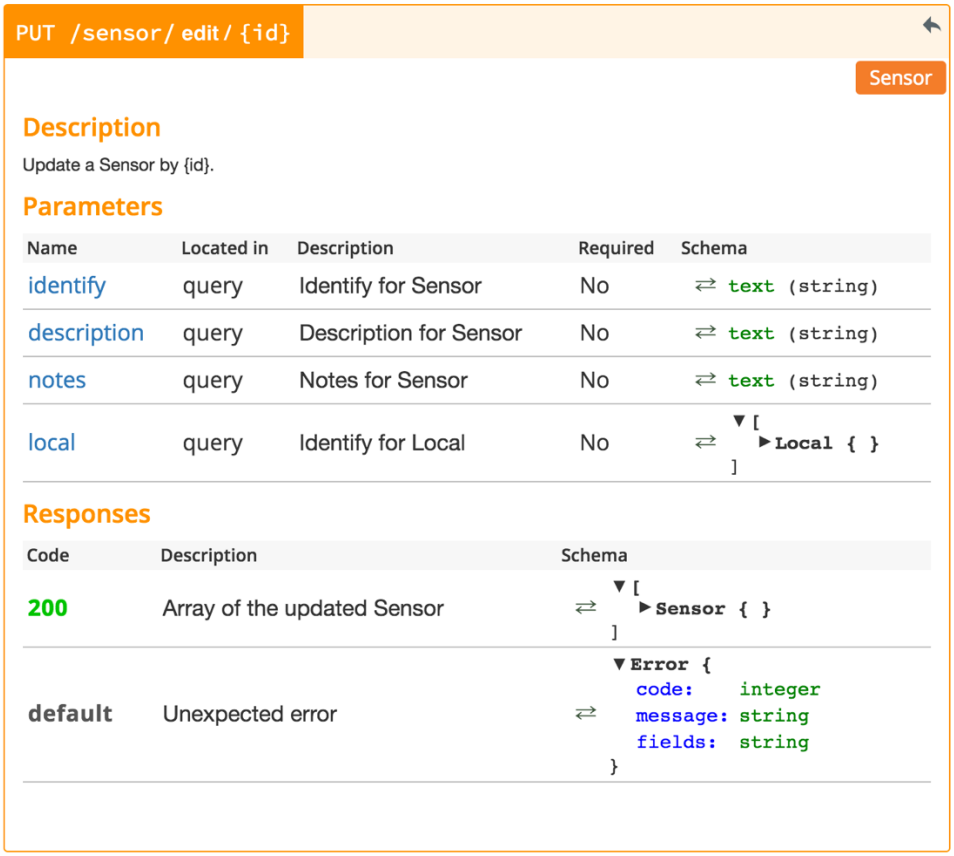
Parameters

Name	Located in	Description	Required	Schema
identify	query	Identify for Sensor	Yes	⇄ text (string)
description	query	Description for Sensor	No	⇄ text (string)
notes	query	Notes for Sensor	No	⇄ text (string)
local	query	Identify for Local	Yes	⇄ ▼ [▶ Local { }]

Responses

Code	Description	Schema
201	Array of the new Sensor	⇄ ▼ [▶ Sensor { }]
default	Unexpected error	⇄ ▼ Error { code: integer message: string fields: string }

Figura 60 – RestFul API: criar um sensor



PUT /sensor/edit/{id}

Sensor

Description

Update a Sensor by {id}.

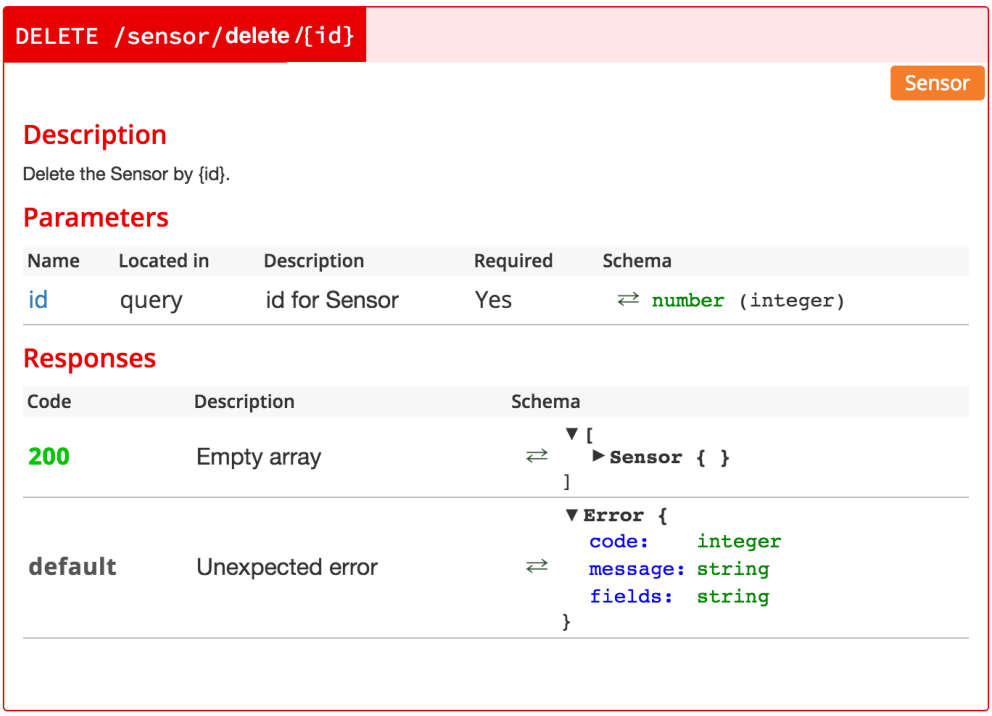
Parameters

Name	Located in	Description	Required	Schema
identify	query	Identify for Sensor	No	⇄ text (string)
description	query	Description for Sensor	No	⇄ text (string)
notes	query	Notes for Sensor	No	⇄ text (string)
local	query	Identify for Local	No	⇄ ▼ [▶ Local { }]

Responses

Code	Description	Schema
200	Array of the updated Sensor	⇄ ▼ [▶ Sensor { }]
default	Unexpected error	⇄ ▼ Error { code: integer message: string fields: string }

Figura 61 – RestFul API: editar um sensor



DELETE /sensor/delete/{id}

Sensor

Description

Delete the Sensor by {id}.

Parameters

Name	Located in	Description	Required	Schema
id	query	id for Sensor	Yes	⇄ number (integer)

Responses

Code	Description	Schema
200	Empty array	⇄ ▼ [▶ Sensor { }]
default	Unexpected error	⇄ ▼ Error { code: integer message: string fields: string }

Figura 62 – RestFul API: apagar um sensor

9.2.7 Medidas do Sensor

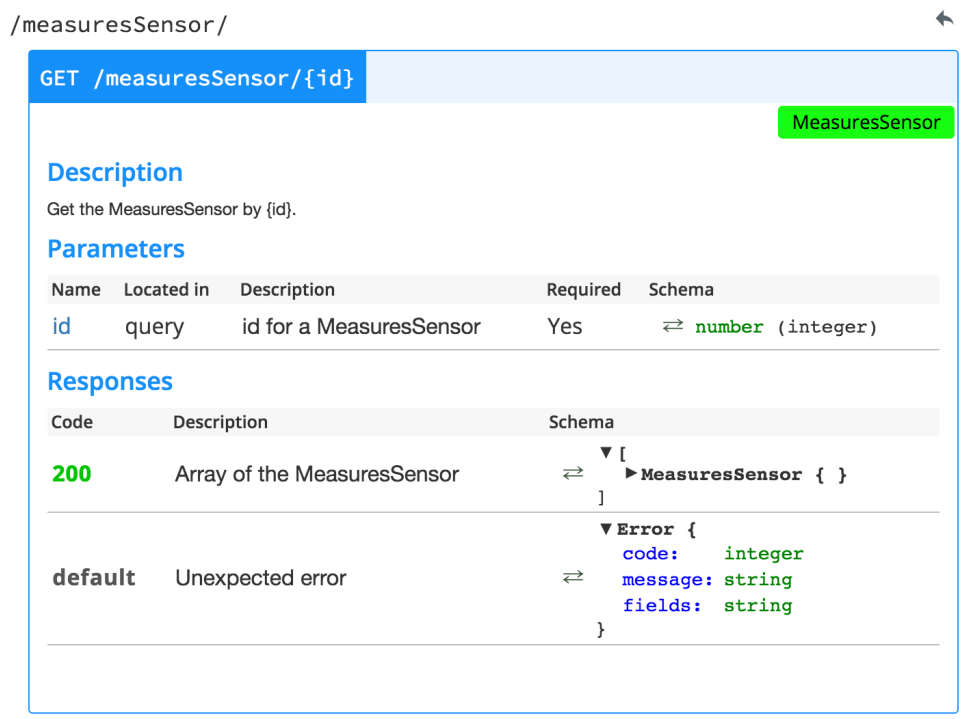


Figura 63 – RestFul API: leitura da medida do sensor

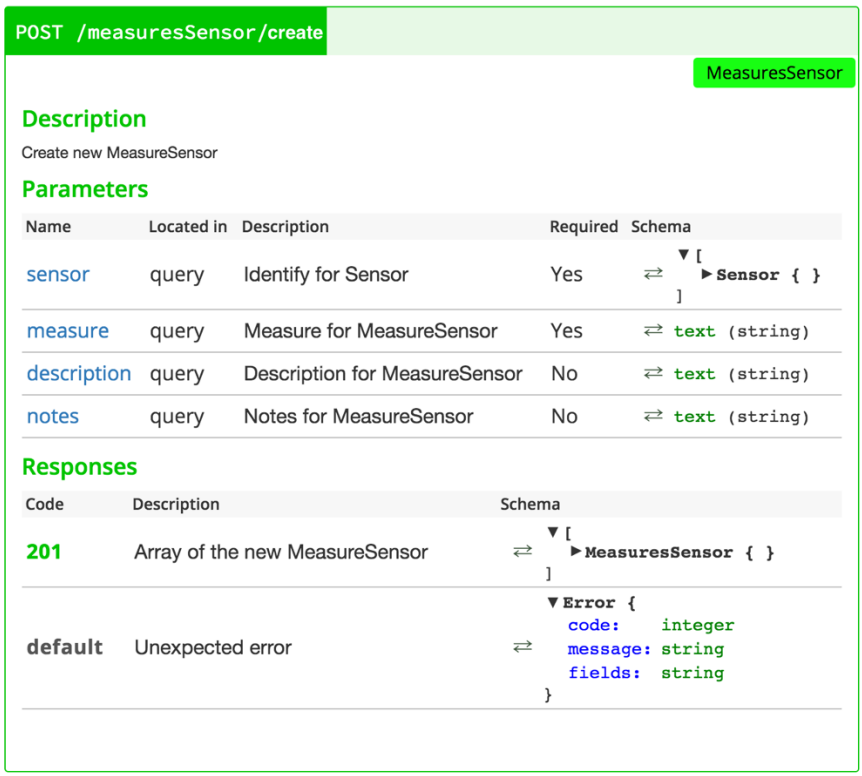


Figura 64 – RestFul API: criar medida do sensor

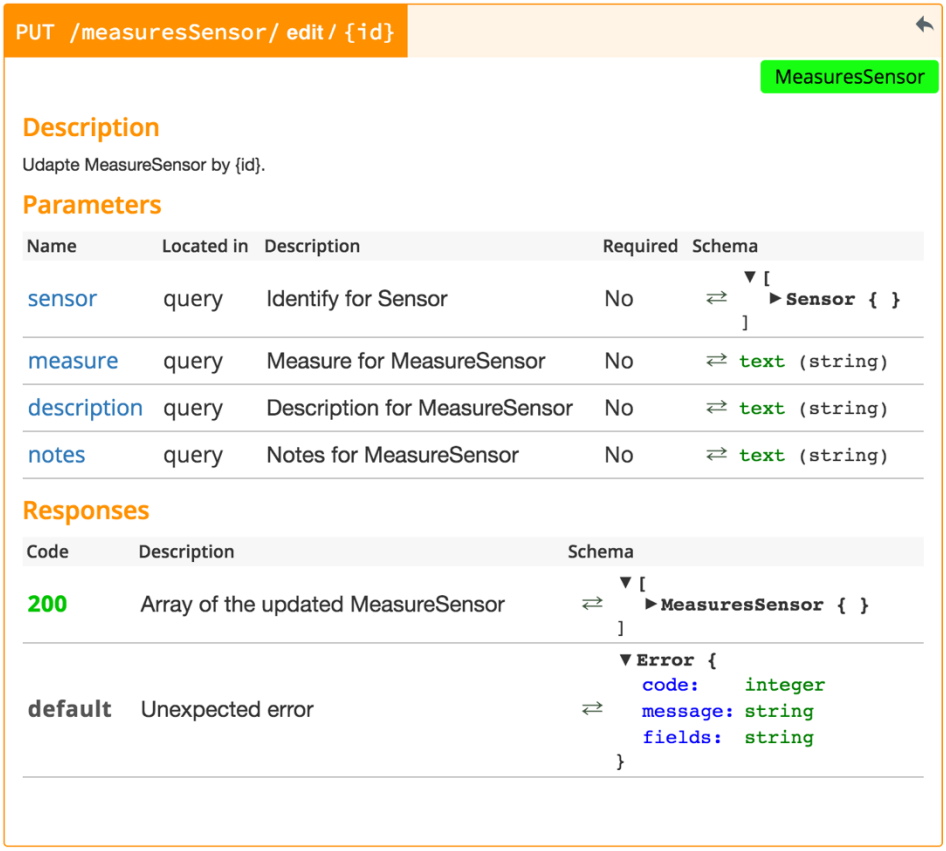


Figura 65 – RestFul API: editar medida do sensor

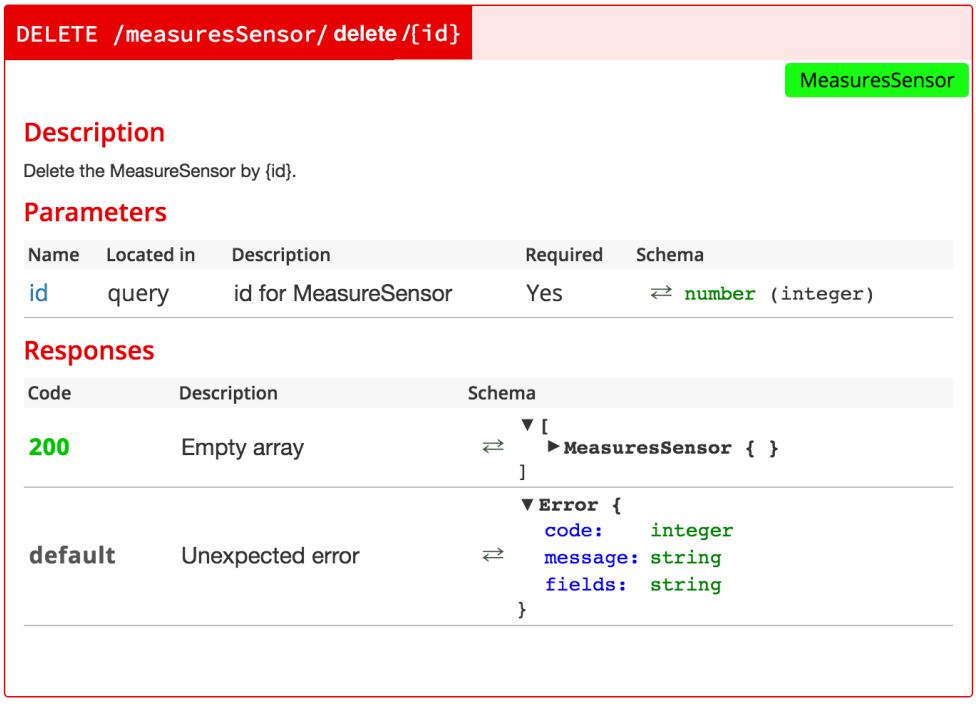


Figura 66 – RestFul API: apagar medida do sensor

9.2.8 Utilizador

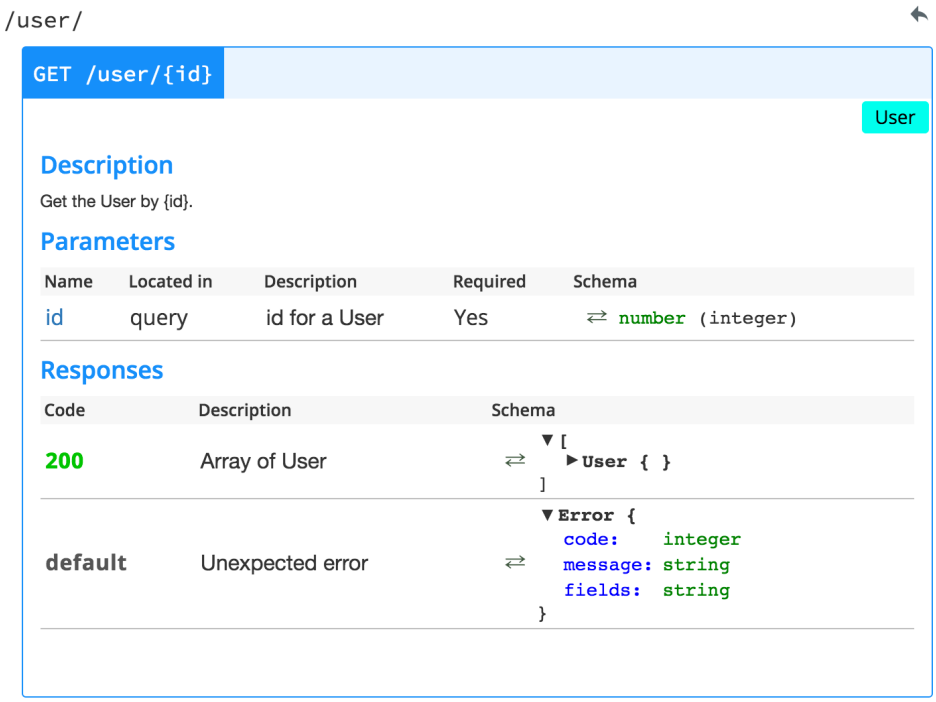
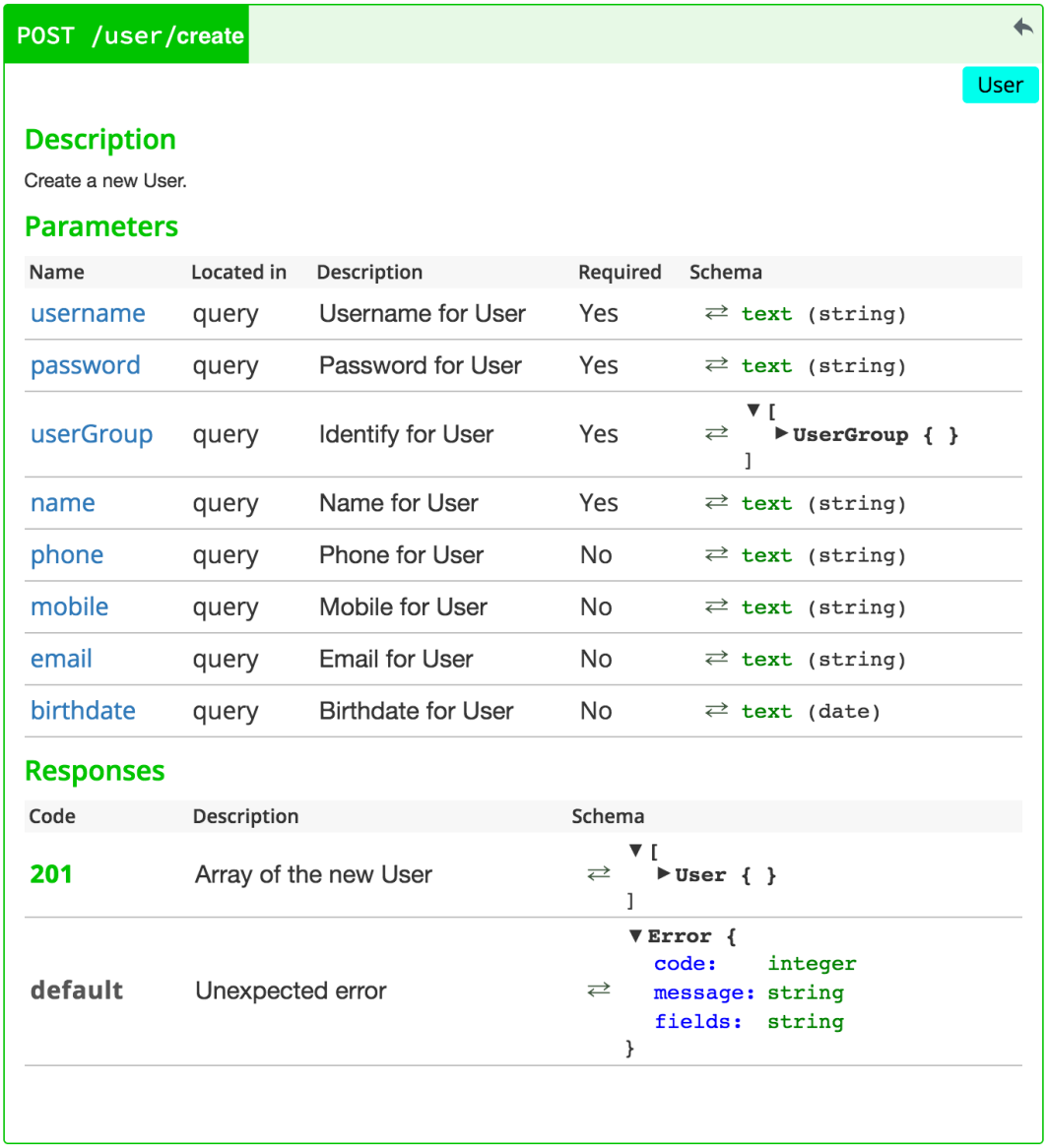


Figura 67 – RestFul API: leitura do utilizador



POST /user/create

User

Description

Create a new User.

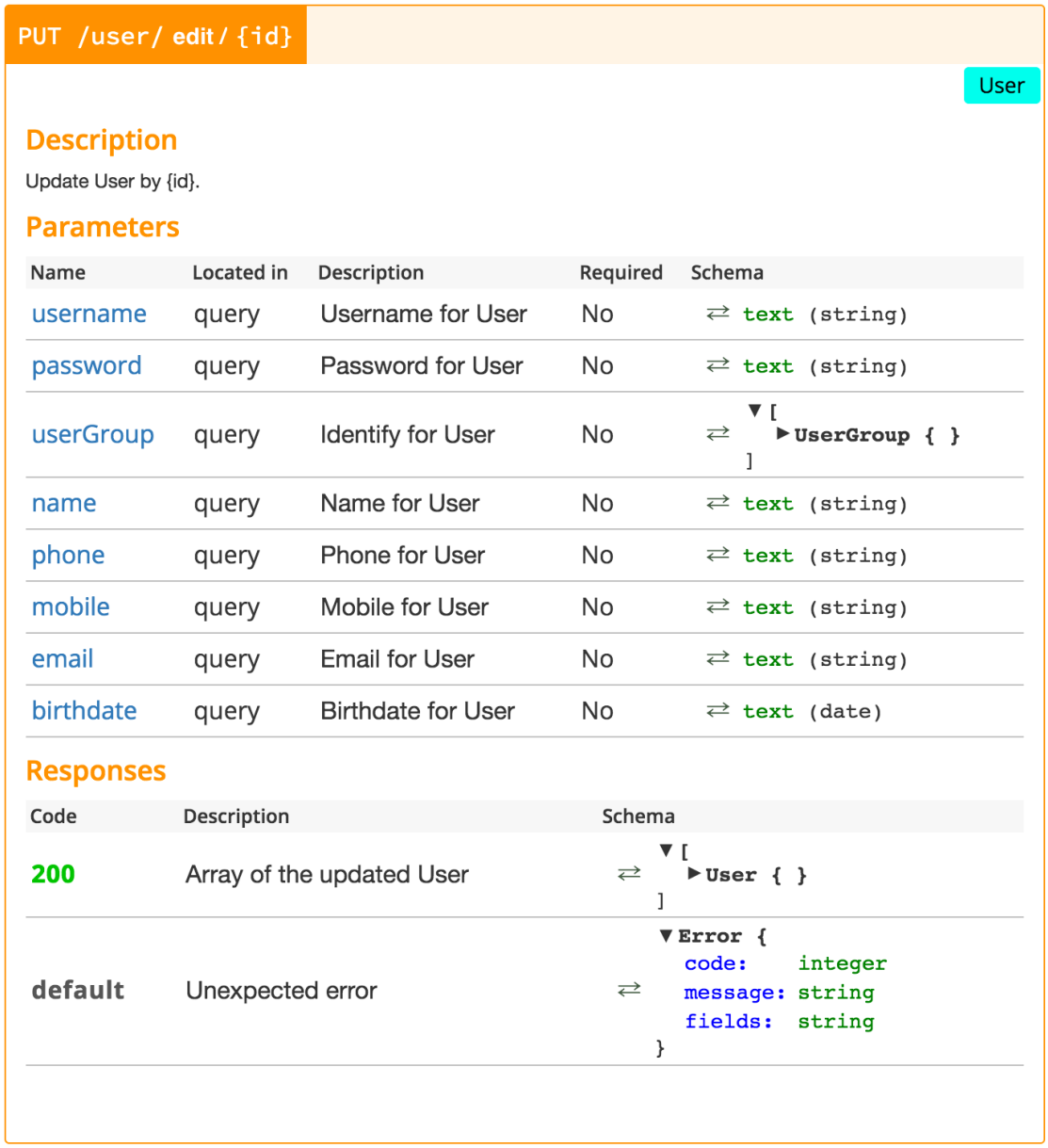
Parameters

Name	Located in	Description	Required	Schema
username	query	Username for User	Yes	⇄ text (string)
password	query	Password for User	Yes	⇄ text (string)
userGroup	query	Identify for User	Yes	⇄ ▼ [▶ UserGroup { }]
name	query	Name for User	Yes	⇄ text (string)
phone	query	Phone for User	No	⇄ text (string)
mobile	query	Mobile for User	No	⇄ text (string)
email	query	Email for User	No	⇄ text (string)
birthdate	query	Birthdate for User	No	⇄ text (date)

Responses

Code	Description	Schema
201	Array of the new User	⇄ ▼ [▶ User { }]
default	Unexpected error	⇄ ▼ Error { code: integer, message: string, fields: string }

Figura 68 – RestFul API: criar utilizador



PUT /user/ edit / {id}

User

Description

Update User by {id}.

Parameters

Name	Located in	Description	Required	Schema
username	query	Username for User	No	⇄ text (string)
password	query	Password for User	No	⇄ text (string)
userGroup	query	Identify for User	No	⇄ ▼ [▶ UserGroup { }]
name	query	Name for User	No	⇄ text (string)
phone	query	Phone for User	No	⇄ text (string)
mobile	query	Mobile for User	No	⇄ text (string)
email	query	Email for User	No	⇄ text (string)
birthdate	query	Birthdate for User	No	⇄ text (date)

Responses

Code	Description	Schema
200	Array of the updated User	⇄ ▼ [▶ User { }]
default	Unexpected error	⇄ ▼ Error { code: integer message: string fields: string }

Figura 69 – RestFul API: editar utilizador

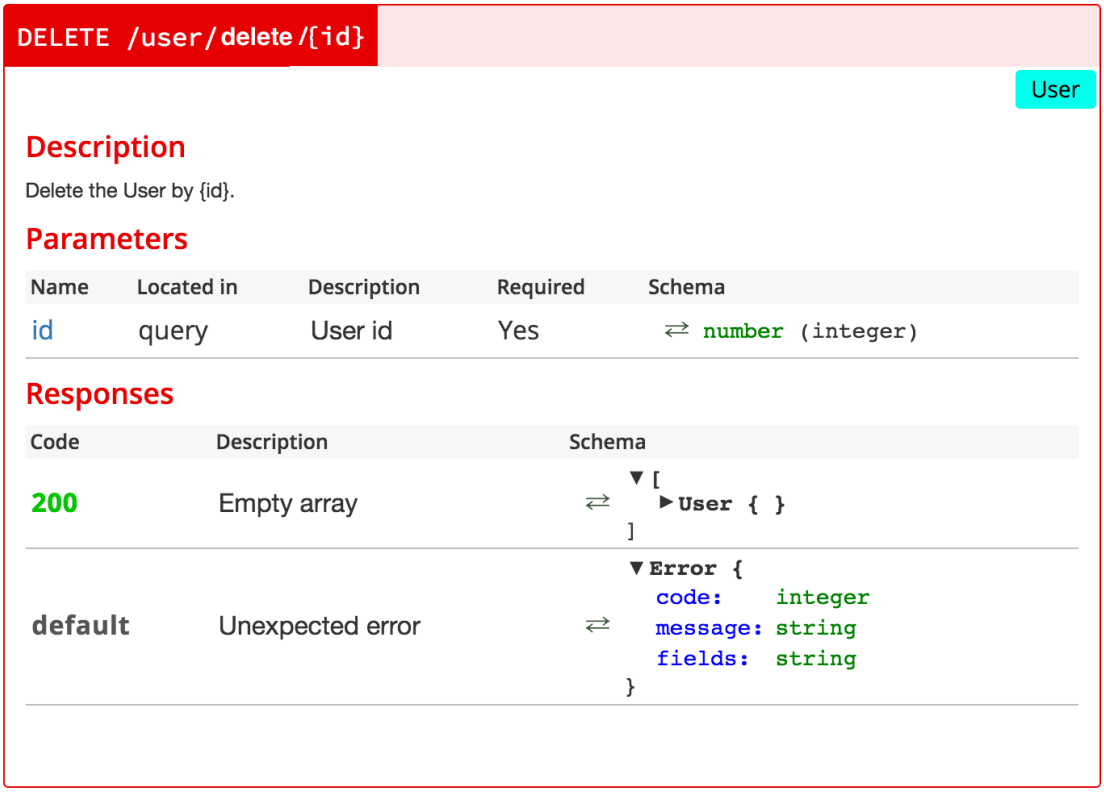


Figura 70 – RestFul API: apagar utilizador

9.2.9 Grupo de Utilizadores

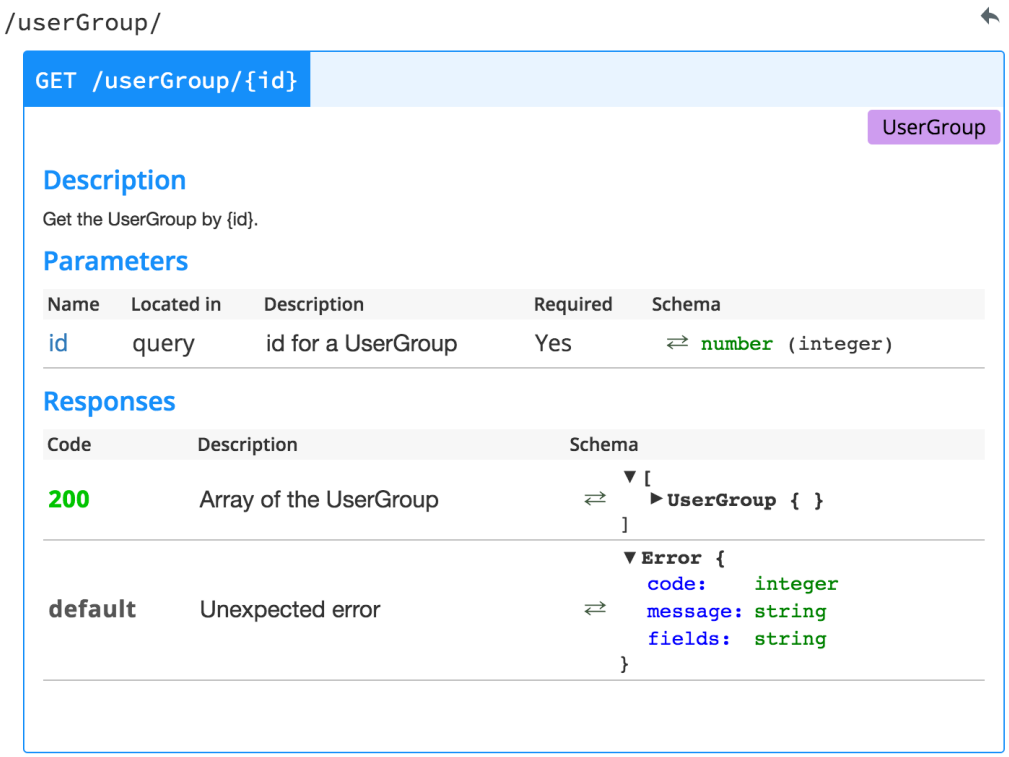


Figura 71 – RestFul API: leitura de grupo de utilizadores

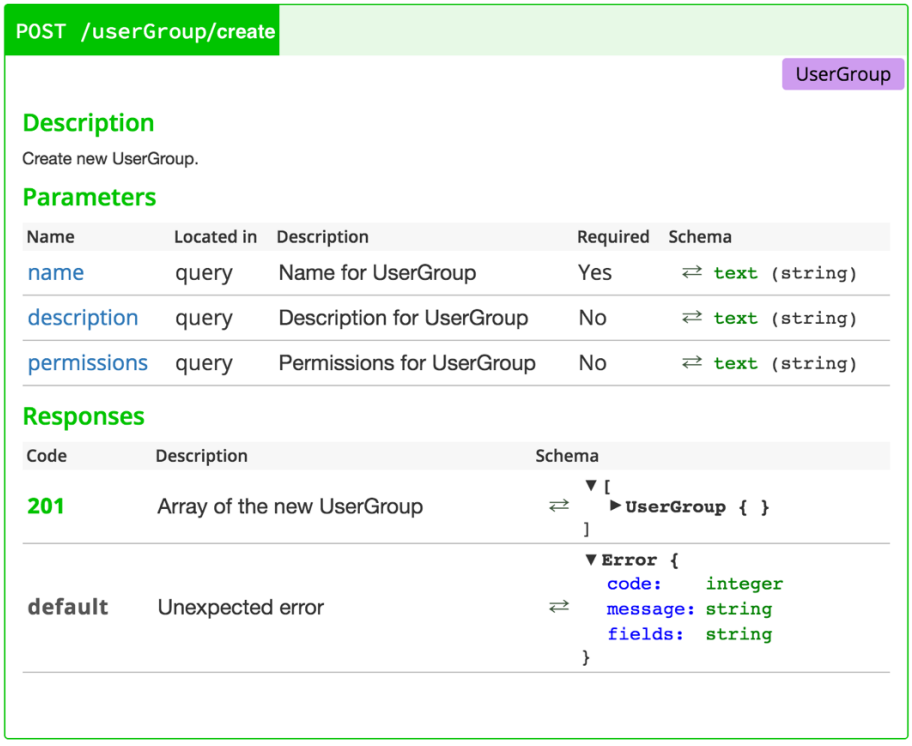


Figura 72 – RestFul API: criar grupo de utilizadores

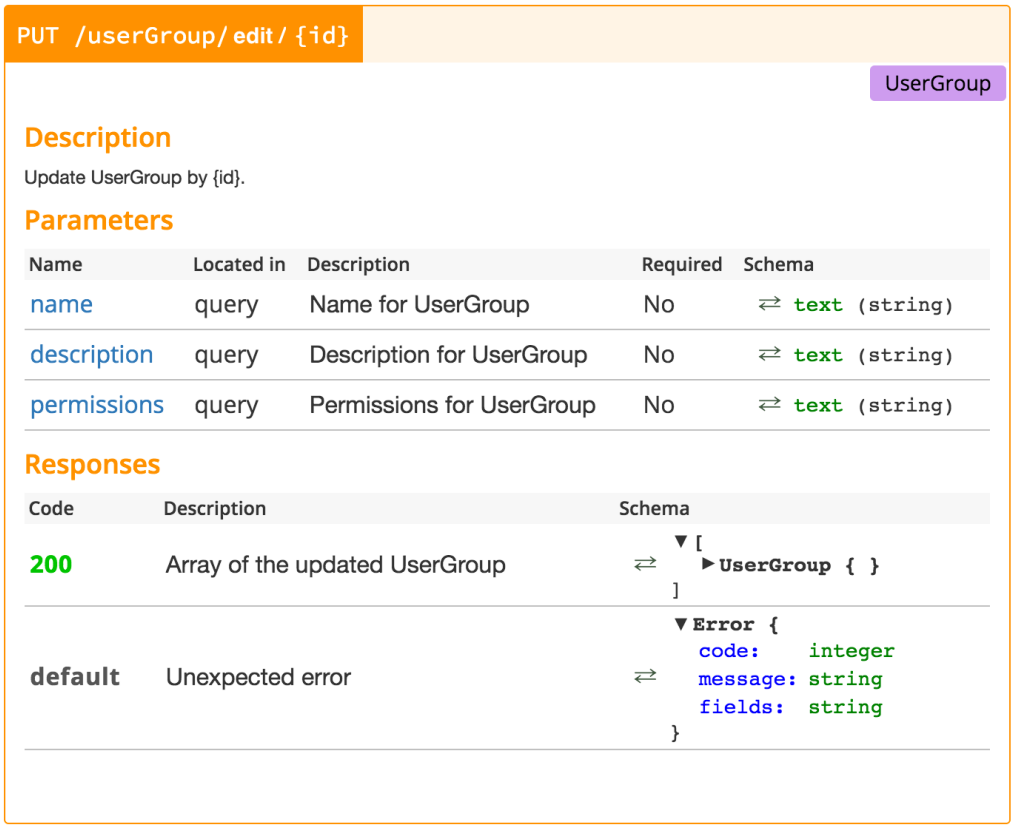


Figura 73 – RestFul API: editar grupo de utilizadores

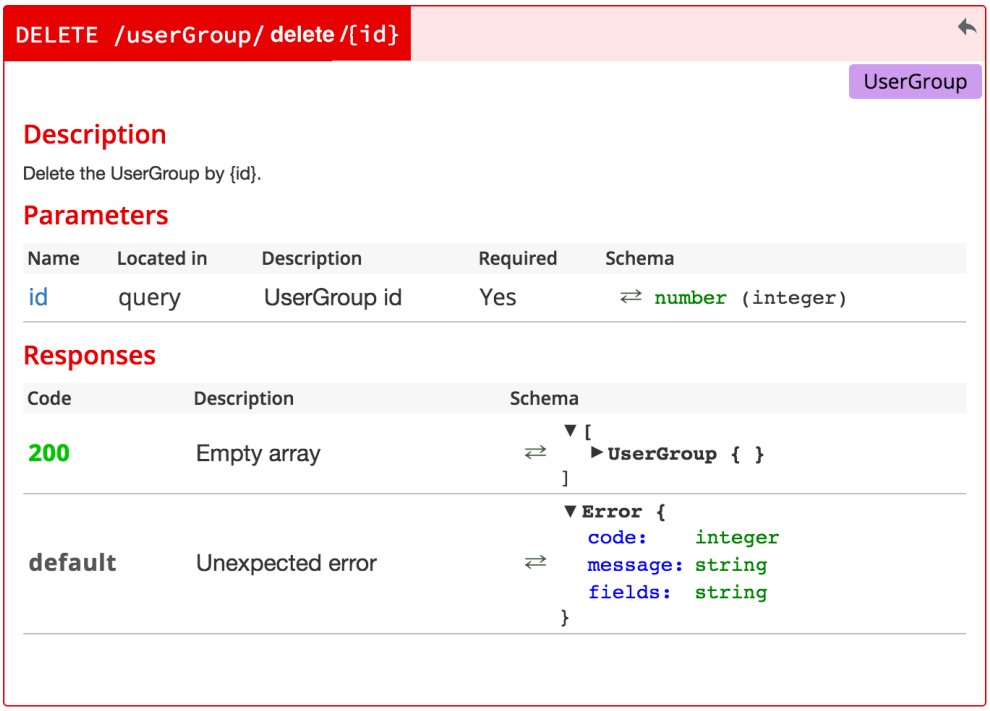


Figura 74 – RestFul API: apagar grupo de utilizadores

9.2.10 Responsável

/sponsor/{id}

GET /sponsor/{id}

Sponsor

Description

Get the Sponsor by {id}.

Parameters

Name	Located in	Description	Required	Schema
id	query	id for Sponsor	Yes	⇄ number (integer)

Responses

Code	Description	Schema
200	Array of the Sponsor	⇄ ▼ [▶ Sponsor { }]
default	Unexpected error	⇄ ▼ Error { code: integer message: string fields: string }

Figura 75 – RestFul API: leitura de responsável

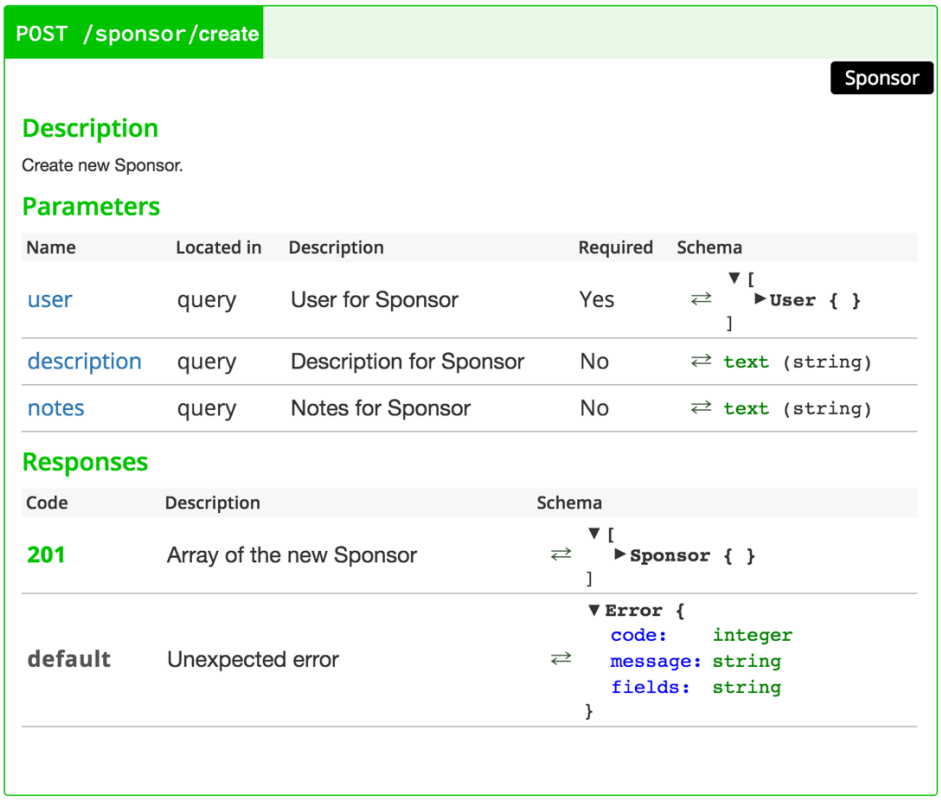


Figura 76 – RestFul API: criar responsável

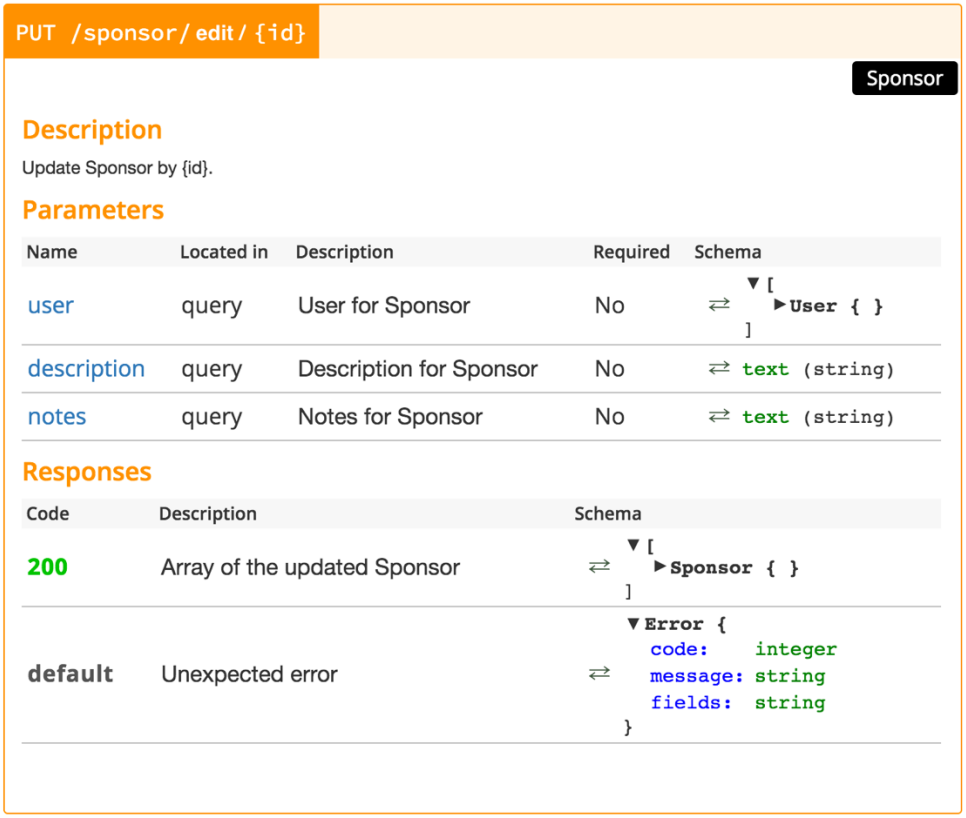


Figura 77 – RestFul API: editar responsável

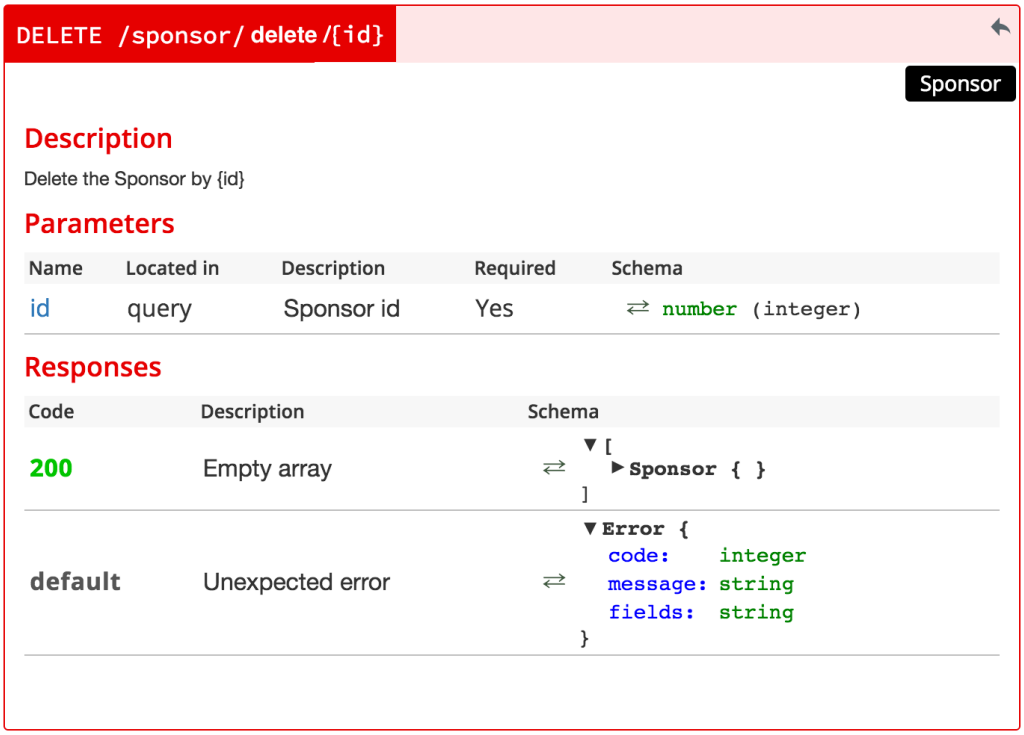


Figura 78 – RestFul API: apagar responsável

9.3 Aplicação *Web*

9.3.1 Homepage

Figura 79 – Aplicação *web*: Imagem Inicial

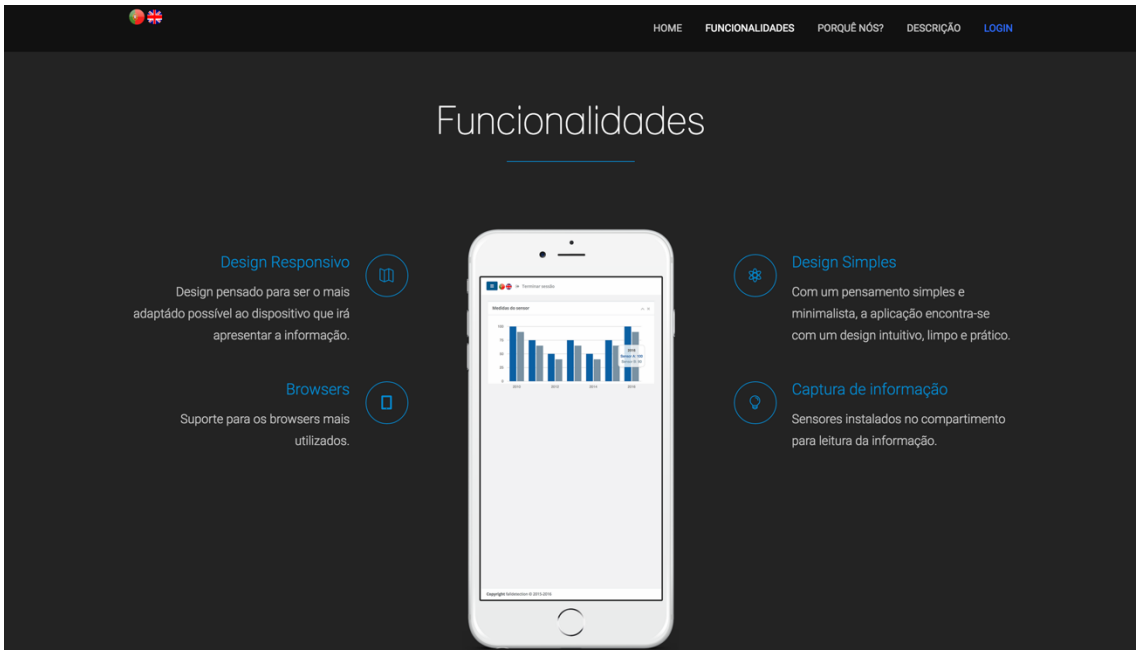


Figura 80 – Aplicação *web*: Aprestação das funcionalidades

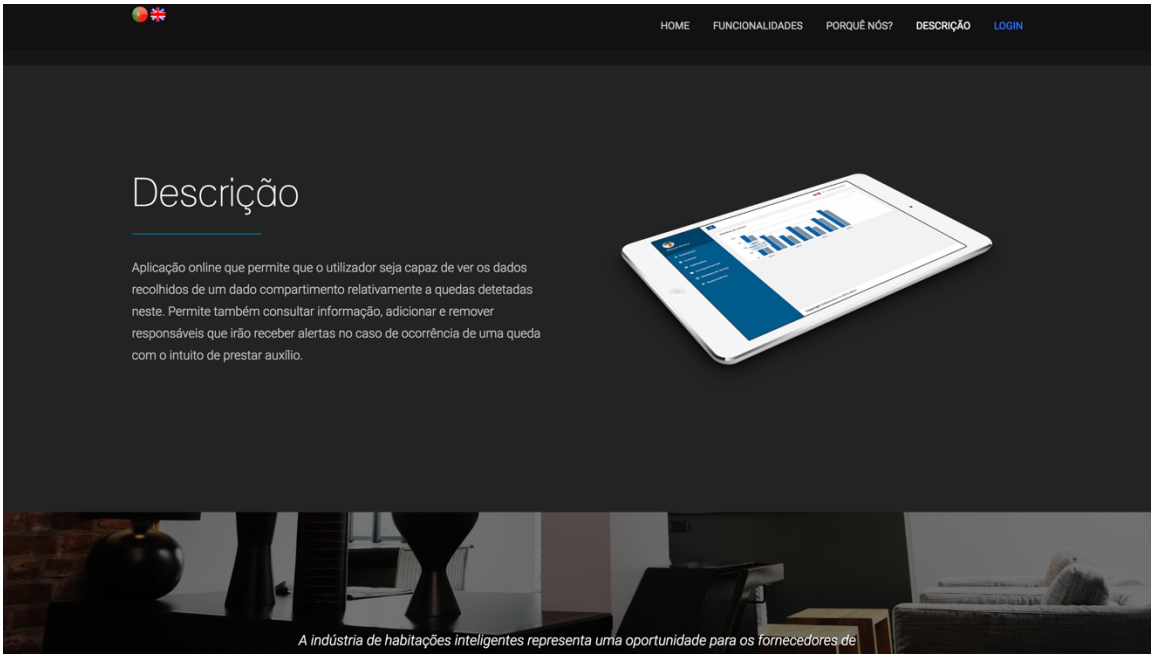


Figura 81 – Aplicação *web*: Descrição da solução

9.3.2 Sistema

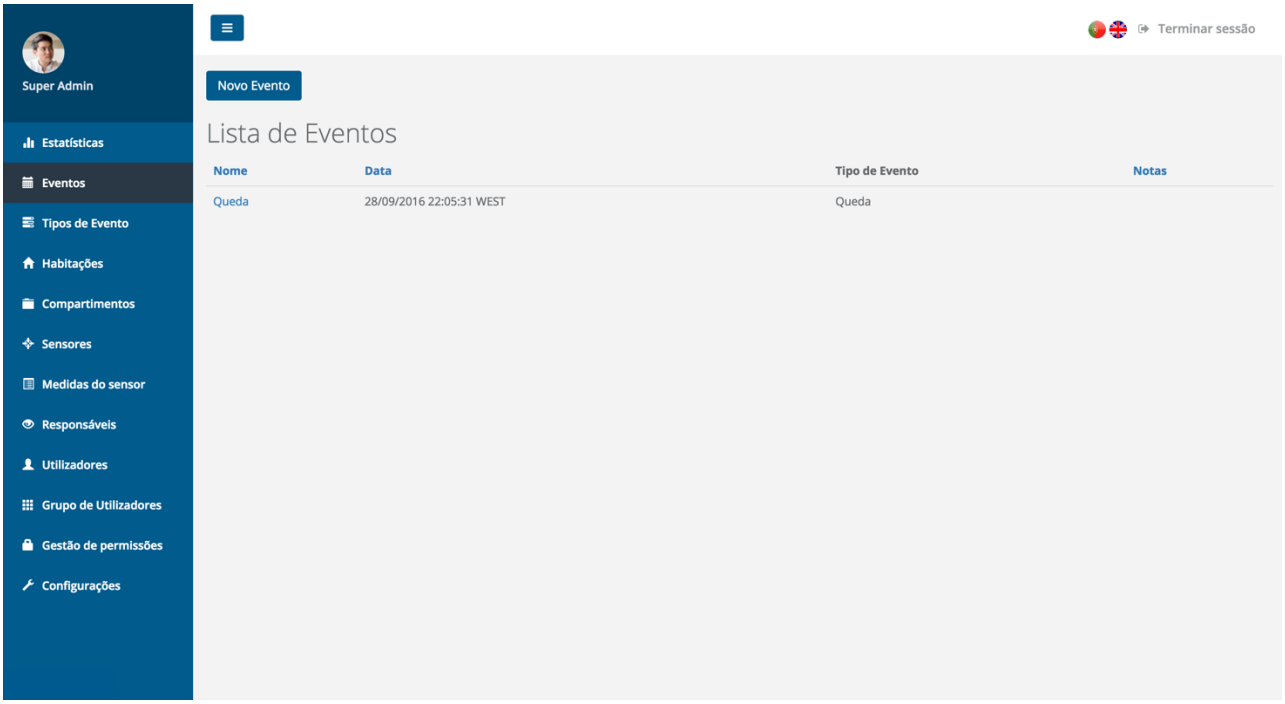


Figura 82 – Aplicação *web*: Lista de eventos detetados

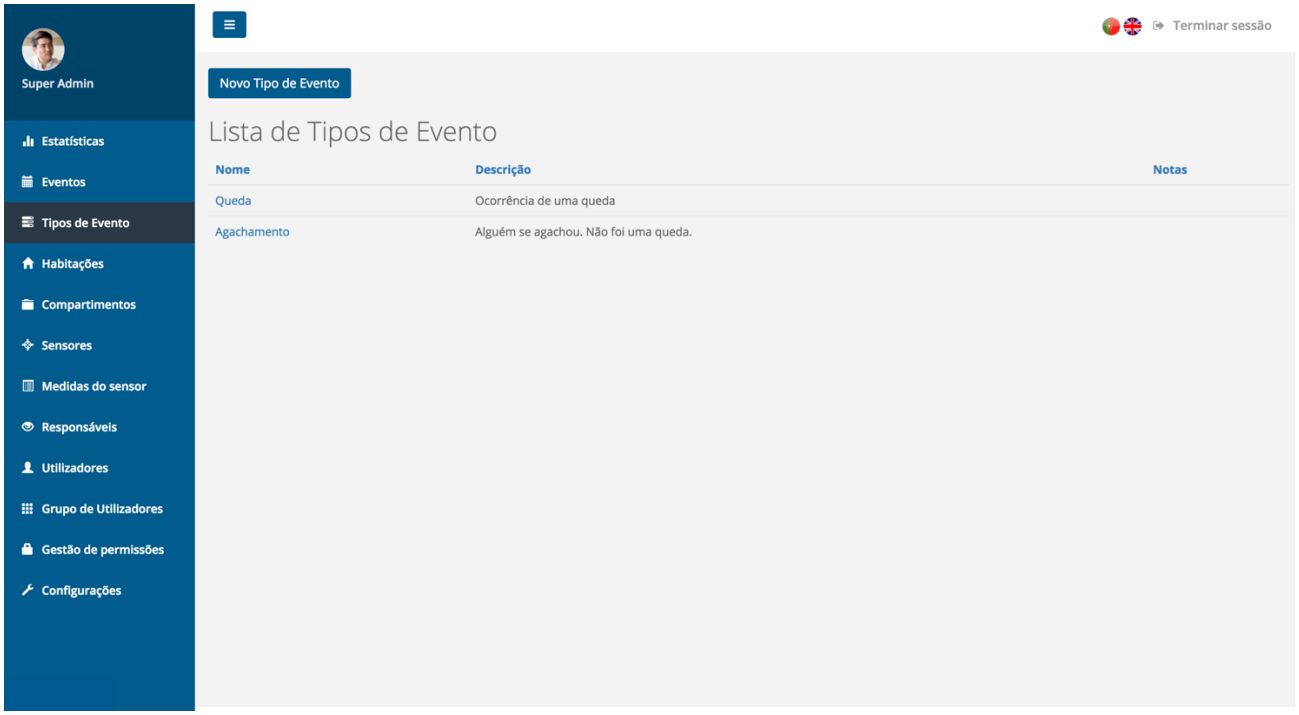


Figura 83 – Aplicação *web*: Tipos de evento

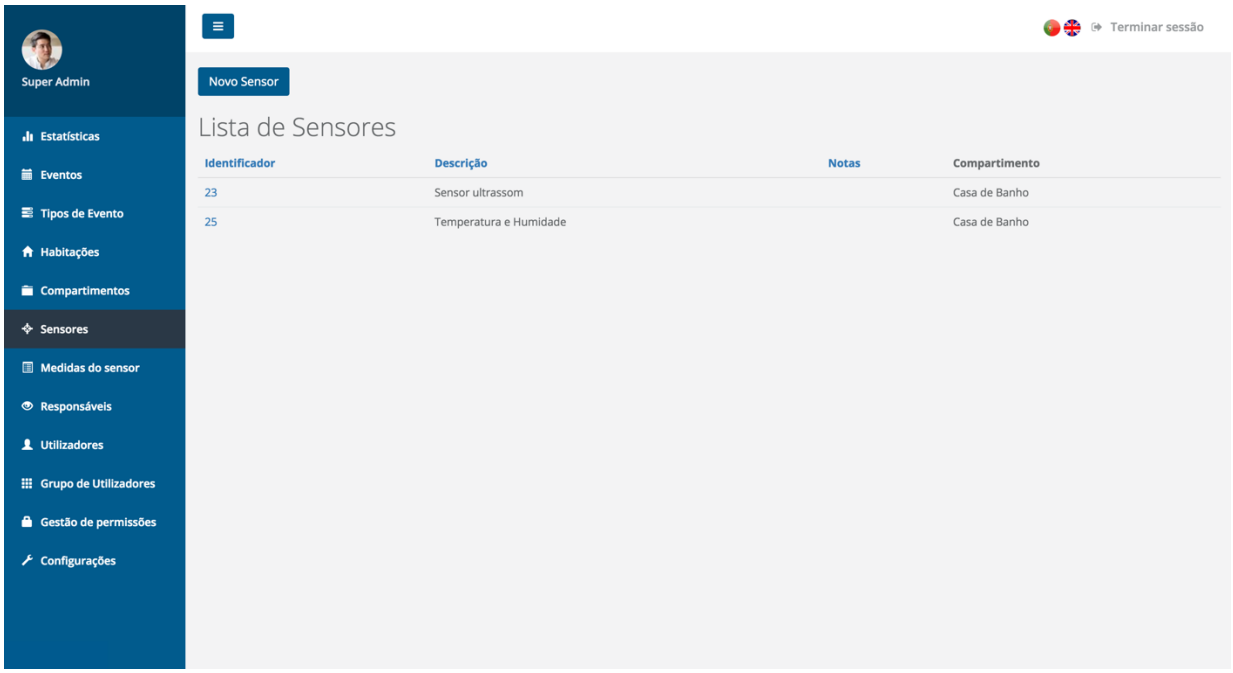


Figura 84 – Aplicação *web*: Sensores configurados

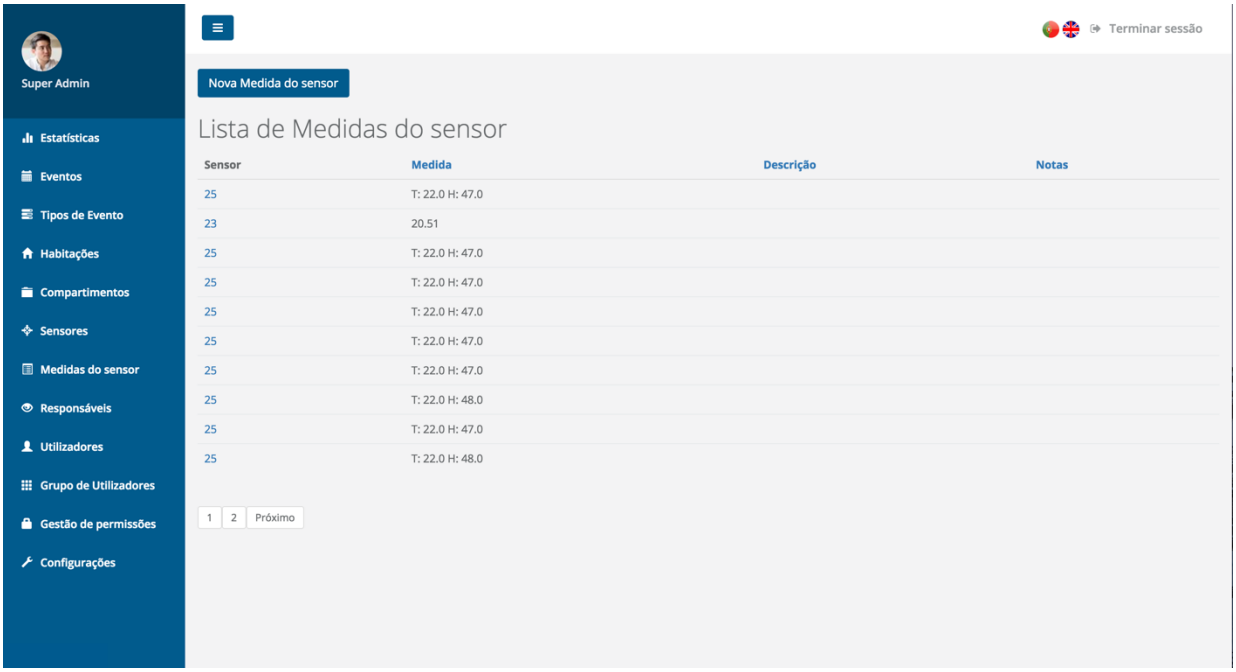


Figura 85 – Aplicação *web*: Medidas recolhidas pelos sensores

Figura 86 – Aplicação *web*: E-mail padrão para aviso de deteção de queda