# Sistemas de gestão para aplicações de domótica

Por

José Daniel Fernandes Carvalho

**Orientador:** Doutor Raul Manuel Pereira Morais dos Santos

**Co-orientador:** Doutor Francisco de Sousa Pereira

Dissertação submetida à
UNIVERSIDADE DE TRÁS-OS-MONTES E ALTO DOURO
para obtenção do grau de
MESTRE
em Engenharia Electrotécnica e de Computadores, de acordo com o disposto no Regulamento Geral dos Ciclos de Estudos Conducentes ao Grau de Mestre na UTAD DR, 2.ª série -N.º 133 - Regulamento n.º 658/2016 de 13 de julho de 2016


Vila Real, 2019

# Sistemas de gestão para aplicações de domótica

Por

José Daniel Fernandes Carvalho

**Orientador:** Doutor Raul Manuel Pereira Morais dos Santos

**Co-orientador:** Doutor Francisco de Sousa Pereira

Dissertação submetida à
UNIVERSIDADE DE TRÁS-OS-MONTES E ALTO DOURO
para obtenção do grau de
MESTRE
em Engenharia Electrotécnica e de Computadores, de acordo com o disposto no
Regulamento Geral dos Ciclos de Estudos Conducentes ao Grau de Mestre na UTAD
DR, 2.ª série -N.º 133 - Regulamento n.º 658/2016 de 13 de julho de 2016

Vila Real, 2019

*Orientação Científica :*

**Doutor Raul Manuel Pereira Morais dos Santos**

Professor Associado c/Agregação do
Departamento de Engenharias da Escola de Ciências e Tecnologia da
Universidade de Trás-os-Montes e Alto Douro

**Doutor Francisco de Sousa Pereira**

Professor Auxiliar do
Departamento de Engenharias da Escola de Ciências e Tecnologia da
Universidade de Trás-os-Montes e Alto Douro

Desenvolvimento de um sistema
de gestão para aplicações domóticas

*José Daniel Fernandes Carvalho*

Submetido na Universidade de Trás-os-Montes e Alto Douro
para o preenchimento dos requisitos parciais para obtenção do grau de
Mestre em Engenharia Electrotécnica e de Computadores

**Resumo** — A domótica tem como objetivo o controlo e automatização de sistemas numa habitação sejam elas feitas remotamente ou no próprio local, através de um sistema centralizado e/ou distribuído. Esta é uma área que se encontra em constante desenvolvimento graças à evolução existente em todas as áreas da eletrónica e das redes sem fios. As instalações destes sistemas são feitas usualmente através da implementação de interruptores, sensores, atuadores, entre outros, existindo já diversas tecnologias mais avançadas como o uso de câmaras de vídeo que registam eventos e servem como medida de vigilância. Estes equipamentos comunicam entre si recorrendo a ligações sem fios segundo a norma IEEE 802.11 (Wi-Fi) e com o suporte de uma nuvem computacional. O objetivo deste trabalho consiste na criação de uma interface web amigável e interativa que permitirá ao utilizador de um sistema domótico uma fácil e prática monitorização/controlo da sua própria casa. Para tal, foram selecionadas ferramentas de baixo custo como o *Raspberry Pi*, que neste cenário, será o elemento central do sistema, funcionando como intermediário de todas as mensagens que serão realizadas através do protocolo orientado para comunicação máquina-a-máquina, o MQTT. Pretende-se, também, a integração de captura de imagens proveniente de uma câmara que estará acoplada ao Raspberry Pi. O sistema desenvolvido foi testado em todas as suas vertentes de modo a garantir o bom funcionamento do mesmo.

**Palavras Chave:** Domótica, interfaces web, comunicações M2M, sistemas embebidos, redes sem fios.

Development of a management system
for home automation applications

*José Daniel Fernandes Carvalho*

Submitted to the University of Trás-os-Montes and Alto Douro
in partial fulfillment of the requirements for the degree of
Master of Science in Electrical Engineering and Computers

**Abstract** — The aim of domotics is to control and automate systems in a home whether it is done locally or remotely and using a centralized and / or distributed system. This is an area that is constantly evolving thanks to developments in all areas of electronics and wireless networks. These systems installations are usually made through the implementation of switches, sensors and actuators or through more advanced methods such as the implementation of cameras that record events and serve as a surveillance measure. These devices communicate with each other using wireless connections according to the IEEE 802.11 (Wi-Fi) standard and are supported by a computational cloud. This works' main objective is to create a friendly and interactive web interface that will allow to a home automation system's user an easy and practical monitoring/control of their own home. For this purpose were selected low-cost tools to use like Raspberry Pi, which in this scenario is the central element of the system serving as an intermediary of all messages that will be performed through the protocol designed for machine-to-machine communication, the MQTT.

It is also intended to integrate image capture from a camera that will be attached to Raspberry Pi. This developed system has been tested in all its aspects to ensure the proper functioning of the same.

**Key Words:** Domotics, web interfaces, M2M communications, embeded systems, wireless networks.

# Agradecimentos

Os meus agradecimentos ao Magnífico Reitor da Universidade de Trás-os-Montes e Alto Douro.

Ao Professor Doutor Raul Manuel Pereira Morais dos Santos, Professor Associado com Agregação do Departamento de Engenharias da Universidade de Trás-os-Montes e Alto Douro, orientador deste trabalho, pela sua motivação, pelas suas sugestões, ideias e orientações. Ao Professor Doutor Francisco de Sousa Pereira, Professor Auxiliar do Departamento de Engenharias da Universidade de Trás-os-Montes e Alto Douro, co-orientador deste trabalho, pela sua motivação, pelas suas sugestões, ideias e orientações.

A todos os meus amigos do *muy* nobre curso de Engenharia Eletrotécnica e de Computadores da Universidade de Trás-os-Montes e Alto Douro por todo o apoio, conselhos e camaradagem, não só nesta etapa, mas ao longo de todo o meu percurso académico.

Aos meus amigos de sempre, de São Torcato, pelo apoio, conselhos e camaradagem ao longo desta jornada.

À minha namorada Juliana por estar sempre presente, nos bons e maus momentos. Por todo o apoio, força, bons momentos proporcionados, conselhos transmitidos e paciência durante este percurso, que foram fundamentais para a conclusão do meu

percurso académico.
Por fim, agradecer às pessoas mais importantes na minha vida e que sem elas nada seria possível, a minha família, principalmente os meus pais e irmãos. Por todo o apoio, amizade, conselhos de vida, força e paciência que me deram para a conclusão do meu percurso académico.
A todos, um sincero obrigado!

UTAD, José Daniel Fernandes Carvalho
Vila Real, Janeiro de 2019

# Índice geral

# Índice de tabelas

# Índice de figuras

# Acrónimos e abreviaturas

## Lista de acrónimos

| **Sigla** | **Expansão** |
|---|---|
| IoT | *Internet of Things* |
| TCP | *Transmission Control Protocol* |
| M2M | *Machine-to-Machine* |
| MQTT | *Message Queuing Telemetry Transport* |
| WSN | *Wireless Sensor Networks* |
| I/O | *In/Out* |
| CoAP | *Constrained Application Protocol* |
| DIY | *Do It Yourself* |
| IP | *Internet Protocol* |
| AP | *Access Point* |
| DHCP | *Dynamic Host Configuration Protocol* |
| GPIO | *General Purpose Input/Output* |
| LED | *Light-Emitting Diode* |
| JSON | *JavaScript Object Notation* |

| **Sigla** | **Expansão** |
|---|---|
| PHP | *Hypertext Preprocessor* |
| KNX | *Konnex Networks* |
| IDE | *Integrated Development Environment* |
| NAT | *Network Address Translation* |
| HTTP | *Hypertext Transfer Protocol* |
| USB | *Universal Serial Bus* |
| PIR | *Passive Infrared Sensor* |
| LDR | *Light Dependent Resistor* |

# Lista de abreviaturas

| **Abreviatura** | **Significado(s)** |
|---|---|
| e.g. | por exemplo |
| et al. | e outros (autores) |
| etc. | etecetera, outros |
| i.e. | isto é, por conseguinte |
| vid. | veja-se, ver |
| vs. | versus, por comparação com |

# 1 Introdução

A domótica é uma tecnologia que tem como objetivo a gestão de todos ou de parte dos recursos eletrónicos existentes numa casa e pode ser feita de forma local ou remota. Neste primeiro capítulo é feita uma pequena introdução ao tema em questão, apresentado os principais objetivos e motivação deste trabalho. É, ainda, apresentado um enquadramento em que este assunto se insere através de uma abordagem à domótica e à Internet das Coisas (IoT - *Internet of Things*).

## 1.1 Domótica

A domótica está muitas vezes associada ao conceito de casa inteligente onde é possível o controlo de vários dispositivos eletrónicos de maneira a promover a eficiência energética, a qualidade de vida, a comodidade e até mesmo a segurança (Realinho, 2015).

Apesar desta ideia não ser recente, a sua adaptação ao quotidiano das pessoas tem-se mostrado um processo lento. Isto deve-se ao facto de a maior parte das soluções existentes, para além do custo elevado, serem desenhadas como um acréscimo às

tradicionais instalações elétricas, ou implementadas durante a construção da habitação (processo mais simples). Visto, também, que não existe uma norma global no que toca a este tipo de instalações, e consequente incompatibilidades entre equipamentos, torna-se ainda mais complexo a implementação de um sistema domótico.

Na maior parte dos casos, os recursos usados são interruptores, sensores e atuadores que possibilitam a automatização dos equipamentos eletrónicos com base no processamento dos dados que geralmente é feita através de uma unidade central de processamento onde todos os dispositivos eletrónicos se encontram ligados (Pellegrino et al., 2006). Gradualmente, vê-se um aumento da implementação deste tipo de tecnologias, que acompanha igualmente o desenvolvimento de outras tantas necessárias e que se apoiam para sustentar a domótica como o caso dos protocolos de comunicação entre equipamentos, como o Wi-Fi e a evolução de pequenos mas eficazes dispositivos *low-cost*, como sensores e atuadores, de fácil inserção na rede e simples de controlar.

## 1.2 Internet das Coisas

Atualmente, a Internet é usada por milhões de pessoas em todo o mundo, seja para enviar e-mails e aceder a conteúdo multimédia bem como aceder às redes sociais entre outras aplicações que a Web permite. Para além destas formas de a utilizar, surge uma mais recente que consiste em usar as plataformas Web como uma nuvem onde máquinas são capazes de comunicar entre si através de dados recolhidos por sensores, por exemplo, e onde o utilizador é capaz de aceder a essa informação, sendo assim possível a criação de redes de objetos. Desta forma abre-se caminho a novas funcionalidades e novas ideias a explorar e a serem implementadas. Esta quantidade imensa de dados gerados pela coleção de dados pode ser então analisada e criar assim um ambiente inteligente capaz de satisfazer as necessidades do utilizador de uma forma transparente. Desta forma surge o conceito de computação ubíqua que, como referido anteriormente, é uma forma transparente de comunicação entre Homem e máquina, onde as máquinas serão os objetos do dia a dia repletos de

tecnologia, sensores, maioritariamente, que captam os movimentos dos habitantes, seja doméstico, ou no trabalho, medem a temperatura, som, nível de luz entre outros. Através da análise destes dados e através de atuadores é possível então a criação de ambientes inteligentes que permitem uma melhoria na qualidade de vida, uma maior comodidade, bem como uma poupança a nível financeiro que poderá advir de uma gestão mais eficiente de recursos

Como seria de esperar, a exploração destas ideias abre diversos caminhos. Um deles é mesmo a implementação destas tecnologias em casas inteligentes onde, através de uma nuvem IoT, se torna possível a comunicação entre aparelhos. Das várias plataformas e protocolos de interface máquina-a-máquina (M2M) existentes, foi escolhido o protocolo MQTT (*Message Qeueing Telemetry Transport*), um protocolo bastante simples e leve, desenhada especificamente para a comunicação M2M (Miorandi et al., 2012).

## 1.3 Motivação e enquadramento

A motivação desta dissertação reside em investigar sistemas embebidos em contexto de domótica, a integração destes em habitações e interfaces gráficas que permitam aos utilizadores interagirem com os recursos disponíveis de uma forma mais natural. O estudo das interfaces e a gestão de uma rede de dispositivos distribuídos por uma habitação permitirão um ganho importante de competências nesta área de trabalho, aliado à instalação de um protótipo de sistema de interface baseado em Raspberry Pi, uma ferramenta de baixo custo com inúmeras aplicações. Contribui-se assim para dissertar sobre um tema atual e dinâmico fomentando novas soluções e abordagem à domótica que teve uma grande evolução nos últimos anos.

## 1.4 Organização da dissertação

Esta dissertação encontra-se estruturada em seis capítulos.

Neste primeiro capítulo faz-se uma introdução ao tema da dissertação e são apresentados as motivações e objetivos do trabalho.

No capitulo 2 é feito um estudo mais aprofundado às tecnologias que sustentam o conceito IoT, caraterísticas de dispositivos de uma rede de sensores e de algum dos protocolos de comunicação usados para este tipo de redes. É feita também uma abordagem do uso destas tecnologias na domótica. Por fim, são analisadas algumas soluções comerciais nessa área.

O capítulo 3 é referente à explicação do sistema proposto que engloba a sua arquitetura, a criação da rede onde o sistema irá operar, a comunicação entre elementos, bem como os casos de uso.

No capítulo 4 é apresentado como se procedeu à implementação do sistema proposto, explicando as ferramentas usadas, tanto de *hardware* como de *software*. Também neste capítulo, são apresentadas as várias páginas da interface Web criada.

O capítulo 5 dedica-se à apresentação dos testes realizados e respetivos resultados.

Por fim é feita uma conclusão geral do projeto, apresentando ainda algumas perspetivas de trabalho futuro.

# 2 Internet das Coisas

Neste capítulo são apresentados alguns conceitos subjacentes às tecnologias IoT, aos dispositivos que lhes estão associados e as suas aplicações na domótica. De facto, as estatísticas têm apontado a domótica/*smart homes* como uma das principais aplicações da IoT, segundo se pode ver na Figura 2.1 (Datafloq).

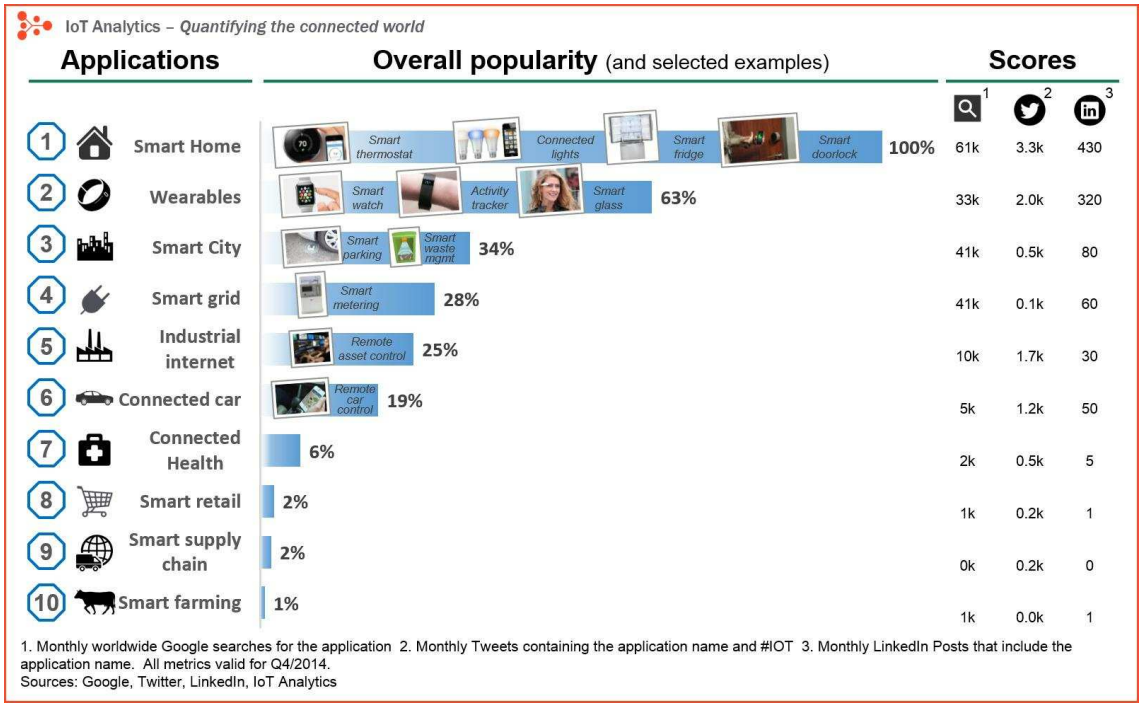


**Figura 2.1** – Aumento das aplicações na área da automação industrial.

## 2.1 O conceito de IoT

As tecnologias do domínio da IoT são das que apresentam mais desenvolvimento nas chamadas ICT (*Information and Computing Tecnologies*). Em muita literatura não existe uma definição clara e transversal sobre este paradigma, isto pelo simples facto dos dois termos que o compõe – "Internet" e "Things" – criam, à partida, alguma controvérsia. O primeiro remete para uma rede orientada enquanto que o segundo, *objetcts* surge de uma forma genérica que serão integrados num *framework* comum. Outro dos fatores de incerteza são como as próprias entidades de pesquisa, empresas e *stakeholders* se referem a esta ideia, seja ela de uma forma orientada à Internet ou mais orientada aos objetos, o que é certo é que a sua visão depende também dos seus interesses e finalidades. Semanticamente, os dois termos juntos significam uma "rede mundial de objetos interligados com endereços únicos baseados num sistema de comunicação normalizado" (Li et al., 2015). Ou seja, isto significa mais uma visão

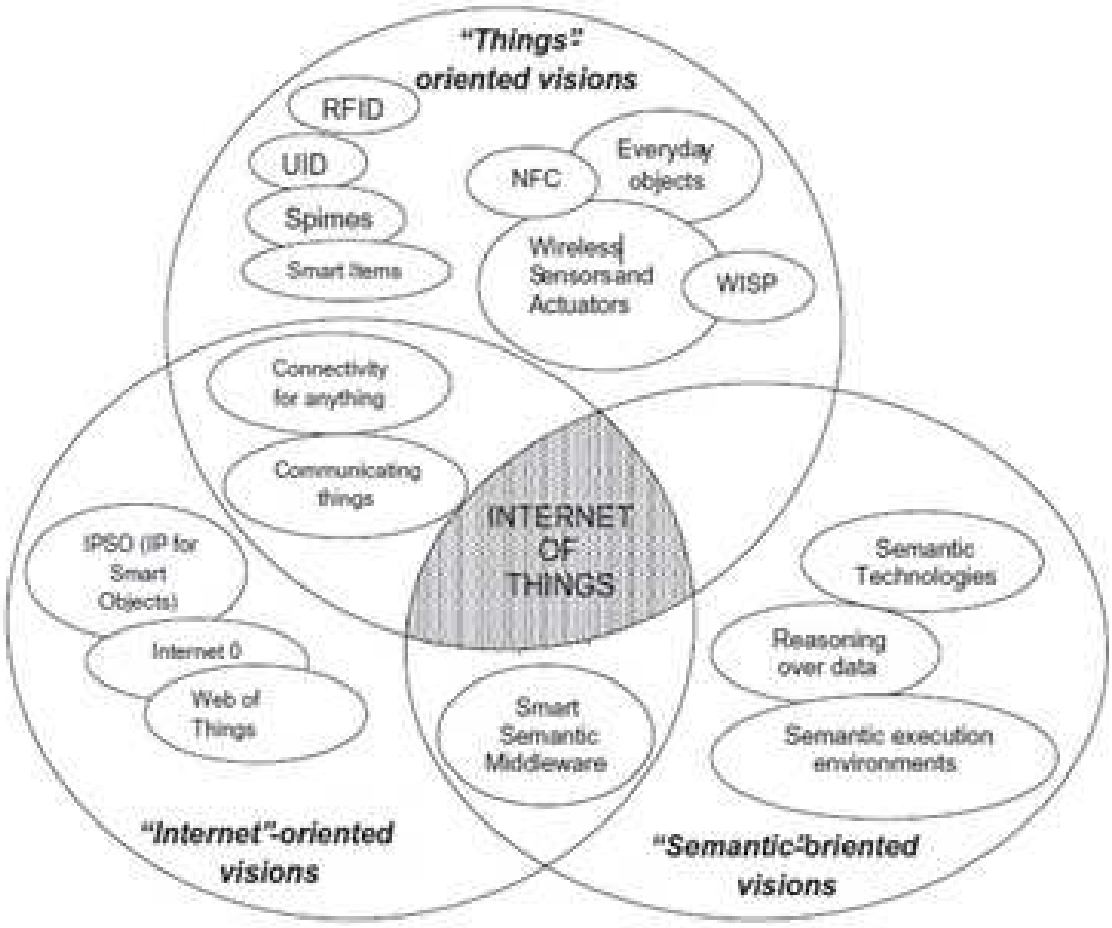


**Figura 2.2** – As várias visões da IoT (Li et al., 2015).

do paradigma. A Figura 2.2 ilustra algumas ideias deste conceito onde se pretende juntar todas as suas visões, constituindo, assim a Internet das Coisas:

Como referido por Li et al. (2015), a ideia base deste conceito é a presença à nossa volta de inúmeros objetos, como etiquetas RFID (*Radio-Frequency Identification*), sensores, atuadores, etc, que, através de esquemas de endereçamento, são capazes de comunicar entre si (comunicação M2M) de forma a chegar a um objetivo comum definido pelo utilizador.

## 2.2 Dispositivos para IoT

De uma forma conceptual e segundo Miorandi et al. (2012), os objetos inteligentes que compõem as tecnologias IoT baseiam-se em três pilares, sendo eles:

- Capacidade de se identificarem;
- Capacidade de comunicarem;
- Capacidade de interagir.

Desta forma é possível criar uma rede de objetos interligados entre si ou entre *end-users* ou outras identidades na rede. Esses objetos são definidos usualmente pelas seguintes condições:

- Possuírem um corpo físico com caraterísticas físicas associadas, como altura e forma, por exemplo.
- Têm capacidades mínimas de comunicação entre elas, a capacidade de serem descobertas, receber e responder mensagens.
- São detentoras de um único identificador.
- Possuem um nome e um endereço, sendo o nome humanamente legível para processamento de dados e o endereço, por sua vez, é uma *string* capaz de ser lida por máquinas, possibilitando a comunicação.
- Possui capacidades básicas de computação que podem ir desde combinar uma dada mensagem a uma etiqueta RFID, até à capacidade de realizar operações mais complexas, como a descoberta de serviços e gestão da rede.
- Pode possuir sensores de fenómenos físicos (por exemplo, temperatura, luz, radiação eletromagnética) ou ter a capacidade de disparar atuadores.

Este último fator é a chave para permitir diferenciar os objetos inteligentes das entidades tradicionais de uma rede.

## 2.3 Redes de sensores sem fios

Com os avanços na tecnologia em circuitos integrados de baixa potência e comunicações sem fios tornou-se possível a criação de pequenos dispositivos de baixo custo, também eles de baixa potência, úteis para aplicações remotas de sensoriamento. A combinação destes fatores promoveu as redes de sensores que consistem num grande número de sensores inteligentes capazes de recolher, processar, analisar e difundir dados importantes dos ambientes onde estão instalados. Todos os dados dos sensores são compartilhados entre nós e enviados para um sistema distribuído ou centralizado para análise (Gubbi et al., 2013).

As funções desejáveis para as redes de sensores sem fios incluem:

- Facilidade de instalação;
- Autoidentificação:
- Autodiagnóstico;
- Confiabilidade;
- Reconhecimento de tempo para coordenação com outros nós;
- Funções de software e processamento digital de sinal (*Digital Signal Processing*-DSP);
- Protocolos de controlo padronizados;
- Interfaces de rede (Townsend and Arms, 2004).

## 2.4 Aplicações na domótica

Com a criação de redes de objetos 'inteligentes' ligados entre si, é possível o desenvolvimento de várias aplicações nas mais diversas áreas, entre elas a área da domótica.

Uma dessas aplicações, é um sistema de saúde omnipresente. Através de múltiplos sensores embebidos espalhados pela habitação, é possível retirar informação sobre parâmetros fisiológicos dos habitantes que por sua vez podem possuir, ou não, sensores embebidos no corpo ou implantados, sendo possível a recolha de valores, como tensão arterial, glicose no sangue, etc. Essa informação é posteriormente passada e guardada por servidores, que pode ser gerida por uma aplicação móvel. Para além de uma aplicação para casas 'inteligentes', é um sistema ideal para monitorização em lares de idosos, que permite que um médico consiga fazer um diagnóstico remotamente. Desta forma é possível reduzir custos hospitalares, pois, os pacientes são acompanhados de uma forma mais assídua antecipando alguma doença, ou mal-estar, e consequente tratamento (Friedewald et al., 2005).

Em termos de automação residencial, os utilizadores têm a possibilidade de controlar vários aspetos no ambiente da casa como a luz, estores, regulação de temperatura, bem como o controlo de equipamentos domésticos como frigoríficos, máquinas de lavar roupa, entre outros, de forma remota. Para além do controlo direto do utilizador, haverá a possibilidade de, através da recolha de dados pelos diversos sensores presentes na habitação, agir autonomamente, realizando ações sobre os dispositivos de acordo com parâmetro estipulados pelos utilizadores. Com isto resultará num consequente aumento da eficiência energética. Outro desenvolvimento interessante, será a criação de uma espécie de rede social para os dispositivos, onde poderão publicar periodicamente informações sobre o seu estado atual, podendo ser visualizadas por utilizadores. Apesar de assim, ser criada uma *framework* usando um sistema de nuvem para aceder à informação, irá ser necessário um novo paradigma de segurança para a total concretização destas ideias (Gubbi et al., 2013).

## 2.5 Comunicação Máquina-a-Máquina

Cada vez mais vemos espalhados por qualquer parte dispositivos ditos inteligentes que são capazes de comunicar entre si de uma forma transparente e autónoma originando assim comunicações do tipo M2M, e que a passo a passo vêm substituindo

as tradicionais operações entre homem-máquina. Estima-se que os dispositivos disponíveis M2M estão a crescer exponencialmente, prevendo um crescimento dos 50 milhões em 2008 para mais de 200 milhões em 2014 e até mais de 50 mil milhões por volta de 2020. Entenda-se por dispositivos M2M qualquer máquina capaz de recolher eventos ou gerar dados por si própria e consequente envio desses dados para outros dispositivos através de uma rede quer seja com ou sem fios (Chen et al., 2004).

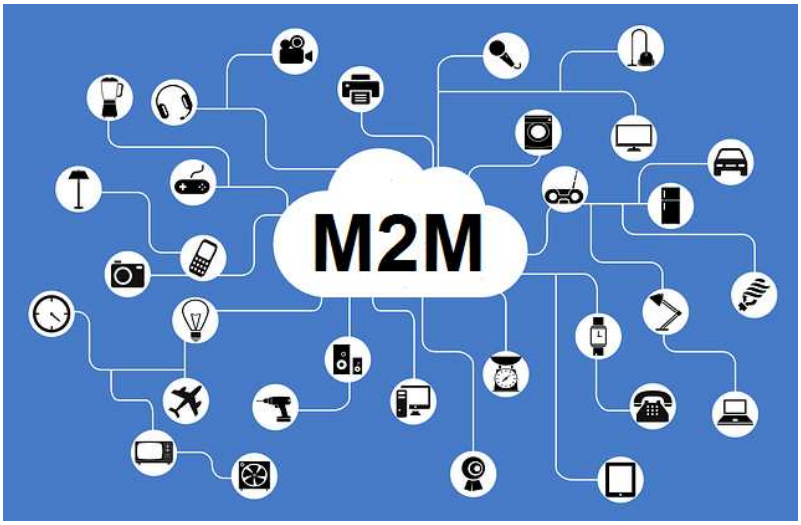


**Figura 2.3** – Ilustração de uma rede M2M (Mittal, 2016)

Uma rede M2M é ligeiramente diferente das tradicionais, uma vez que um dos aspetos mais relevantes é que a maior parte destes dispositivos são alimentados por baterias, o que dita a necessidade de uma maior eficiência energética. Uma rede é constituída por um vasto número de nós que podem ser máquinas ou objetos, tais como computadores, dispositivos móveis, sensores/atuadores inteligentes, entre outros, ou seja, pode ser qualquer coisa que esteja à nossa volta. Nesse sentido, de forma a tantos dispositivos poderem comunicar entre si é necessário que o custo dessas comunicações seja baixo gerando assim, cada nó, pequenas quantidades de informação, mas que toda junta, isto é, todos os dados gerados de todos os nós da rede levam a uma quantidade enorme de dados produzida (Kim et al., 2014). Esta informação pode ser gerida por um *gateway* responsável pela interação entre todos os terminais, pela comunicação num certo espaço e entre outra redes (Zhang et al.,

2011).

Com uma quantidade enorme de objetos embebidos à nossa volta comunicando entre si pode-se imaginar vários cenários onde esses mesmo objetos poderão trazer benefícios para todas as pessoas. Desta forma, é apresentado na Tabela 2.1 (In, 2011) alguns exemplos onde a comunicação M2M pode ser aplicada.

| | |
|---|---|
| **Segurança privada e pública** | Sistemas de vigilância; Controlo de acesso a edifícios;<br>Monitorização ambiental (ex.: desastres ambientais) |
| ***Smart Grid*** | Eletricidade; Gás; Água; Temperatura;<br>Controlo da rede; Resposta à demanda;<br>Metering industrial. |
| **Localização e rastreamento** | Controlo de pedidos; Rastreamento de ativos;<br>Monitorização de pessoas |
| **Veículos** | Segurança do veiculo e do condutor;<br>Navegação aprimorada;<br>Informações de trânsito;<br>Diagnostico remoto do veiculo |
| **Pagamentos** | Ponto de venda, ATM;<br>Máquinas de venda automática;<br>Máquinas de jogos |
| **Cuidados de saúde** | Monitorização dos sinais vitais;<br>Apoio aos idosos ou deficientes;<br>Pontos de tele-medicina de acesso à web;<br>Diagnostico remoto. |
| **Manutenção e controlo remotos** | Automação industrial; Sensores;<br>Iluminação; Bombas;<br>Controlo de máquinas de venda automática. |
| **Dispositivos do consumidor** | Moldura digital, camera digital;<br>*Ebook*; *Hubs* de domótica. |

**Tabela 2.1** – Exemplos de aplicações M2M.

## 2.6 Protocolos MQTT para telemetria

O protocolo MQTT (*Message Queuing Telemetry Transport*), orientado à mensagem, foi promovido pela IBM e é direcionado a comunicações M2M de baixo custo. É assíncrono e corre sobre a pilha protocolar TCP. Para a troca de mensagens baseia-se num protocolo publicação/subscrição (pub/sub), onde os clientes se subscrevem

a tópicos, ficando à escuta de mensagens proveniente dos publicadores (Light). Os princípios deste design são então minimizar a largura de banda da rede e os recursos do dispositivo, tentando ao mesmo tempo garantir a fiabilidade e a garantia de entrega das mensagens. Estas características tornam este protocolo ideal para a comunicação M2M bem como para as IoT visto que, ao contrário dos protocolos de pedido/resposta, não é necessário pedir atualizações do sistema, existindo assim menos largura de banda e reduzidos requisitos computacionais. Isto torna-as ideais para aplicações móveis onde a reduzida largura de banda e a autonomia das baterias são fundamentais (Saraswathi et al., 2018).

O tipo de sistema de publicação/subscrição são baseados em tópicos o que quer dizer que a lista de tópicos é conhecida em antemão. Por exemplo, aquando do desenvolvimento de uma aplicação as subscrições e publicações são feitas de acordo com os tópicos (Gordon, 2010), existindo um negociador (*broker*), conforme ilustrado na figura 2.4.

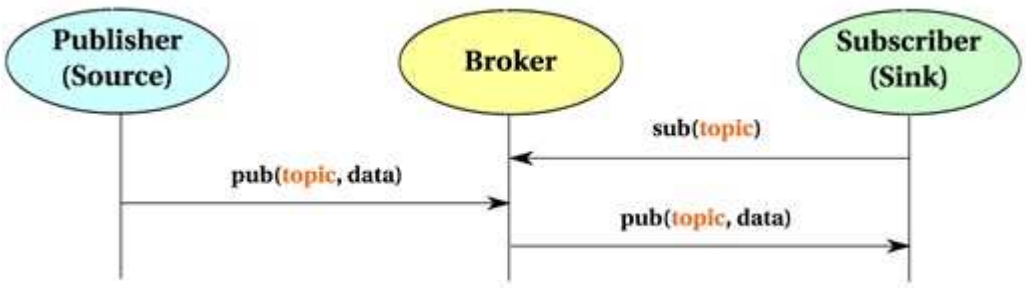


**Figura 2.4** – Modelo de comunicação MQTT (Gordon, 2010).

A interação entre os dispositivos, como já referido, é feita através da publicação de mensagens e subscrição a tópicos, isto é, múltiplos clientes ligam-se ao *broker* (responsável pela gestão das mensagens), que contém os tópicos, e subscrevem-se aos tópicos em que estão interessados. Os clientes também podem publicar mensagens num tópico e todos os outros que estão subscritos a esse tópico receberão essa mensagem. Para melhor perceber esses conceitos, apresenta-se de seguida, informação mais detalhada dos principais elementos da comunicação MQTT.

1. *Broker*: Este é o elemento central do protocolo do tipo publicação/subscrição e é responsável por receber todas as mensagens provenientes dos clientes, filtrar essas mensagens, determinar quem está subscrito ao quê e enviar as mensagens para os clientes subscritos. O *broker* também mantém as sessões de todos os clientes persistentes, incluindo assinaturas e mensagens perdidas. Outra responsabilidade, é a autorização e autenticação dos clientes.

2. Tópicos: No MQTT, a palavra tópico refere-se a uma *string* UTF-8 que o *broker* usa para filtrar mensagens para cada cliente conectado. O tópico consiste em um ou mais níveis. Cada nível do tópico é separado por uma barra invertida ("/"). De seguida serão apresentadas algumas caraterísticas.

    - *Wildcards* - "+"este carácter representa um nível de um tópico, por exemplo: "casa/+/temperatura". O símbolo "#"representa múltiplos níveis num tópico. Este carácter deve ser colocado apenas no final de cada tópico, por exemplo: "casa/alarmes/#", isto significa que o cliente esta a subscreve-se ou publicar a todos os alarmes que existem na "casa".
    - Tópico começados por "$- Estes tópicos são especiais, visto que são tópicos reservados a estatísticas internas do *broker* MQTT, onde os clientes não podem publicar mensagens.
      Um exemplo, é: "$SYS/*broker*/*clients*/*connected*", que nos indica o número de clientes ligados ao *broker* (HiveMQ, b).

3. Publicador:

    - Tópico - Um cliente MQTT pode publicar mensagens num tópico assim que se liga ao *broker*. Esse tópico é usado pelo *broker* que lhe permite ter a informação necessária para reencaminhar a mensagem para outros clientes que estão subscritos.
    - Nível de QoS - Indica o nível da qualidade de serviço da mensagem. O nível de QoS determina que tipo de garantia uma mensagem tem para alcançar o destinatário pretendido (cliente ou *broker*).

- *Retain flag*, que no caso de estar ativa, envia a última mensagem válida (guardada no *broker*) para um tópico especifico.
- *Payload*, que é o verdadeiro conteúdo da mensagem, que pode ser leituras de sensores, texto em qualquer formato, dados encriptados, imagens ou qualquer forma de dados binários.
- Identificador de mensagem, que tal como o nome indica, identifica uma mensagem enquanto ela é transmitida do cliente para o broker. Esta caraterística apenas é relevante para níveis de QoS superiores a zero. Este identificador é definido pelo broker ou pelas bibliotecas do cliente.
- *Flag* DUP - esta flag indica que a mensagem é duplicada e foi reenviada porque o destinatário pretendido (cliente ou *broker*) não confirmou a mensagem original. Isso é relevante apenas para QoS maior que 0.

4. Subscritor: Para receber mensagens de um publicador, o subscritor tem que se subscrever num dado tópico, enviando essa informação ao *broker*. Para caraterizar um subscritor, basta apenas dois elementos, que são:
   - Identificador de mensagens - igual ao do Publicador.
   - Lista de subscrições - a mensagem de subscrição enviada ao *broker* pode conter múltiplas subscrições para o mesmo cliente. Cada subscrição é composta por um tópico e o nível correspondente de QoS.Se existir sobreposição de subscrições para um cliente, o *broker* envia a mensagem para o nível de QoS mais alto (HiveMQ, a).

Para garantir a fiabilidade no envio das mensagens o MQTT assegura três níveis de qualidade de serviço QoS (*Quality-of-Service*):

- *Fire and Forget*: a mensagem é enviada uma vez e não é necessária a confirmação da entrega – QoS0;
- *Delivered at least once*: a mensagem é enviada pelo menos uma vez, sendo a confirmação necessária – QoS1;

- *Delivered exactly once*: um mecanismo chamado *four-way handshake*, ilustrado na Figura 2.5 é usado para garantir que a mensagem é enviada exatamente uma vez – QoS2.

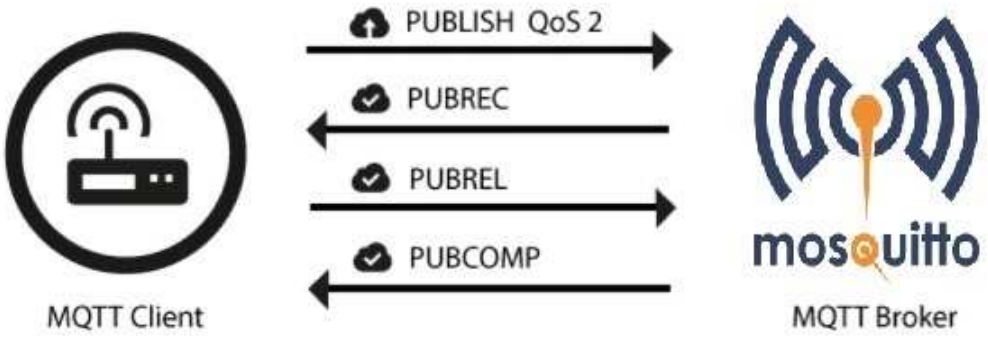


**Figura 2.5** – *Four-way handshake.*

### 2.6.1 Redes de sensores MQTT-SN

Ao contrário do MQTT original, que corre sobre TCP, e tal como o nome indica, é direcionado e otimizado para redes de sensores, correndo sobre o protocolo UDP. Uma das diferenças na comunicação dentro deste tipo de redes é a implementação de um *gateway* que serve como tradutor das mensagens entre os sensores/atuadores e o *broker* que se encontra na 'rede tradicional'. A arquitetura deste protocolo encontra-se ilustrada na Figura 2.6.

O MQTT é uma mais-valia para uma estrutura deste género, visto que os publicadores e subscritores que estão associados de acordo com os tópicos, ao longo do tempo podem ser alterados dinamicamente sem os próprios publicadores e subscritores tenham noção disso. Isto é interessante para redes de sensores, onde os dispositivos podem falhar e assim ser necessário adicionar novos, ou mesmo para poder estender a rede a qualquer momento (Gordon, 2010).

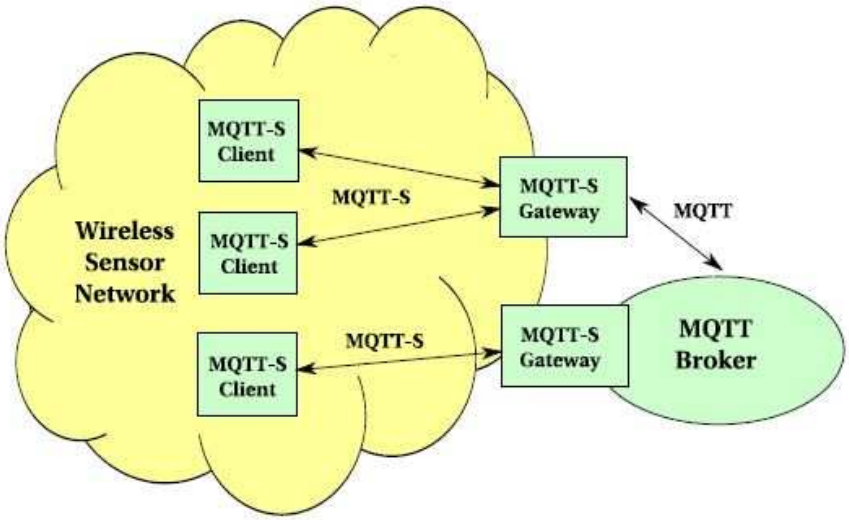


**Figura 2.6** – Arquitetura MQTT-SN (Gordon, 2010) .

### 2.6.2 Segurança no protocolo MQTT

Singh et al. (2015) propuseram um aumento de segurança para o MQTT, denominado SMQTT, onde tinha como principal objetivo o aumento da segurança associado a este protocolo. O sistema funciona da mesma forma, através de mensagens pub/sub, mas a mensagem correspondente a publicação passa agora a ser chamada de *Spublish* onde a mensagem é encriptada com ajuda do protocolo *Attribute Based Encription* (ABE). Para manter o sistema leve adaptaram o esquema ABE baseado no protocolo, também ele leve, de criptografia de curva elíptica. Assim sempre que um cliente queira publicar uma mensagem encriptada usa o comando *Spublish* e todos os subscritores que satisfaçam a política de acesso ao tópico são capazes de desencriptar a mensagem.

## 2.7 Protocolos para dispositivos de recursos limitados

*Constrained Application Protocol* (CoAP) é um protocolo projetado pela *Internet Engineering Task Force* (IETF) orientado a dispositivos com recursos limitados, à semelhança do que acontece no MQTT. Este protocolo baseia-se numa arquitetura chamado *Representational State Transfer* (REST) sendo um estilo de arquitetura que define um conjunto de restrições e propriedades baseados em HTTP funcionando assim num modelo de pedido/resposta síncrono. Para além deste modelo, o CoAP também suporta arquitetura publicação/subscrição, utilizando o método GET. Ao contrário do MQTT, este modelo usa *Universal Resource Identifier* (URI) em vez dos tópicos, sendo as duas maneiras idênticas (Thangavel et al., 2014).

A grande diferença entre estes dois protocolos é que o CoAP é executado no protocolo UDP enquanto o MQTT corre sobre o TCP. Isto acontece devido ao facto do UDP permitir uma redução na ocupação da banda larga e eliminar os cabeçalhos do TCP. Para além disso, o CoAP suporta ainda *unicast* bem como *multicast*, enquanto que o TCP por natureza não é orientado a comunicações *multicast* (Karagiannis et al., 2015). Outro fator é que o UDP é considerado um protocolo não confiável, pelo que o CoAP fornece o seu próprio mecanismo de confiabilidade e respetivo nível QoS (apesar deste não apresentar essa diferenciação) através do uso de quatro tipo de mensagens:

- Confirmadas: precisam de confirmação (*Acknowledgement* - ACK);
- Não confirmadas: não necessitam de confirmação;
- Confirmação: confirma a receção de uma mensagem certa;
- Reset: confirma a receção de uma mensagem que pode não ser processada.

Apesar do CoAP ter sido criado para as tecnologias IoT e para comunicação M2M,

não inclui nenhum sistema integrado de segurança. Para assegurar que as comunicações neste protocolo são seguras, é usado um outro mecanismo chamado de DTLS (*Datagram Transport Layer Security*). O DTLS corre sobre o UDP sendo similar ao TLS do TCP. Este protocolo traz algumas restrições ao CoAP que, como não permite a difusão *multicast*, pode aumentar a largura de banda necessária, os recursos computacionais e diminuir assim a autonomia da bateria dos dispositivos IoT. As principais diferenças dos dois modelos encontram-se sumariados na Tabela 2.2 (Thangavel et al., 2014).

| | **MQTT** | **CoAP** |
|---|---|---|
| **Camada da aplicação** | Camada única | Camada única com duas subcamadas (Mensagens e pedidos/resposta) |
| **Camada de transporte** | TCP | UDP |
| **Mecanismo de segurança** | 3 níveis de QoS | Mensagens confirmáveis, não confirmáveis, reconhecimentos e retransmissões |
| **Arquiteturas suportadas** | Publicação/Subscrição | Pedido/Resposta Publicação/Subscrição usando URI |

**Tabela 2.2** – Comparacão entre MQTT e CoAP (Thangavel et al., 2014).

Para além destes dois protocolos existem outros, como o *Extensible Messaging and Presence Protocol* (XMPP), o *Representational State Transfer* (REST), o *Advanced Message Queuing Protocol* (AMQP) e *WEBSOCKETs*, todos eles protocolos ao nível da camada da aplicação, orientados às tecnologias IoT. As suas diferenças estão resumidas na Tabela 2.3.

| **Protocolo** | **Transporte** | **Níveis QoS** | **Arquitetura** | **Segurança** |
|---|---|---|---|---|
| **CoAP** | UDP | Sim | Pedido/Resposta Publicação/Subscrição (URI) | DTLS |
| **MQTT** | TCP | Sim | Publicação/Subscrição | TLS/SSL |
| **XMPP** | TCP | Não | Pedido/Resposta Publicação/Subscrição | TLS/SSL |
| **REST** | HTTP | Não | Pedido/Resposta | HTTPS |
| **AMQP** | TCP | Sim | Publicação/Subscrição | TLS/SSL |
| **Web Socket** | TCP | Não | Cliente/Servidor Publicação/Subscrição | TLS/SSL |

**Tabela 2.3** – Protocolos para dispositivos de recursos limitados (Karagiannis et al., 2015).

## 2.8 Soluções comerciais de domótica

A maioria das soluções comerciais disponıveis no mercado integram a componente de hardware e software. Isto implica que quando se pretende utilizar uma aplicação móvel de um determinado fabricante é-se obrigado a comprar os seus meios físicos para estabelecer comunicação com a aplicação. No entanto, tem aparecido várias propostas que tentam diminuir este tipo de problemas, como as soluções *open-source*.

### 2.8.1 MEO Smart Home


**Figura 2.7** – MEO Smart Home (Pedro Pinto, 2017a).

O Meo Smart Home é uma plataforma criada pela empresa portuguesa MEO e lançada em 2017. É um sistema de proteção e controlo à distância da sua casa ou empresa que não carece de orçamentos ou deslocações prévias. É fácil de instalar e utilizar. Com recurso a um smartphone, tablet ou PC, o Meo Smart Home permite, nomeadamente:

- Receber alertas em tempo real por chamada, SMS ou email sempre que se detetar uma intrusão (através da abertura das portas, janelas ou detetado movimento através da câmara) ou, por exemplo, a deteção da presença de fumo;

- Verificar, em tempo real, as áreas vigiadas (de dia ou de noite e consultar gravações em vídeo;
- Efetuar gravações automáticas em caso de alerta;
- Consulta de eventos.

O pacote inicial é composto por uma câmara de vídeo, um sensor de abertura de portas e janelas, uma sirene interior, um detetor de fumo, um telecomando, uma central de controlo, bem como uma placa e autocolantes dissuasores, com a possibilidade de adquirir outros dispositivos separadamente, tornando-se adaptável às necessidades de cada cliente e sendo possível a gestão dos dispositivos facilmente.

Na interface, os clientes poderão aceder a vídeo em direto, que pode gravar continuamente durante 24 horas e armazenar gratuitamente até 30 dias. Sempre que for, por exemplo, detetado movimento ou acionado o alarme de intrusão, incêndio ou outro, os clientes receberão uma mensagem por SMS ou uma chamada de alerta. A sirene interior e o detetor de fumo têm também alarmes sonoros. A nível de segurança é possível escolher três perfis diferentes:

- Deteção desativada onde não são emitidos quaisquer tipos de alertas, com exceção do sensor de fumo;
- Deteção personalizada: podendo-se escolher quais os sensores que ficam ativos;
- Deteção ativada: no qual todos os sensores ficam ativos.

Também é possível consultar todos os alertas emitidos, para quem foram enviados e a data dos mesmos. No menu do sistema domótico pode automatizar os equipamentos para os desligar e ligar, consultar valores enviados pelos sensores e definir temporizações. Na secção dos estados dos equipamentos, é possível ver o estado de todos os equipamentos, incluindo bateria e cobertura de cada um. Nas definições, o utilizador pode definir os equipamentos que fazem parte do perfil personalizado, ativar e desativar a ação da sirene, instalar novos acessórios, configurar alertas, número de telemóvel e email, entre outras opções.

O Meo Smart Home tem autonomia para um período máximo de 48 horas através da bateria que está presente na Central Smart Home e, ainda, tráfego ilimitado de dados fixos e móveis necessários para o funcionamento e a utilização da solução (Pedro Pinto, 2017b).

### 2.8.2 GEWISS KNX

A GEWISS é uma empresa italiana fundada na década de 70 em Bergamo. Foi fundada com o objetivo de constante desenvolvimento de produtos eletrónicos e, em 2010, começaram o desenvolvimento de tecnologias domóticas para o controlo de casas automatizadas. Daí nasceu o Chorus, um sistema domótico que oferece soluções de ponta para gestão inteligente e controlo de casas e edifícios. Um sistema orientado à melhoria da vida quotidiana, com melhorias garantidas em segurança e conforto, além de uma maior eficiência energética.

Dentro deste sistema são apresentadas três soluções domóticas ilustradas na Figura 2.8, descritas de seguida:


**Figura 2.8** – Propostas Chorus Bus KNX, Chorus Bus KNX Easy, Wireless ZigBee da marca GEWISS.

- *WIRELESS DEVICES* ZigBee® - Um sistema de comando e controlo de dispositivos sem fios, baseado em tecnologia ZigBee, que oferece soluções inovadoras para a renovação de edifícios, minimizando as intervenções estruturais. O ZigBee é um protocolo de comunicação ao nível da rede (sobre a norma

IEEE 802.15.4), seguro e confiável, devido à troca de dados encriptados o que garante que todos os nós da rede tenham acesso aos dispositivos. Os dispositivos ZigBee podem ser geridos via uma aplicação, com smartphone e tablet, graças ao *Smart Gateway*.

- Chorus - Building Automation BUS KNX – o sistema BUS com protocolo KNX é a solução ideal para habitações e edifícios de grandes dimensões. Um sistema de controlo evoluído conforme a norma europeia EN50090 para a transmissão e a gestão dos dados de automação dos edifícios. O bus KNX é uma solução aberta oferecendo uma importante garantia de interoperabilidade entre os produtos de diferentes fabricantes, fiabilidade, flexibilidade e redução do tempo necessário para a instalação.

- Chorus – Domótica Bus KNX Easy - Sistema BUS com protocolo KNXEasy: um sistema de controlo evoluído conforme a norma Europeia EN50090, para as soluções residenciais, capaz de fazer dialogar com cada componente da instalação.

Estas soluções permitem o controlo de luzes, persianas, temperatura e energia. Permitem também criar o próprio ambiente residencial, integração de alarmes, stream de vídeo e controlo remoto através de telemóvel ou tablet. Para monitorização e controlo dos sistemas domóticos a GEWISS fornece também painéis tácteis e aplicações de software para dispositivos móveis. Apenas o primeiro ponto é que oferece uma solução recorrendo ao ZigBee, sendo as restantes baseadas no protocolo KNX, segundo uma configuração ETS (*S-mode*), ideal para ambientes residenciais e comerciais (Gewiss).

### 2.8.3 Hager Tebus KNX

A Hager, desde o início das suas soluções para domótica, escolheu a norma KNX como protocolo de comunicação a usar nos seus produtos para instalações inteligentes. As soluções KNX da Hager pretendem gerir um conjunto de funções elétricas

da instalação - iluminação, estores motorizados, controlo de temperatura, portões, rega, etc - de forma integrada e racional.

Os principais objetivos visados são a simplicidade na instalação, o conforto de utilização, a gestão eficiente da energia e uma adaptação máxima aos requisitos dos utilizadores. Estão disponíveis inúmeros produtos, para a conceção de soluções mais ou menos extensas, com comunicação por radiofrequência ou via cabos, com diferentes formas de configuração, de acordo com o resultado final pretendido.

A Hager oferece três soluções, sendo elas a *quicklink, easy link* e a *system link*, Figura 2.9.


**Figura 2.9** – Soluções Hager

A primeira, Tebis Quicklink, designa as soluções KNX da Hager baseadas em produtos KNX que comunicam via radiofrequência (868,3 MHz) e que podem ser programados por simples operação dos botões integrados nos próprios produtos. Desta forma é possível definir as funções a realizar e associar emissores aos recetores, de forma simples e rápida, não sendo necessário qualquer tipo de formação ou ferramenta para configuração. Devido à comunicação sem fios dos produtos, esta gama é especialmente interessante e adaptada a obras de renovação, ampliações ou mesmo novas instalações.

A Tebis Easy Link é o ressurgir da solução KNX mais popular da Hager pois disponibiliza a ferramenta de configuração KNX Easy permitindo uma configuração

única no mercado. Utilizando um pacote de software disponibilizados pela Hager, o instalador/integrador poderá usar um PC, tablet ou smartphone para configurar os produtos KNX da Hager. A comunicação entre o dispositivo de configuração e a instalação é feita através de Wi-Fi. Para esse efeito, é utilizado um Kit Configurador KNX Easy da Hager, o TXA100. A programação é bastante intuitiva, baseada em menus e de acordo com o standard KNX. A Hager disponibiliza sessões de formação periódicas relativas à programação via TXA100. A oferta de produtos é bastante vasta e é uma alternativa bastante competitiva face a uma instalação convencional. A gama é composta por produtos TP (*twisted pair*, de ligação ao cabo Bus KNX) e produtos RF , que podem coexistir. Os benefícios para os seus utilizadores são inúmeros, desde o conforto de utilização à poupança de energia. A nível de funcionalidades destacam-se o controlo remoto da instalação via iPhone/iPad/Tablets e smartphone Android, comandos baseados em informações meteorológicas, programação horária, deteção de incêndio, visualização e controlo via Touch panel, etc. Esta solução é a mais indicada para apartamentos, moradias e pequeno terciário.

Por fim temos o System Link destinado a instalações de maiores dimensões ou tecnologicamente mais evoluídas, onde a Hager dispõe de uma oferta de produtos específicos. A programação deverá ser realizada exclusivamente por ETS, um software multimarca disponibilizado pela organização KNX. Esta gama de produtos é caracterizada pela vasta lista de funcionalidades que disponibiliza: temporizações ao ligar e/ou ao desligar, contactos configuráveis, 64 cenários por canal, funções lógicas, contadores de horas de funcionamento e de manobras, comando manual no produto, bloqueio temporário ou permanente por canal, alarmes, etc. A programação por software ETS permite explorar as funcionalidades dos produtos ao máximo, possibilitando a realização de instalações de grandes dimensões. No entanto, o manuseamento do software de configuração deverá ser realizado por integradores com formação certificada KNX (Hager).

### 2.8.4 Sistema MyHome da Btcino

O sistema MyHome da Btcino, Figura 2.10, oferece soluções para promover a segurança, conforto, eficiência energética e comunicação em qualquer tipo de residência e setor terciário. As suas múltiplas aplicações permitem criar sistemas com alta eficiência energética que satisfazem a norma europeia EN 15232. Através de dispositivos eletrónicos programáveis e instalações modulares, o sistema MyHome possibilita a integração mais fácil e acessível no tradicional sistema elétrico dando também a possibilidade de no futuro aumentar a rede de dispositivos sem grandes alterações estruturais. O sistema utiliza cabos de par entrelaçado para ligação ao barramento e pode ser integrado em diferentes sistemas como é o caso do KNX e DALI. Para alem disso, através de interfaces e uso do protocolo TCP/IP MyHome é capaz de controlar o sistema de áudio NUVO, integrar dispositivos e sistemas de outras marcas e controlo e monitorização remota da casa através da rede telefónica, telemóvel ou ligação através da Internet.

Como referido, o sistema MyHome oferece uma simples integração dos dispositivos. Isto deve-se a duas razões principais – a primeira é a utilização do *Driver Manager F459* configurado via interface e que através do protocolo TCP/IP permite o controlo de sistemas de temperatura de algumas marcas. Todas as funções de aquecimento e arrefecimento são operadas por *touch screen* ou termostatos. A segunda razão deve-se ao uso do protocolo desenvolvido pela Bticino, o *Open Web Net* (um protocolo aberto para redes elétricas) e disponibilizado pela comunidade MyHome para a integração com modelos padronizados tais como o Konnex, BACNET, DALI e dispositivos TCP/IP.

Existem dois tipos de dispositivos no sistema sendo eles:

- Controladores, ligados apenas ao barramento;
- Atuadores, ligados ao barramento e à alimentação (230 VAC/50Hz).

Se for necessária a expansão de um sistema já existente, e para evitar obras, o


**Figura 2.10** – Ilustração do sistema MyHome da Btcino.

sistema pode ser expandido com o auxílio de interfaces *wireless*/radio e dispositivos controlados por radio (baseados em Zigbee) (Bticino, 2016).

### 2.8.5 Outras soluções

Existem disponíveis online estudos comparativos entre sistemas domóticos. Por exemplo, a ASecureLife (ASecureLife, 2018) elaborou um estudo onde enumerou alguns dos melhores sistemas domóticos em 2018. Entre esses sistemas existem opções profissionais, sistemas DIY (*Do It Yourself* – faz tu mesmo) e sistemas baseados na segurança. Aqui serão apresentados três desses sitemas, que, segundo o *site*, são os melhores do mercado, atualmente.

Em primeiro temos o Samsung SmartThings (Fig. 2.11), um distema domótico DIY, a um preço acessível e sem despesas de instalação, ativação ou monitorizações mensais.


**Figura 2.11** – Logótipo do sistema domótico SmartThings.

Inclui opções de automação para luzes, tomadas de energia e aquecimento, ventilação e ar condicionado (HVAC), acompanhado de sensores de fumo, fugas de água, portas, garagem e janelas. Sendo um sistema DIY torna-se fácil o utilizador instalar ou alterar alguma parte do sistema, uma vez que pode ser controlado e monitorizado por um smartphone. Disponibiliza três kits: um kit de iluminação e dois de automação residencial – para começar, ou pode-se construir o próprio sistema a gosto. O SmartThings Hub é o centro do sistema e tem a capacidade de se conectar a outros produtos de automação que usando as frequências Z-Wave, ZigBee ou Wi-Fi, o que a coloca num patamar acima de outros *hubs* que apenas usam uma frequência. Apesar de ser um sistema DIY, o utilizador pode acrescentar um sistema de segurança em tempo real com um acréscimo no valor mensal.

Em segundo lugar encontra-se o HomeSeer, que se destaca pela extensa compatibilidade entre produtos de automação residenciais e o seu controlador, segundo o estudo, a que se conecta a mais frequências – Z-Wave, Insteon, X10, UPB, etc - entre todas as empresas estudadas.


**Figura 2.12** – Logótipo do sistema domótico HomeSeer.

Tem também um canal dedicado ao IFTTT (*if this, then that*) que permite uma total customização do ambiente onde o utilizador está. Os controladores que vêm por

defeito são mais caros que outros *hubs* DIY, mas pode-se comprar apenas o software, instalando-o num computador de casa e torna-lo num *hub*. Suporta sistemas de iluminação, controlo de clima, fechaduras de portas, garagem, cameras, sistemas de segurança, áudio/vídeo, sensores de ambiente e sistemas para o exterior como aspersores e piscina. Mas visto que os principais produtos do HomeSeer, são os seus controladores e software, este sistema tem muitos poucos equipamentos de automação compatíveis (como sensores, termostatos, fechaduras, etc.) pelo que tem de ser adquiridos à parte.

Por fim, aparece o Nest que não é tecnicamente um sistema de automação residencial completo mas, fornece um componente principal para sistemas de automação residencial DIY.


**Figura 2.13** – Logótipo do sistema Nest.

Os produtos que o Nest inclui são um termostato de aprendizagem capaz de ser programável segundo o utilizador, uma combinação de detetor de fumo e monóxido de carbono e câmaras de segurança, tanto interior como exterior. A instalação destes produtos pode ser feita por profissionais, com um valor acrescido e podem ser controlados e notificar o usuário através das aplicações para iOS e android. Todos os dispositivos conectam-se entre si através de uma aplicação sem necessitar de programação adicional. A aplicação é também compatível com vários produtos, o que permite uma fácil expansão do sistema domótico. Por ser um sistema maioritariamente DIY a única opção de subscrição é a opção de armazenamento de vídeo que pode guardar imagens de 10 a 30 dias.

Outra solução existente no mercado é a Livingdam, uma empresa portuguesa fundada em 2002 e pertencente ao Grupo Azevedo's desde 2008. A empresa apresenta

um projeto na área da domótica, com o nome Home Automation, caracterizado pela integração, e respetivo controlo remoto, das diferentes funcionalidades existentes numa habitação, como é o caso da iluminação, climatização, segurança, audio e vídeo, com o objetivo de garantir conforto, segurança, entretenimento, poupança de energia, estética e design.

A integração oferece ao utilizador o controlo fácil e intuitivo, através de uma única aplicação, comando ou painel tátil, de todas as funcionalidades *home automation* e *home theater*, nomeadamente, iluminação, climatização, áudio, vídeo e segurança. Com isso, a Livingdam, pretende flexibilizar as funcionalidade nas habitações, num príncipio *user friendly*. Esta solução pretende também a promoção da eficiência energética nas habitações e edifícios, podendo proporcionar reduções até 30% do consumo de energia. (Livingdam)


**Figura 2.14** – Exemplo de uma interface da Livingdam.

### 2.8.6 Soluções *open-source*

Soluções comerciais como as analisadas anteriormente são, por norma, dispendiosas, muitas das vezes obriga a uma remodelação na própria habitação por falta de flexibilidade na sua instalação. Em termos de segurança não são os mais fiáveis, pois estão a ser instalados na privacidade de uma habitação dispositivos que controlam o quotidiano de cada um. Existe assim uma preocupação pela segurança e privacidade por parte dos utilizadores, que preferem ter uma solução de código aberto, sendo capazes de controlar e conceder permissão a usuários explicitamente autorizados (Baker, 2017). Para tal, entusiastas de movimentos DIY, começaram a desenvolver sistemas deste género, implementando, ferramentas como o Raspberry Pi para a criação de interfaces e mecanismos de automação residencial. Estas soluções são caraterizadas por um custo inferior em relação às vistas anteriormente, uma maior flexibilidade na construção da rede domótica, uma segurança e privacidade maior, pelos fatores já mencionados. Assim serão apresentadas algumas das plataformas *open-source* mais usadas.

1. **Home Assitant**: é uma das plataformas mais usadas nesta área, sendo facilmente implementada num dispositivo que consiga executar o Python 3, sendo mais usado através de um Rasperry Pi. Pode ser integrada em vários sistemas de *open-source*, bem como soluçoes comerciais, permitindo a ligação, por exemplo, ao IFTTT, informações meteorológicas ou ao dispositivo Amazon Echo, para o controlo de alguns aspetos de uma habitação. Algumas funcionalidades deste sistema encontram-se na Figura 2.15, retirada de uma demonstração disponível em `https://www.home-assistant.io/demo/`.

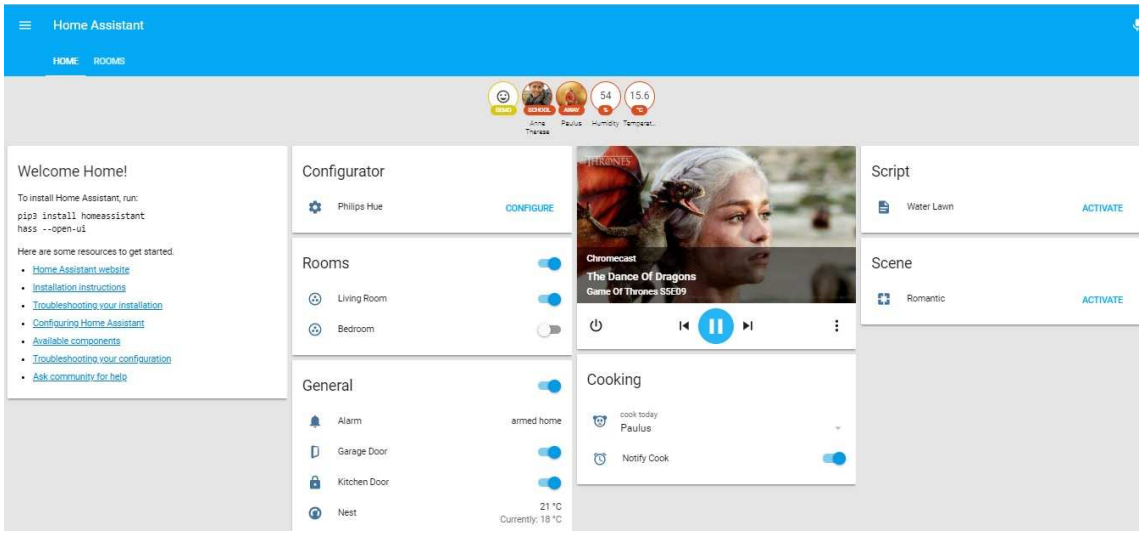


**Figura 2.15** – Demonstração da interface do sistema Home assistant (Assistant).

2. **Domoticz**: é um sistema de automação residencial com uma ampla biblioteca de dispositivos suportados, desde estações meteorológicas a detectores de fumo e controladores remotos, e um grande número de integrações adicionais de terceiros, tal como documentado no site do projeto (https://www.domoticz.com/wiki/Integrations_and_Protocols). A sua interface é escalável em HTML5, tornando-o acessível a partir de navegadores de computadores e *smartphones* mais modernos, é leve, sendo executado em vários dispositivos de baixo consumo energético, como o Raspberry Pi.


**Figura 2.16** – Exemplo da página Temperatura na interface Domoticz (Domoticz, 2015).

3. **openHab**: o OpenHAB (abreviação de *Open Home Automation Bus*) é uma das ferramentas de automação residencial mais conhecidas entre os entusiastas de dos sistemas de código aberto, com uma grande comunidade de utilizadores e um grande número de dispositivos e integrações suportados. Escrito em Java e utiliza o Apache Karaf juntamente com o Eclipse Equinox, agrupando tudo no servidor HTTP Jetty. O openHAB é portátil na maioria dos principais sistemas operativos, suportando assim centenas de dispositivos. O sistema é projetado para ser o mais fléxivel possível, tornando mais fácil para os *developers* adicionarem os seus próprios dispositivos ou *plugins* ao sistema. O OpenHAB também tem aplicações de interface nativas para iOS e Android para controlo de dispositivos, bem como ferramentas de *design* para a criação e desenvolvimento de uma interface única para cada utilizador. Nas Figuras 2.17 e 2.18 estão representadas duas interfaces, uma considerada básica e a segunda de uma aplicação num dispositivo com iOS.

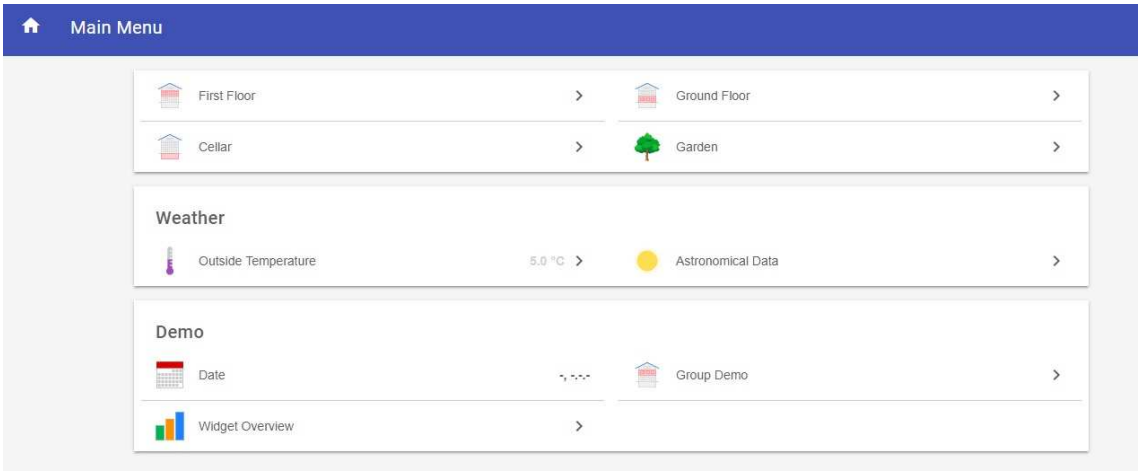


**Figura 2.17** – Modelo básico de interface com campos acessíveis da OpenHAB (OpenHAB).


**Figura 2.18** – Interface no sistema operativo iOS - OpenHAB (MySensors).

### 2.8.7 Resumo

Pelo estudo feito é possível afirmar que, aos sistemas domóticos, estão associados vários benefícios. Através de redes de sensores é possível recolher várias informações de uma casa, como temperatura, humidade ou luminosidade que permitem aumentar o conforto, a monitorização e segurança de uma habitação. No entanto, apesar destas vantagens, a nível comercial, estes sistemas ainda não se apresenta com um papel relevante. Isto deve-se ao facto de não existir uma norma padrão para este tipo de sistemas. Também a vasta opção de sistemas domóticos comerciais existentes, não tendo grandes compatibilidades entre si, não ajuda à expansão dos sistemas.

A maior parte das soluções comerciais disponíveis no mercado já integram sistemas de rede sem fios, facilitando a sua implementação, visto que não é necessário a instalação de nova cablagem nas habitações já construídas. Apesar disto, estas soluções são por norma dispendiosas. Para tal e, tendo em conta as soluções estudadas, conclui-se que a melhor solução, seria um sistema sem fios de baixo custo. Desta forma, é apresentada uma proposta de um sistema domótico no Capítulo 3.

# 3 Proposta de um sistema domótico

Neste capítulo será apresentada a estrutura de uma proposta do sistema domótico, os casos de uso referentes a todos os aspetos do sistema e a comunicação entre os dispositivos usados na rede proposta. Também serão apresentadas algumas especificações sobre os mesmos.

## 3.1 Arquitetura do sistema

O sistema implementado está representado na Figura 3.1. Nela podemos encontrar o Raspberry Pi como um dos principais nós do sistema, pois é ele que permite a comunicação entre todos os dispositivos. Neste caso funciona como um servidor (*broker*) e também difunde uma rede. Esta rede é designada de "APRPi" e é capaz de se ligar a Internet através de um cabo ethernet.

É nesta rede que os dispositivos I/O serão capazes de se ligar, onde lhes será atribuído um endereço IP dentro da gama 172.24.1.50 e 172.24.1.150. Esses mesmos dispositivos deverão ter a capacidade de se ligar automaticamente na rede e ao *broker*, recolher dados a partir dos sensores periodicamente e alterar o estado dos

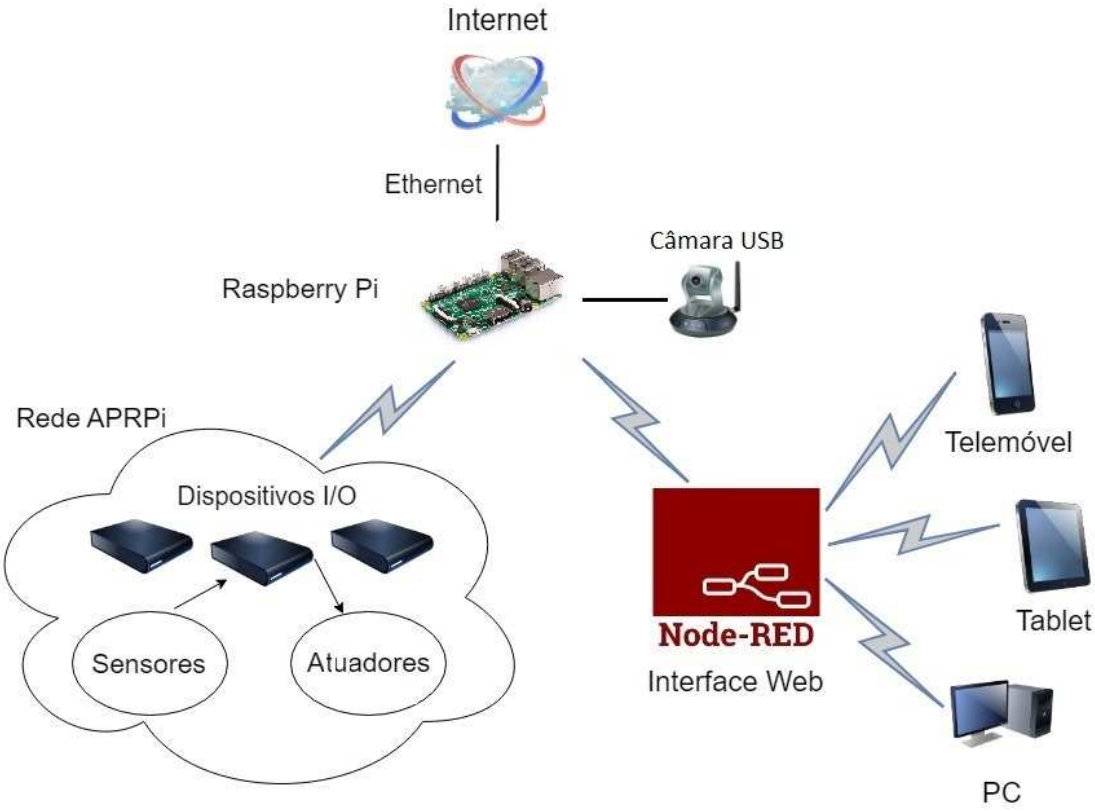


**Figura 3.1** – Arquitetura do sistema proposto.

atuadores através da interação com a interface. Sendo assim possível ao utilizador executar funções como ligar/desligar luzes, controlar portas, eletrodomésticos, portões ou mesmo acionar temporizadores. É-lhe também permitida a consulta dos dados recolhidos pelos sensores de temperatura, humidade e luminosidade, bem como a notificação de alarmes técnicos, como incêndios, inundações ou intrusões.

Tudo isto acessível a partir da interface web, criada através da aplicação Node-Red, que pode ser acedida através do endereço 172.24.1.1:1880/ui, num telemóvel ou PC que esteja conectado a rede.

Também no Raspberry Pi estará ligada uma câmara para poder obter imagens das divisões da casa ou para a eventualidade de surgir algum alarme.

## 3.2 Criação da rede de dispositivos E/S

Para suportar a comunicação entre todos os elementos do sistema, criou-se uma rede através do Raspberry Pi que funciona como um ponto de acesso. As trocas de mensagens são feitas através do MQTT, na porta 1883, e através de pedidos HTTP estabelecidos na porta 8080.

Assim, nesta secção, serão descritos os comandos para a configuração do Raspberry Pi, desde a ligação à internet através do cabo Ethernet até à difusão dessa ligação. Para tal, é necessário:

- Ativar um ponto Wi-Fi;
- Atribuir endereços IP dinâmicos aos dispositivos que se liguem na rede;
- Associar as redes Wi-Fi e Ethernet.

Para a atribuição de endereços IP, instalaram-se no Raspberry Pi, dois pacotes através do seguinte comando:

```
sudo apt-get install dnsmasq hostapd
```

Onde o primeiro, o `Dnsmasq`, é um servidor combinado de DHCP e DNS adequado para este tipo de redes mais pequenas (Kelley, 2018). O segundo - HostAPD - permite o uso do Wi-Fi embutido no Raspberry Pi tornando-o num ponto de acesso para a conexão de outros dispositivos, possibilitando também a configuração desse mesmo ponto de acesso, como adicionar uma palavra-passe. Após esta instalação, a primeira coisa a fazer é configuração da interface `wlan0` com um IP estático. Para tal, é necessário fazer com que essa interface seja ignorada para posteriormente a poder editar. Em versões mais recentes do Raspbian, a configuração de interfaces é feita pelo `dhcpd`, sendo necessário abrir o ficheiro de configuração `dhcpcd.conf`, através do comando:

```
sudo nano /etc/dhcpcd.conf
```

Acrescentando no final do ficheiro - `denyinterfaces wlan0` - e de seguida alterar essa interface da seguinte forma:

```
allow-hotplug wlan0
iface wlan0 inet static
address 172.24.1.1
netmask 255.255.255.0
network 172.24.1.0
broadcast 172.24.1.255
#wpa-conf /etc/wpa_supplicant/wpa_supplicant.conf
```

Neste ponto é necessário o reinício do serviço `dhcpd` e recarregar a configuração para a interface `wlan0` com:

```
sudo ifdown wlan0;
sudo ifup wlan0
```

Depois de configurado o dhcpd, é necessário configurar o HostAPD. Para tal abre-se o ficheiro de configuração através do comando `sudo nano /etc/hostapd/hostapd.conf`

e é aqui que irá ser definido o nome e palavra passe da rede. Tem-se também que guardar esta configuração para que seja iniciada ao ligar novamente o Raspberry Pi, primeiro abrindo o ficheiro `/etc/default/hostapd`, encontrar a linha `#DAEMON_CONF=` e substituir por `DAEMON_CONF="/etc/hostapd/hostapd.conf"`.

Nesta altura a rede criada já aparece, mas ainda não atribui endereços IP, problema que é resolvido configurando o Dnsmasq, localizado em `/etc/dnsmasq.conf`. Configuração do ficheiro:

```
interface=wlan0
listen-address=172.24.1.1
bind-interfaces
server=8.8.8.8
domain-needed
bogus-priv
dhcp-range=172.24.1.50,172.24.1.150,12h
```

Esta configuração permite a atribuição de IP na gama de 172.24.1.50 até ao 172.24.1.150, utilizando como servidores DNS os servidores da Google: 8.8.8.8.

Por fim para garantir que o tráfego da rede alcance a Internet é necessário habilitar o encaminhamento de pacotes e para tal abre-se o ficheiro `/etc/sysctl.conf`, removendo o comentário na linha `net.ipv4.ip_forward=1`. Para ativar esta configuração imediatamente usa-se o seguinte comando:

```
sudo sh -c "echo 1 >/proc/sys/net/ipv4/ip_forward"
```

Também é necessário partilhar a conexão à internet do Raspberry Pi, configurando o serviço NAT (*Network Address Translation*) entre a interface `eth0` associado à ligação Ethernet e a interface `wlan07`. Isto é feito através dos seguintes comandos:

```
sudo iptables -t nat -A POSTROUTING -o eth0 -j MASQUERADE
sudo iptables -A FORWARD -i eth0 -o wlan0 -m
state --state RELATED,ESTABLISHED -j ACCEPT
sudo iptables -A FORWARD -i wlan0 -o eth0 -j ACCEPT
```

Este conjunto de regras necessitarão de ser ativadas sempre que o Raspberry seja reiniciado, portanto em primeiro lugar corre-se o comando - `sudo sh -c "iptables-save > /etc/iptables.ipv4.nat"` – e em segundo lugar é necessário abrir o ficheiro `/etc/rc.local`, adicionado por cima da linha `exit 0`, o seguinte:

```
iptables-restore < /etc/iptables.ipv4.nat
```

Para concluir, resta apenas iniciar os serviços instalados:

```
sudo service hostapd start
sudo service dnsmasq start
```

Fica assim o sistema habilitado à ligação à Internet, apesar da dependência física do cabo Ethernet.

## 3.3 Comunicação entre dispositivos

A comunicação entre dispositivos é feita sobre o protocolo TCP/IP ao nível da camada física e é suportada por Wi-Fi, o que facilita a integração/remoção de dispositivos da rede. Através da criação da rede, abordada no subcapítulo anterior, cada um desses dispositivos I/O serão identificados pelo seu IP, atribuído aquando da conexão à rede criada.

Para tal, foi necessária a escolha de um dispositivo que possibilitasse essas funções. Foi então escolhido o ESP8266 devido ao seu baixo custo (entre 3 a 5 euros) e pela realização de trabalhos anteriores com este módulo. Apresenta também uma boa capacidade de memória, compatibilidade com o Arduino IDE e uma antena Wi-Fi integrada. Para além destas, o *chip*, desenvolvido pela empresa chinesa Espressif, apresenta caraterísticas mais gerais, como pinos para entradas e saída (16 GPIO), capacidade de conversão analógica-digital de 10 bits, interface periférica serial (*Serial Peripheral Interface* - SPI). Possui ainda uma ROM de inicializão de 64 kB, RAM de instruções de 64 kB e RAM de dados de 96 kB (Espressif, 2018).

A troca de mensagens, entre dispositivos e servidor (Raspberry Pi) são maioritariamente feitos através do protocolo MQTT, falado em capítulos anteriores, e através também de pedidos HTTP(GET) por parte dos dispositivos ao servidor. Nesses pedidos são enviados alguns parâmetros para os dispositivos, como o estado de cada um – Ativo/Desativo - e o seu valor atual como "ON/OFF".

Os parâmetros por sua vez são codificados em JSON (*JavaScript Object Notation*) que se carateriza por ser um formato leve de troca de dados, fácil de leitura humana e fácil para análise e gestão por parte das máquinas. O JSON são assim pares constituídos por atributo/valor(es), como mostra o seguinte exemplo:

- "objecto":"valor"

O modelo de comunicação encontra-se simplificado na figura 3.2 em baixo:

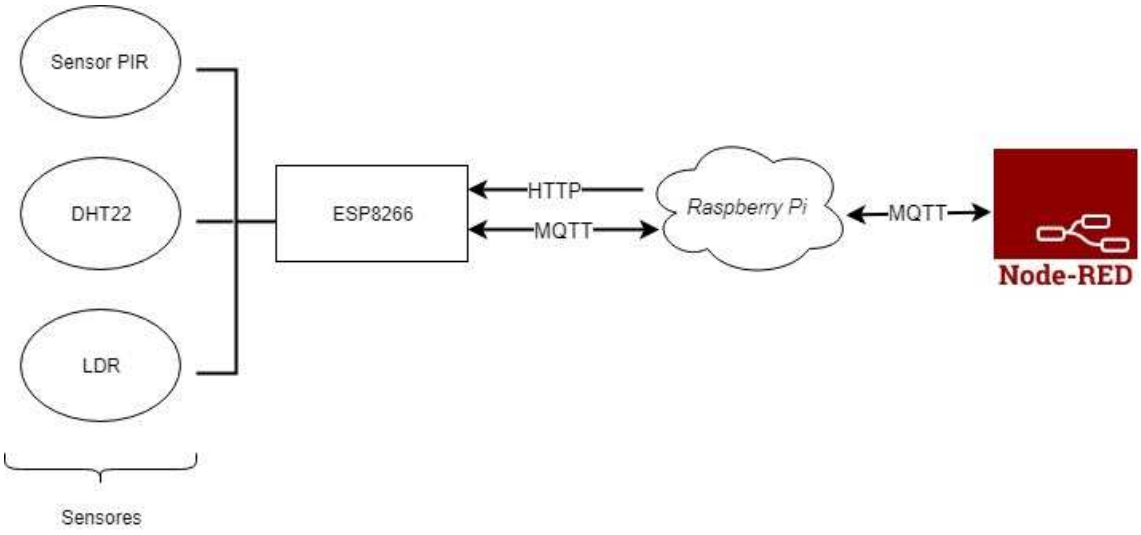


**Figura 3.2** – Modelo de comunicação simplificado utilizado para a criação do sistema domótico.

## 3.4 Casos de uso

Parte das funcionalidades do sistema encontram-se na Figura 3.3 onde estão apresentadas de uma forma ilustrativa e de melhor compreensão as interações que acontecem

no sistema.

Na figura é possível distinguir os dois atores do sistema, sendo eles representados pelo tempo e pelo utilizador. O utilizador, como se pode verificar, interage diretamente com a interface sendo capaz de mudar os estados dos atuadores e ter acesso à informação dos sensores. Outro ator é o tempo que tem a capacidade de mudar também os estados dos atuadores através da introdução, por parte do utilizador, de datas e horas especificas para a realização de tarefas. A criação deste diagrama teve por base os requisitos do sistema listados no anexo A.

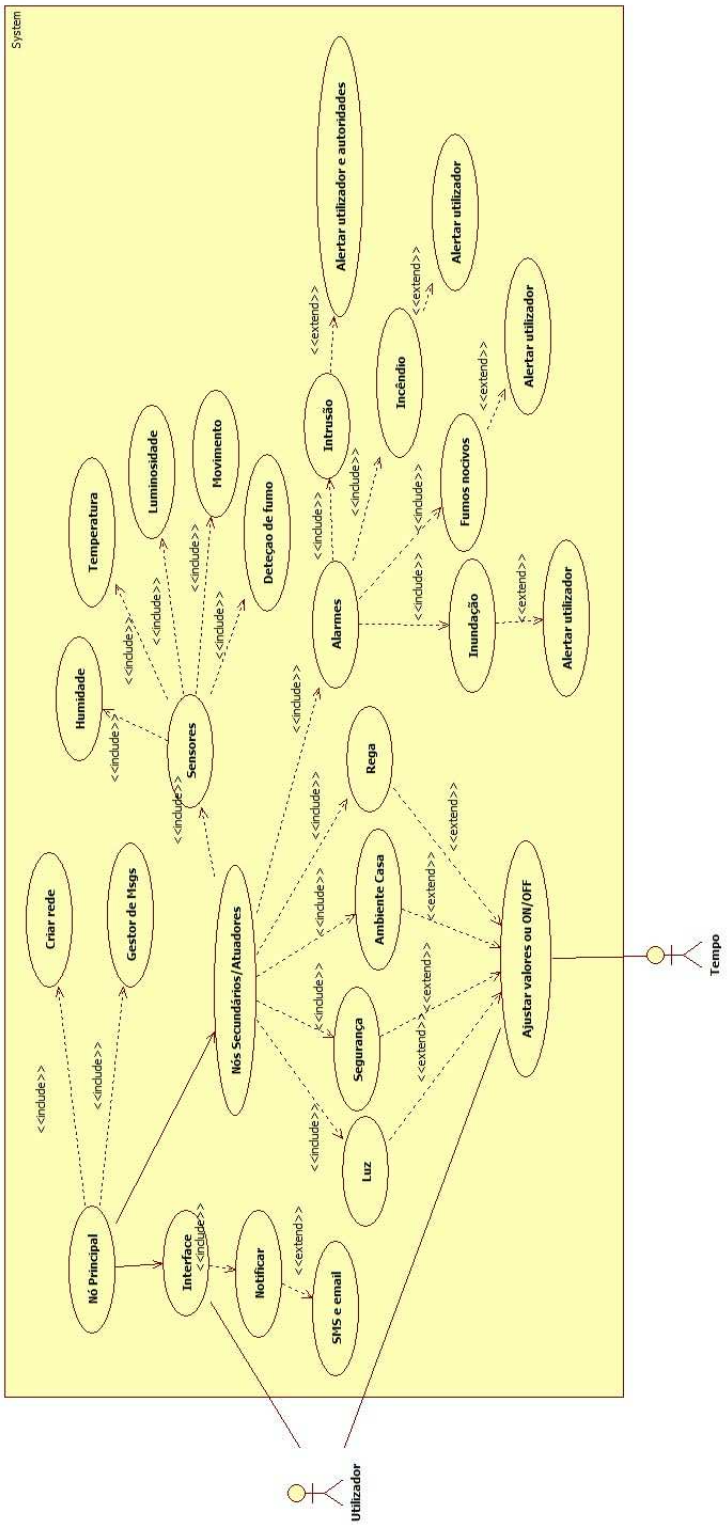


**Figura 3.3** – Casos de uso do sistema.

# 4 Implementação do protótipo

Neste capítulo serão apresentadas as ferramentas utilizadas para a implementação do sistema domótico cuja arquitetura foi descrita no capítulo anterior. Dessas ferramentas consta o Raspberry Pi 3, o microcontrolador ESP8266 e o Node-Red, ferramenta usada para a concepçao da interface. Dentro de cada um destes tópicos serão abordados com um maior foco as configurações, comandos e algoritmos, associados a cada elemento.

## 4.1 Raspberry Pi

O Raspberry Pi 3 é um *Single Board Computer* (SBC) produzido pela *Raspberry Pi Foundation*, de modo a divulgar e educar as pessoas na era da computação. É caracterizado pelo baixo custo e alto desempenho e é devido a essas caraterísticas que permitiu à fundação o desenvolvimento e construção de modelos cada vez mais recentes, mais potentes e melhorados.

As especificações do modelo usado encontram-se enumeradas de seguida:

- Quad Core 1.2GHz Broadcom BCM2837 64bit CPU;

- 1GB RAM;
- BCM43438 wireless LAN and Bluetooth Low Energy (BLE) on board;
- 100 Base Ethernet;
- 40 pinos GPIO;
- 4 portas USB 2.0;
- Saida de áudio analógica e digital;
- Porta HDMI;
- Porta CSI para conectar uma camara Raspberry;
- Porta DSI para conectar *touchscreen*;
- Porta para cartões MicroSD;
- Fonte de energia Micro USB.

Neste trabalho, o modelo do Raspberry Pi usado foi o Raspberry Pi 3 Modelo B sendo ele uma das principais unidades de processamento dos dados. Para além de funcionar como um ponto de acesso Wi-Fi para os dispositivos se ligarem ao sistema, é também um servidor MQTT, ou seja, o *broker*, responsável pelo encaminhamento das mensagens entre dispositivos e a interface, também ela criada numa aplicação nativa do Raspberry Pi, o Node-Red. Para instalar um sistema operativo no Raspberry Pi há várias escolhas, mas a opção tomada passou pelo sistema operativo Raspbian recomendado pela *Raspberry Pi Foundation*. A instalação do sistema operativo passa por descarregar do site oficial do produto uma imagem que deve ser gravada num cartão MicroSD com a ajuda da ferramenta gráfica Etcher, para facilitar esse processo e que pode ser descarregado também no site oficial, e posteriormente inserir o cartão na porta correspondente. Nesta altura o Raspberry estará em condições de começar a ser utilizado (Pi).

### 4.1.1 Servidor MQTT – *Broker*

Este tópico, como já foi abordado em capítulos anteriores, é um protocolo de mensagens extremamente simples e leve com um mecanismo de publicação/subscrição, desenhado para dispositivos de parcos recursos e redes de baixa largura de banda, alta latência ou não confiáveis. Apesar disto, tenta garantir a confiabilidade e entrega das mensagens com diferentes níveis de QoS. É, portanto, ideal para comunicação máquina-a-máquina ou para aplicações em tecnologias IoT, onde a largura de banda e a bateria dos dispositivos são fundamentais.

Como servidor MQTT, foi utilizado o Mosquitto. Para o instalar no Raspberry Pi, é necessário primeiramente importar a chave do repositório com:

```
wget http://repo.mosquitto.org/debian/mosquitto-repo.gpg.key
sudo apt-key add mosquitto-repo.gpg.key
```

Depois, tem-se que tornar o repositório disponível através dos seguintes comandos:

```
cd /etc/apt/sources.list.d/
sudo wget http://repo.mosquitto.org/debian/mosquitto-jessie.list
```

A versão do Raspbian é a Jessie, sendo, portanto, o nome que consta no comando em cima.

Nesta fase, atualizam-se os pacotes com `sudo apt-get update` e instala-se o servidor `mosquitto` e respetivos clientes, respetivamente, com os seguintes comandos:

```
sudo apt-get install mosquitto
sudo apt-get install mosquitto-clients
```

Para verificar que tudo está operacional basta executar o comando `mosquitto` no qual deverá ter uma resposta deste género:

```
1516727717: mosquitto version 1.4.14 ( build date Mon , 10 Jul 2017
23:48:43 +0100) starting
1516727717: Using default config .
1516727717: Opening ipv4 listen socket on port 1883.
```

Para iniciar em *background* este servidor deve-se executar o comando:

```
sudo /etc/init.d/mosquitto start
```

Com este serviço a funcionar, pode-se enviar comandos na própria consola do Raspberry do tipo:

```
mosquitto_sub -h HOST  -p 1883 -u " USERNAME " -P "
PASS " -t " teste1 " - Para subscrever ao tópico.

mosquitto_pub -h HOST  -p 1883 -u " USERNAME " -P "
PASS " -t " teste " -m "mensagem " - Para publicar num tópico.
```

Outros parâmetros podem ser adicionados a estes comandos, como referido em capítulos anteriores, mas estes campos são os essenciais para se iniciar a comunicação entre pelo menos dois dispositivos.

### 4.1.2 Node.js

O Node.js é um interpretador JavaScript de código aberto, orientado a eventos assíncronos e projetado para criar aplicativos de rede escalonáveis, como um servidor web (Node.js), tal como usado neste trabalho. Neste servidor irá ter a informação sobre o sistema da forma de um objeto JSON, que necessitará de ser executado a cada reínicio do Raspberry Pi.

Para o instalar, basta atualizar o administrador de pacotes do Raspberry Pi, visto já em cima e prosseguir para o download e instalação do Node.js através do comando:

```
curl -sL https://deb.nodesource.com/setup_8.x | sudo -E bash -
```

e para instalar executar o comando:

```
sudo apt-get install -y nodejs
```

Desta forma é então possível criar um servidor HTTP, no qual o conteúdo será do tipo *application/json* e que contém o objeto JSON descrito no anexo B. O servidor estará à escuta no endereço `http://172.24.1.1:8080`.

## 4.2 Microcontrolador

Neste trabalho, cada dispositivos de E/S é baseado no microcontrolador o ESP8266 NodeMCU v1.0, ilustrado na figura 4.1. O NodeMCU é uma plataforma de código aberto e inclui *firmware* que é executado no módulo ESP8266, baseado no módulo ESP-12E que juntamente com um conjunto de componentes e caraterísticas, como a memoria flash de 4MB e uma interface USB para possibilitar a comunicação série, facilita o desenvolvimento de aplicações (NodeMcu). Esse firmware usa uma linguagem de programação Lua, que permite que toda a programação relevante seja feita usando o Arduino IDE, ou seja, em C/C++, tornando-se assim a principal plataforma de desenvolvimento para aplicações nos vários módulos ESP8266. Com isto, muitas das bibliotecas e funções usadas no Arduino podem ser utilizadas neste dispositivo, aumentando as possibilidades no desenvolvimento de aplicações.

As funções principais do microcontrolador é o de controlar os atuadores ligados aos pinos GPIO, através de comandos enviados através da interface, ter a capacidade de se ligar à rede, ler e processar os valores retirados dos sensores, executar pedidos HTTP via Wi-Fi para processamento de dados em formato JSON.

**Figura 4.1** – Imagem do NodeMCU (Banggood).

### 4.2.1 Entradas e saídas

Na Figura 4.2 podem ser analisados os pinos da placa NodeMCU ESP8266, na qual se verifica que nem todos os pinos são iguais, possuindo funções especificas e que nem todas foram usadas neste projeto.

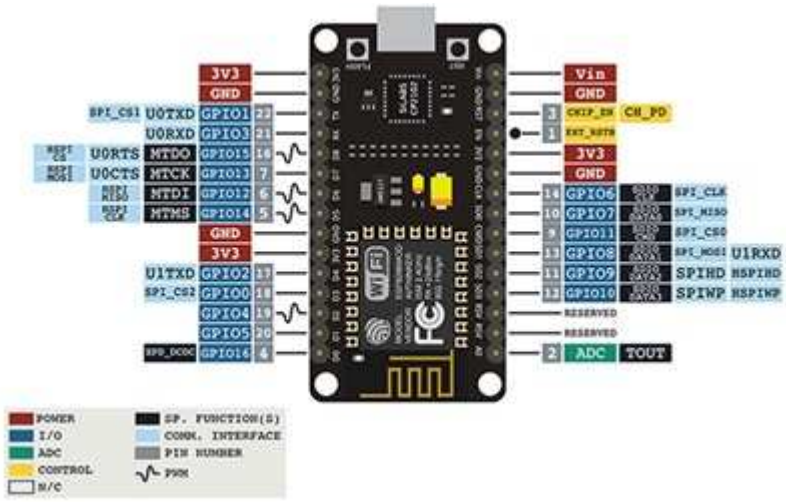


**Figura 4.2** – Diagrama de pinos da placa NodeMCU ESP8266 (da Eletrônica, 2018).

De forma a construir uma plataforma de testes, foram usadas duas placas NodeMCU ESP8266, que passarão a ser designadas de ESP8266_1 e ESP8266_2, às quais foram ligados aos pinos os sensores e os atuadores (neste caso apenas LED's). A lista seguinte resume os sensores/atuadores ligados a cada dispositivo.

Dispositivo ESP8266_1:

- A0 – Ligado ao sensor analógico de luminosidade LDR;
- GPIO0 – Atuador 2 ligado ao sensor analógico de luminosidade LDR
- GPIO4 – Atuador 1;
- GPIO5 – Sensor DHT (Temperatura e humidade relativa).

Dispositivo ESP8266_2:

- GPIO5 – Sensor PIR;
- GPIO13 – Alarme técnico (pino que suporta interrupções).

São nestes pinos onde o sistema vai receber comandos, despoletar ações, alarmes e leituras de sensores. No caso do GPIO13, este pino suporta interrupções, ou seja, ações que permitem executar certas tarefas importantes em segundo plano e que são ativadas por um evento externo. (Arduino, b)

### 4.2.2 Sensores

Neste projeto foram utilizados três sensores, descritos de seguida.

- **DHT22 - Sensor de temperatura e humidade**

  Este sensor é usado tanto para a deteção de temperatura como de humidade. Consiste em quatro pinos e usa um sensor capacitivo para medir a humidade

relativa do ar e através dele gerar um sinal no pino de dados. O primeiro pino corresponde a alimentação VDD (3 a 5V), o segundo é referente ao pino de dados, o terceiro fica sem ligação e o último liga-se à terra. Segundo a folha de dados os valores de humidade e temperatura variam, respetivamente, entre os 0% e os 100% com uma precisão entre ±2%, enquanto que a temperatura pode variar entre -40ºC e 80ºC, com uma precisão de $\pm 0,5$ºC.

**Figura 4.3** – Exemplo de um DHT22 (Nerokas).

- **LDR - *Light Dependent Resistor***

  LDR, é um dispositivo cuja resistividade varia de acordo com a incidência eletromagnética, ou seja, é sensível à luz. São constituídos por duas células, construídas por materiais semicondutores com elevada resistência e que apresentam respostas espetrais semelhante ao que acontece com o olho humano. Quando a luz incide sobre a resistência, devido às propriedades fotocondutivas do material a resistência baixa. De acordo com a folha de dados a resistência da célula varia entre os 9 kΩ (unidade de intensidade luminosa para 10 lux) a 400 Ω (1000 lux), o que faz variar a sua resistência entre 9 kΩ a 400 Ω, respetivamente. As suas aplicações variam desde o controlo automático de luzes, sistemas de alarme contra roubo, entre outras.

**Figura 4.4** – Resistência dependente de luminosidade (BUYECOMPONENTS).

- **Sensor PIR para deteção de movimento**

  É um sensor eletrónico que mede a luz infravermelha que irradia dos objetos no seu campo de visão. Através dos potenciómetros embutidos no sensor é possível ajustar o *delay* do sensor entre três a cinco minutos e a sensibilidade do alcance até sete metros. O seu *output* varia entre dois estados sendo eles, *High* e *Low*. As suas aplicações são maioritariamente a deteção de movimentos, de forma a automatizar luzes de presença, por exemplo.

**Figura 4.5** – Exemplo de um sensor PIR do tipo HC-SR501 (GorillaBuilderz).

### 4.2.3 Desenvolvimento de *software*

O software foi implementado nos dispositivos, via USB, utilizando o IDE do Arduino, tirando assim partido das várias bibliotecas *open-source* já existentes que são compatíveis com o módulo ESP8266 e podem ser executadas diretamente no próprio. Das bibliotecas usadas, de realçar a biblioteca `ArduinoJson`, para o tratamento de objetos JSON e a `PubSubClient` que fornece ferramentas para a comunicação MQTT. Para um melhor entendimento do desenvolvimento do *software* é apresentado na Figura 4.6 um fluxograma do mesmo.

O primeiro passo passa pela definição das variáveis, correspondentes aos sensores e atuadores listados anteriormente. Depois segue-se da função `setup_wifi()`, onde se procede a ligação à rede local Wi-Fi, configurando o nome da rede (SSID) e a password. É nesta função onde é atribuído o endereço IP, dentro da gama definida quando a rede foi criada. Esta função é uma das funções pré-definidas das bibliotecas.

Após se verificar o sucesso na ligação à rede é usada a função `callback()` e é nesta função onde são executados os comandos quando chega alguma mensagem de publicação a um tópico por parte da interface. Caso o tópico corresponda a uma subscrição, o dispositivo atua. Nesta fase é iniciada também uma ligação HTTP ao link `http://172.24.1.1:8080`, onde são executados pedidos GET à página, de modo a recolher as informações necessárias à atuação do atuador, neste caso será um "on" ou um "off". Aqui, também com o auxílio da ferramenta *ArduinoJson Assistant* disponível em https://arduinojson.org/ e com a biblioteca `ArduinoJson` é possível, fazer o *parse*, ou seja, partir o objeto JSON, de modo a obter o valor pretendido mais facilmente. De seguida, é apresentado um exemplo de como se processa essa operação:

```
const char* nos0_canal0_valorAtual1_ligado =
nos0_canal0["valorAtual"][1]["ligado"];// "on"
```

Neste caso, é extraído do campo o valor "on", que permite a ligação do atuador.

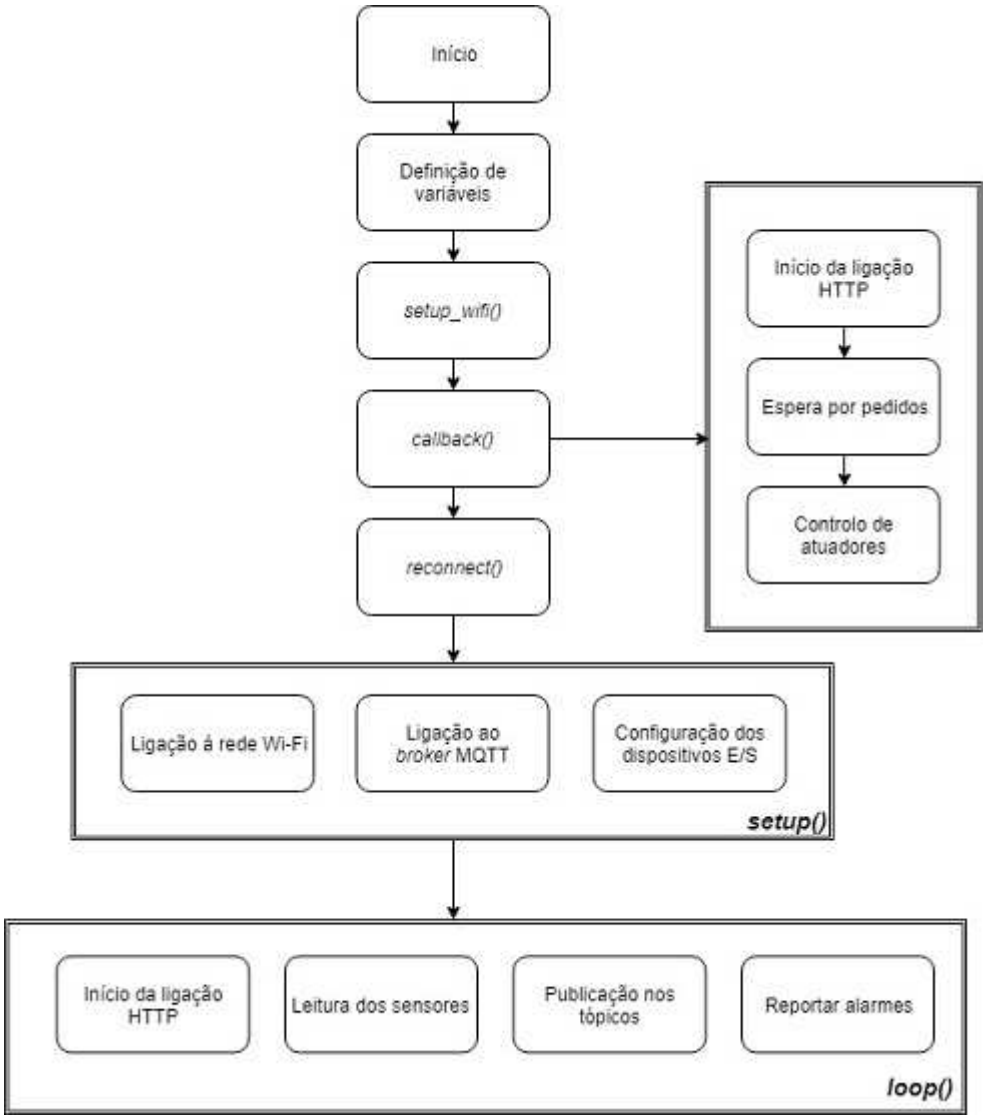


**Figura 4.6** – Fluxograma do *software implementado.*

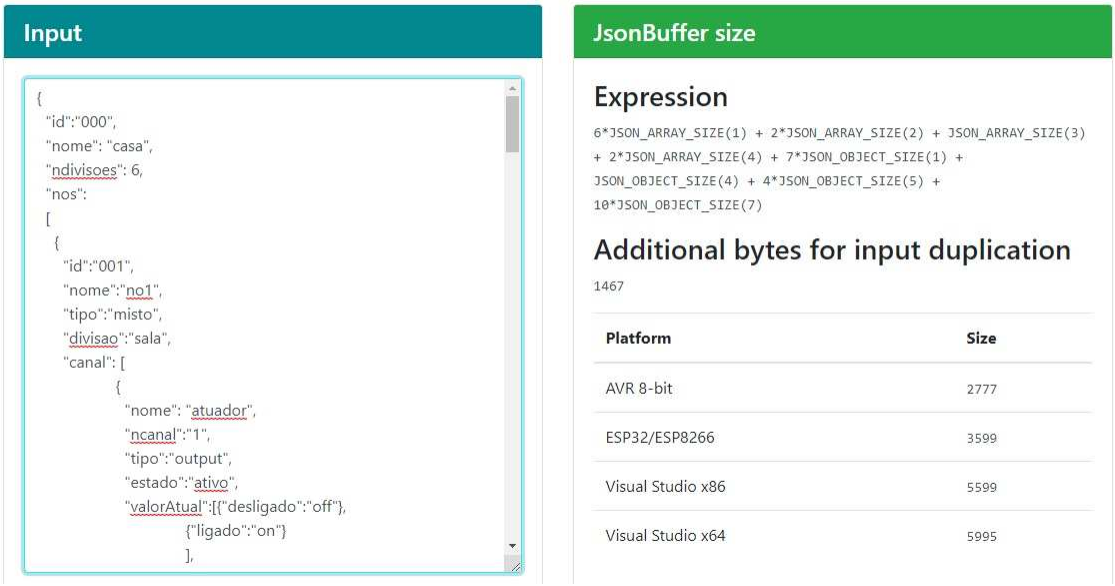


**Figura 4.7** – Assistente de configuração JSON.

Esta ferramenta é também bastante útil no cálculo do tamanho do *buffer* a alocar o objeto JSON, como mostra a Figura 4.7. Apesar de receber mensagens em tópicos é só na função `reconnect()` onde são feitas as subscrições aos tópicos e onde é feita as tentativas de ligação ao *broker* MQTT.

Após estas funções, é executada a função `setup()`, onde é iniciada a porta série com um *baud rate* de 115200, os pinos onde os sensores e atuadores estão conectados e respetivas caraterísticas (INPUT ou OUTPUT) e se iniciam as ligações à rede e ao servidor MQTT.

Por fim, no função `loop()`, são feitas as leituras dos sensores e consequente análise e publicação nos tópicos correspondentes. Nesta função também é aberta mais uma vez um pedido HTTP do tipo GET ao *localhost* na porta 8080, para receção das variáveis necessárias à publicação dos valores de temperatura, humidade e do sensor de luminosidade, como mostra o comando seguinte:

```
const char* nos0_canal1_valorAtual0_temperatura =
nos0_canal1["valorAtual"][0]["temperatura"]; // "String(t)"
```

No caso do sensor DHT, são feitas as leituras de trinta em trinta segundos, recorrendo a variáveis auxiliares de tempo, e é nesse tempo que são feitas as publicações nos tópicos. No caso da temperatura, é usada uma função da biblioteca para este tipo de sensor, que calcula os valores lidos pelo sensor, em graus Celsius. Para facilitar a leitura e publicação dos valores, é executada uma função (`dtostrf`) onde, os mesmos, são armazenados num *buffer*. Também a partir da leitura do sensor e analisando os valores de temperatura e humidade, são impostas regras, de modo a publicar alertas no alarmes técnicos do sistema. Para tal, no caso do valor de temperatura passe um valor superior a 60ºC, é publicado no tópico referente aos alarmes de incêndio, sendo o utilizador notificado através da interface. No caso da humidade é publicado nos alarmes técnicos, se o valor da variável for superior a 80%. Estes valores foram atribuídos, tido em conta a folha de dados do componente em questão, mas também meramente para demonstração do funcionamento dos tópicos configurados.

No caso do sensor LDR, são estabelecidos parâmetros para que o LED possa ser ativo ou desativo consoante essas especificações e consoante a incidência luminosa. A leitura do sensor é feita através do pino analógico A0, onde são retirados valores entre 0 e 1023, que variam de acordo com a intensidade luminosa. A gama de valores definida para atuação do LED, associado ao LDR, foi feita com base em testes de modo a obter os valores que mais se adequam a dada intensidade luminosa. Os seus valores são publicados, juntamente com os valores de temperatura e humidade, para a interface a cada trinta segundos. Por fim, a ligação HTTP é fechada.

Para o alarme técnico referente às intrusões foi feito um algoritmo recorrendo a interrupções. Para este teste foi simulado um circuito fechado, onde quando a ligação é interrompida gera um alerta através da publicação ao tópico correspondente. Esta ideia teve como base circuitos que se instalam à volta de uma janela ou porta. Para tal, é necessário implementar uma função que possa ser ativada por uma interrupção - função executada fora do código principal - onde, quando o estado do pino altera é recebida essa informação e posteriormente publicado no tópico com a mensagem "Alarme – Intrusão". A função é definida da seguinte forma:

```
attachInterrupt(digitalPinToInterrupt(pin), ISR, mode);
```

Onde o primeiro argumento é o pino sobre o qual ocorrerá a interrupção. O segundo é a função a chamar quando a interrupção ocorre; esta função não deve ter parâmetros e não deve devolver nada. Esta função é por vezes referida como uma rotina de serviço de interrupção. O terceiro argumento que define quando a interrupção deve ser acionada, neste caso, será `CHANGE` para acionar a interrupção sempre que o pino alterar o valor (Arduino, a).

No sensor PIR o código utilizado é bastante simples, basta recolher a leitura do pino onde o sensor está ligado, através do `digitalRead(sensor)`, caso esteja num estado `HIGH`, significa que existiu movimento e assim publica essa mensagem. No caso do estado ser LOW, nada reporta à interface, apenas à porta série, para efeitos de teste.

Importante referir que os valores das publicações, segundo o criador da biblioteca usada para a comunicação MQTT, devem ser do tipo *string* (PubSubCLient).

## 4.3 Tópicos do sistema

Para o desenho do sistema foram necessários definir alguns tópicos nos quais os microcontroladores e os nós da interface se poderão subscrever ou publicar mensagens, de acordo com a função de cada um. Todos os alarmes começam com o indicativo "casa" que é a topo do tópico, sendo aprofundados nos campos à frente separados pela barra. De seguida serão apresentados os tópicos específicos, separando os tópicos referentes a subscrições e os tópicos referentes a publicações.

**Subscrições**:

- casa/sala/atuador.

**Publicações**:

- casa/geral/consumos/eletricidade;
- casa/geral/consumos/caudalAgua;
- casa/sala/temperatura;
- casa/sala/humidade;
- casa/sala/ldr;
- casa/geral/alarmes/intrusão;
- casa/geral/alarmes/humidade;
- casa/geral/alarmes/pir.

Apesar de existirem dois atuadores no sistema, apenas um está conectado ao MQTT e só esse, pode receber comandos via MQTT. O outro atuador, o LED ligador ao LDR, atua de acordo com as leituras feitas a partir do sensor e é, como já visto na capítulo referente ao *software* implementado, controlado através de regras que o ligam ou desligam se esses valores passarem certos níveis.

## 4.4 Interface com o `Node-RED`

O Node-Red é uma ferramenta de programação baseada em fluxos, desenvolvida inicialmente pela *IBM Emerging Technologies* que agora faz parte da *JS Foundation*. Este tipo de programação foi inventado por J. Paul Morrison em 1970 e é uma forma de descrever uma aplicação através de caixas negras ou nós como são chamados no `Node-RED`. Cada nó tem uma função bem definida, onde a informação que chega a esses nós é processada de acordo com a sua função e passada, se for o caso, para o próximo nó. A rede é responsável pelo fluxo dos dados entre os nós. O aspeto e a criação das redes de nós são de fácil acessibilidade para uma a maior parte dos utilizadores, não sendo necessário entender na perfeição o código por dentro do nó.

Para criar e editar os fluxos, basta abrir num navegador web e colocar o endereço onde se encontra a correr o `Node-RED`, neste caso, será no `http://localhost:1880`, e basta arrastar os nós para o espaço de trabalho, conectá-los e fazer *deploy*. O conjunto de nós padrão pode ser atualizado e expandido facilmente, nós esses criados pela comunidade do `Node-RED`, permitindo assim a dinamização dos fluxos. Para instalar os nós fundamentais para a elaboração de uma *dashboard* é preciso instalar a partir do Raspberry ou no próprio endereço indicado em cima, o `node-red-dashboard`, através do comando `npm i node-red-dashboard` e caso exista alguma dependência por falta do `npm`, instalar esse pacote com `sudo apt-get install nodejs npm`. Ou no link, no canto superior direito, clicar sobre as três barras horizontais e instalar através do `Manage pallete` os nós pretendidos. Os fluxos criados podem ser guardados ou partilhados de forma fácil e rápida através de ficheiros JSON.

Inicialmente a sua função era permitir a visualização e manipulação de tópicos MQTT, portanto uma ferramenta ideal para desenvolver este projeto e elaborar a interface (Foundation). O Node-Red já vem nativo no Raspbian e facilmente se arranca o serviço através do comando `node-red-start` e para o desligar executar o comando `node-red-stop`. Neste caso é necessário que seja iniciado a cada *reboot* do Raspberry Pi e para tal é executado o seguinte comando - `sudo systemctl enable nodered.service` – assim sempre que se iniciar o sistema, a interface estará prontamente disponível para a aceder no endereço `http://localhost:1880/ui`. Para garantir a segurança dos fluxos é preciso instalar uma ferramenta de administrador, através do comando `sudo npm install -g node-red-admin` no diretório `cd /.node-red`. De seguida é necessário gerar uma hash para a password, com o comando `node-red-admin hash-pw`. Copia-se a *hash* e adiciona-se no ficheiro `/.node-red/settings.js` da seguinte forma:

```
adminAuth: {
type: "credentials",
users: [{
username: "Nome do utilizador",
password: "Hash gerada",
```

```
permissions: "*"
}]
},
```

Estas linhas encontram-se comentadas, portanto é preciso remover de comentário e fazer o mesmo com a seguinte linha:

```
httpNodeAuth: {user:"YOUR USERNAME",pass:"YOUR HASH HERE"}
```

Deve-se reiniciar o serviço `Node-RED` e tem-se assim um acréscimo de segurança ao sistema. A página de autenticação encontra-se representada na Figura 4.8.

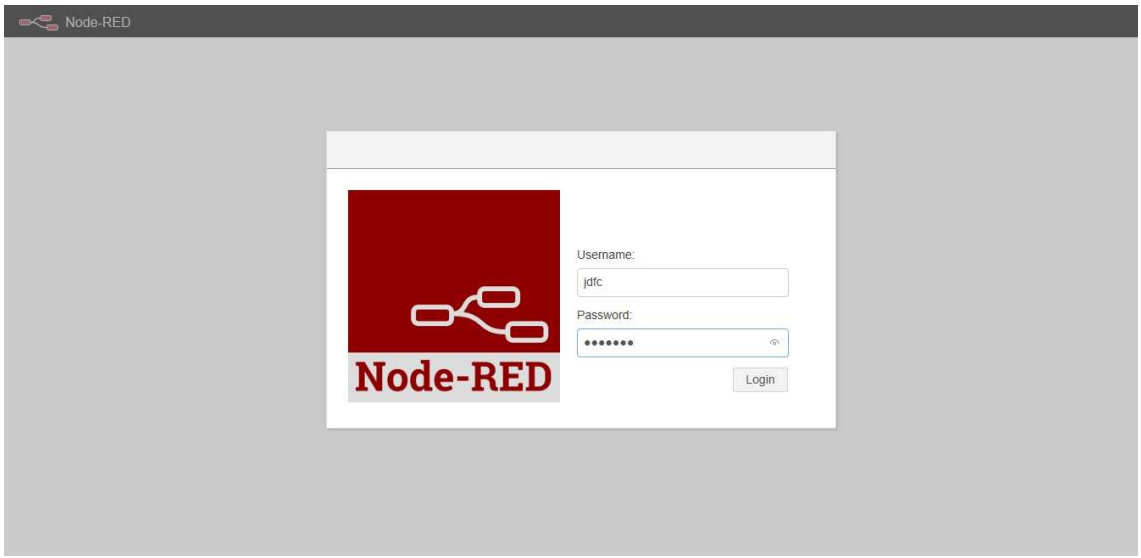


**Figura 4.8** – Página de autenticação para a edição da *dashboard*.

Depois, basta clicar nos nós pretendidos e fazer as conexões necessárias entre eles para criar um fluxos. De seguida, é apresentado um desses fluxos criado no `Node-RED`:

O principal objetivo na criação da interface é ser *user-friendly*, isto é, uma interface intuitiva, simples e de fácil compreensão por parte do utilizador, com as leituras dos sensores bem evidentes e manuseio dos atuadores com facilidade, tornando assim a sua experiência agradável. Esta interface pode ser acedida através de um *browser*,

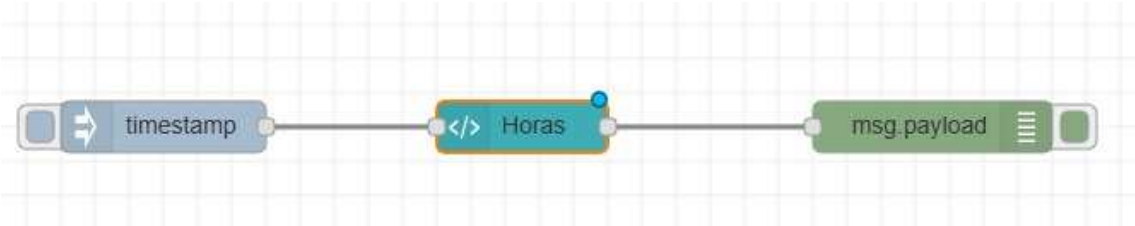


**Figura 4.9** – Exemplo de um fluxo criado no `Node-Red`.

introduzindo o IP do *host*, porta e indicativo para a interface, que neste caso traduz-se no seguinte endereço: `172.24.1.1:1880/ui`. Então basta aceder a esse link, seja a partir de um computador, dispositivo móvel ou ate mesmo uma *smartTV*.

Para demonstrar os aspetos mencionados em cima da interface serão apresentados de seguida as páginas que a constituem, como é feita a navegação entre as páginas e como se processa a interação entre o utilizador e o sistema.

### 4.4.1 Barra de navegação

Para navegar entre as diferentes páginas da interface, no topo superior esquerdo, existe três barras horizontais, na qual basta clicar para mostrar as páginas às quais podemos aceder. Pretende-se que este ícone se encontre sempre visível em qualquer parte da interface, apesar do *scroll* da página, permitindo uma maior facilidade e comodidade na transição entre as paginas. A barra de navegação, neste trabalho, é constituída por cinco elementos, sendo eles:

- **HOME**: que permite visualizar uma mensagem de saudação ao utilizador, horas e uma estação meteorológica simplificada.
- **Geral**: onde pode ser visualizado aspetos gerais da casa, como os consumos energéticos e de água.
- **Sala**: página que simula uma divisão da casa, onde podem ser vistos os valores obtidos por alguns sensores;

- **Quarto**: nesta página pode ser visualizado valores obtidos por sensores e ativar/desativar o atuador;
- **Alarmes**: interface onde é possível receber notificações de alarmes e recolher imagens.

A seguinte figura mostra essa barra:


**Figura 4.10** – Barra de navegação.

### 4.4.2 Página Home

É a pagina apresentada na abertura da interface, ao introduzir o IP para o efeito. Esta página é meramente informativa, dando ao utilizador algumas informações importantes, como as horas, o estado de tempo e uma mensagem de saudação. O estado do tempo é atualizado constantemente, a partir de um pedido ao site OpenWeatherMap (OpenWeather), através de uma chave API, dando informação referente à

cidade de Braga. Para o efeito foi necessária a instalação de um conjunto de nós específicos para o OpenWeatherMap. Esta informação vem num formato JSON, onde no Node-Red através do nó `function` e através de um algoritmo, implementado em JavaScript, é capaz de fazer *parse* aos campos JSON, recolhendo a informação desejada. Com essa informação, dependendo do estado do tempo, é possível gerar imagens apelativas e ilustrativas das condições meteorológicas, exibindo texto com a descrição das condições meteorológicas e da temperatura atual.

O relógio, neste caso, é composto pelo nó `template`, no qual é possível escrever programas em linguagem HTML, exibindo assim o diminutivo do dia, diminutivo do mês, dia, ano e horas:minutos.

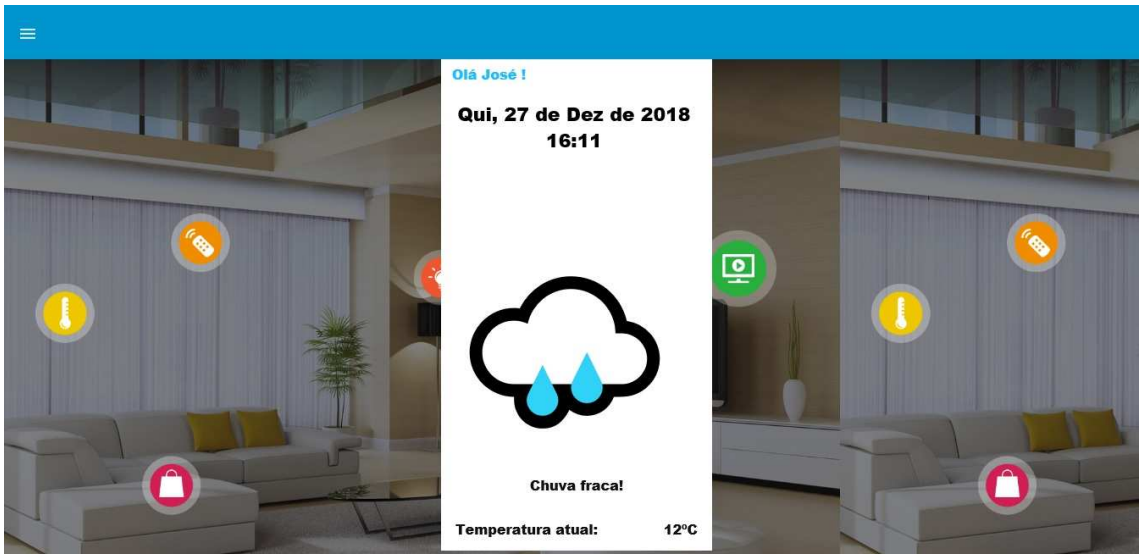


**Figura 4.11** – Página Home.

### 4.4.3 Página geral

Nesta pagina são apresentados valores dos consumos de energia e água. Visto que é um aspeto algo relevante para uma grande maioria de pessoas, esta interface permite a visualização desses valores de uma forma chamativa, apresentando os mesmos em medidores do tipo de um manómetro com ajuda e do nó `gauge`.

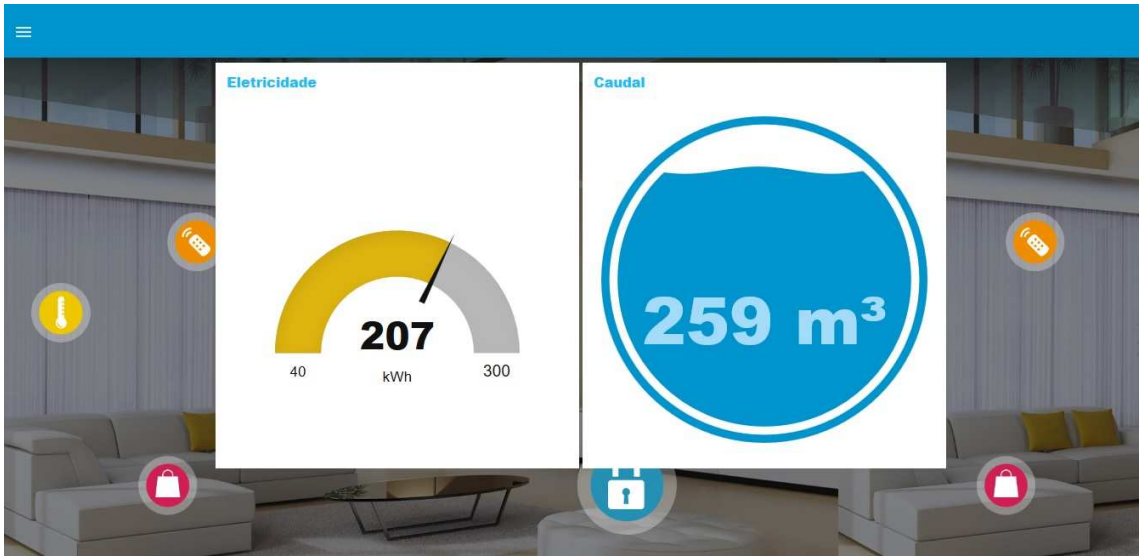


**Figura 4.12** – Consumos apresentados na página Geral.

Apesar do sistema estar preparado para receber esses valores, os apresentados na Figura 4.12, são gerados aleatoriamente através de uma função gerada no ESP8266, que publica num tópico esses mesmos valores no nó `mqtt in` que corresponde a uma subscrição e passa esses valores para um nó do tipo `gauge`, para uma visualização mais apelativa.

### 4.4.4 Página Sala

Nesta página, que simula uma divisão da casa, são apresentados valores recolhidos pelo sensor DHT22. Como explicado num subcapítulo anterior, esses valores são publicados periodicamente num tópico. Para receber esses valores, é usado novamente o nó `mqtt in`, que recebe os dados, e para a sua visualização recorreu-se aos nós `chart` (gráfico) e `gauge`.

No caso dos valores da temperatura, para uma melhor observação dos dados foi feito uma média dos valores obtidos através de dois nós `smooth`, que regista a informação dos últimos valores obtidos, faz uma media dos mesmos e passa essa informação, neste caso para um para o gráfico, outro para o `gauge`. Para o gráfico são analisados

dez dos últimos valores, feita a média e transmitida então para o nó `chart`, estes valores permitem uma melhor interpretação do mesmo, tornando-o mais claro e fácil de leitura. O nó `chart` apresenta no eixo dos Y, a temperatura numa gama dos zeros graus Celsius até aos quarenta e cinco graus Celsius. No eixo do X, tem-se o tempo, no qual podem ser definidos intervalos na ordem dos segundos, minutos, horas, dias e semanas. Neste caso está definido para apresentar valores das últimas oito horas. Para o `gauge`, apenas são usados dois dos últimos valores, para ter uma leitura quase instantânea dos valores de temperatura que é apresentada.

No caso dos valores de humidade também se utilizou um nó `smooth`, mas apenas um que faz a média entre os últimos dois valores obtidos e transmitindo essa informação para os nós `gauge` e `chart`. No nó `chart` é apresentado no eixo dos Y a percentagem de humidade, de zero a cem porcento, e no eixo dos X as horas, referente às últimas oito, como a temperatura.

Passando com o indicador por cima das linhas dos gráficos, é possível ver a temperatura ou humidade que fazia a determinada hora.

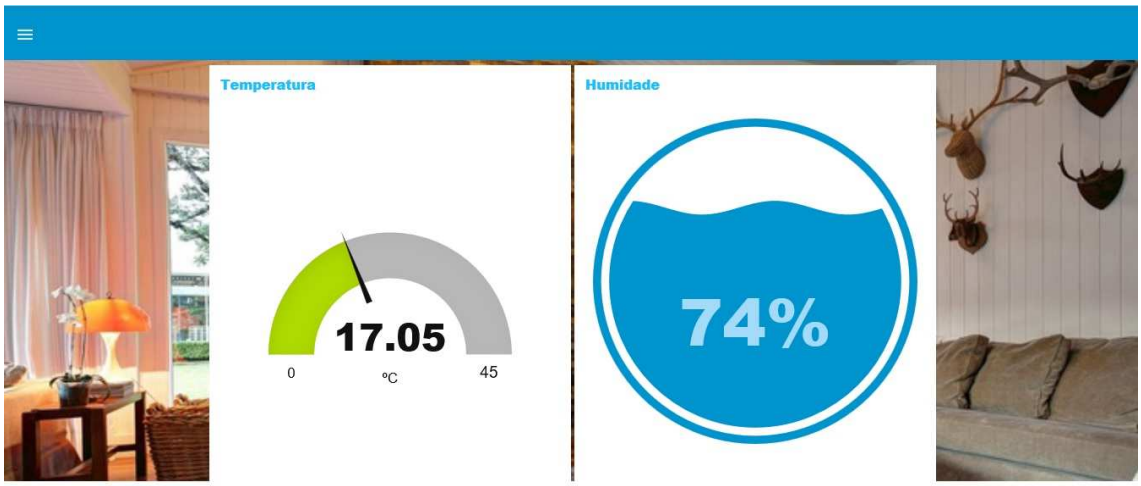


**Figura 4.13** – Gauges referentes à temperatura e humidade.

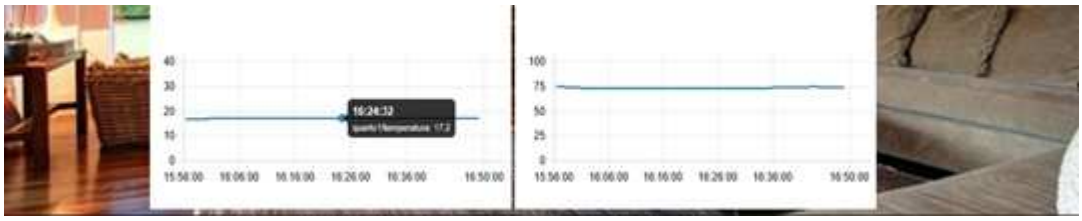

**Figura 4.14** – Gráficos com os valores de temperatura e humidade.

### 4.4.5 Página Quarto

Mais uma vez esta página representa uma divisão da casa. Nela podemos encontrar, como mostra a Figura 4.15, dados recolhidos do sensor de luminosidade e o estado no qual se encontra o LED a ele associado. É possível também ver um bloco com as ações a realizar sobre o atuador.

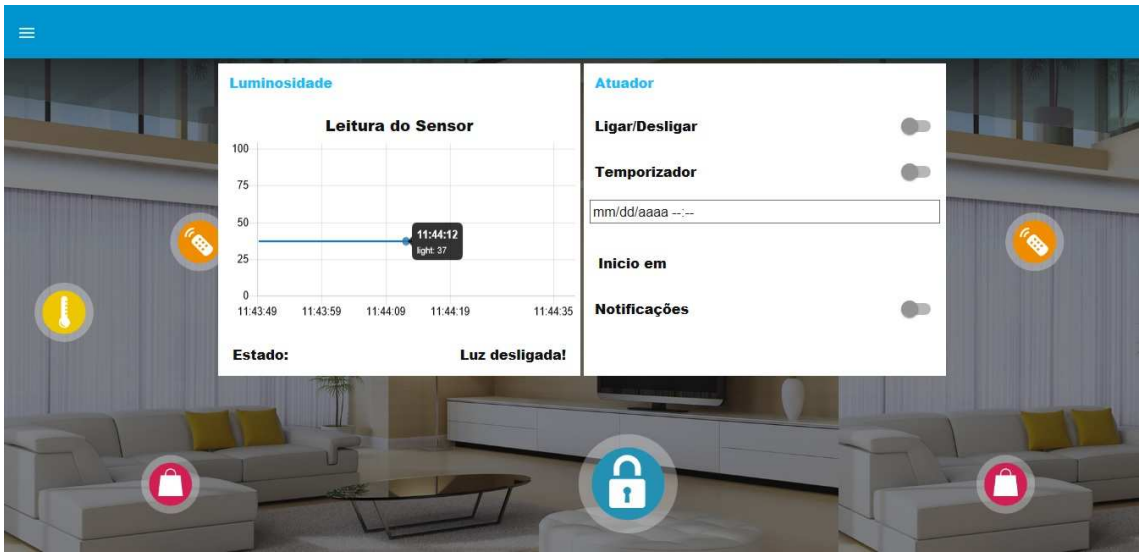


**Figura 4.15** – Página da divisão designada Quarto.

Para o grupo da Luminosidade, é usado o nó `mqtt in` para a receção dos dados, o nó `chart` para a visualização dos dados e o nó texto que recebe também informação do nó `mqtt` se a luz se encontra ligada ou desligada, de acordo com o algoritmo implementado no ESP8266.

O bloco Atuador, é então responsável pela interação direta com o sistema. Neste

caso apenas foram realizados testes com LED's, mas que permitiram comprovar a possibilidade de controlo de dispositivos através da interface web. Para interagir com o sistema são apresentadas três soluções:

- Ligar/desligar;
- Temporizador, com um *input* para data e hora e respetiva contagem decrescente;
- Notificações.

A primeira é um envio direto de uma mensagem, por parte do utilizador, que é composta por um *on* para Ligar ou por *off* para Desligar. Essas definições encontram-se no JSON, no servidor HTTP, onde o dispositivo I/O faz um GET dessas informações e na função `callback()`, definida anteriormente, é capaz de realmente atuar sobre o atuador, neste caso, o LED. No Node-Red é usado um nó `mqtt out` que tem a função de publicar mensagens em tópicos. Neste caso publicará no tópico "casa/quarto/atuador", onde o ESP8266 se encontra à escuta de pedidos. Para enviar esses pedidos é usado um nó `switch` que permite enviar no *payload* da mensagem *strings, booleans*, números, formatos JSON, *buffers* e *timestamp's*. Neste caso em particular o atuador está à espera de mensagens do tipo *string – on* ou *off*.

Depois, através do nó `template` foi criado uma barra na qual podemos definir uma data e hora (Figura 4.16) para ativar o atuador. Através de funções criadas nos nós `function`, é possível visualizar uma contagem decrescente que se encontra em segundos. Após chegar ao zero, é enviado, noutro nó `function` essa informação para o nó `mqtt out` que faz ligar o LED. Essa mesma informação é enviada, também para uma função que faz com que o utilizador através de um botão, opte por receber notificação ou não, dessa informação. Caso o botão das notificações esteja ligado, é-nos dada a informação que o temporizador chegou ao fim, exibindo no canto superior direito uma notificação durante cinco segundos.

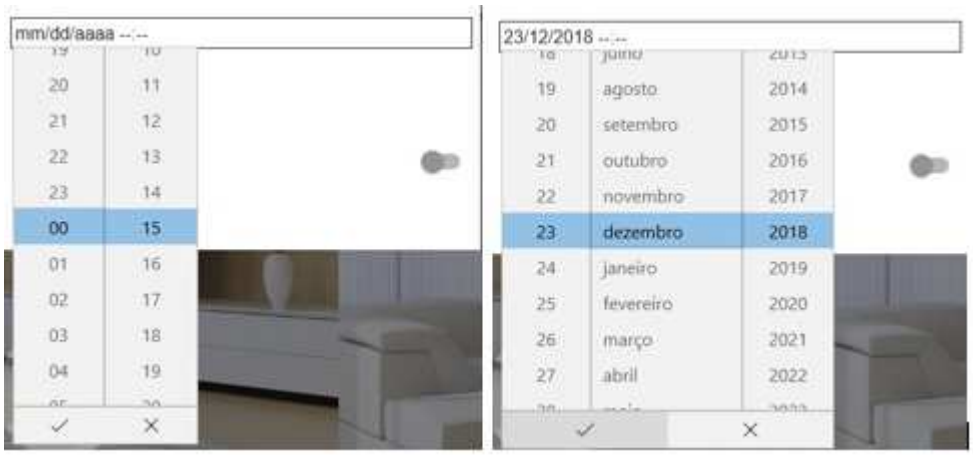


**Figura 4.16** – Definição da data e hora no temporizador.

### 4.4.6 Página Alarmes

Nesta página podem ser consultados todos os possíveis alarmes que o sistema pode gerar. Sendo eles:

- Intrusão;
- Humidade;
- Temperatura;
- Movimento;

A página é dividida então em cinco blocos, como mostra a Figura 4.17, onde podem ser visualizados os campos dos alarmes. Cada alarme, no Node-Red, é representado por um nó `mqtt in`, onde são publicados os alertas detetados nos ESP8266, de acordo com as funções vistas anteriormente.

Para complementar esta página, é possível observar um grupo, de nome "Imagem"no qual é possível a captura de imagens em tempo real, exibindo-a no mesmo bloco. Essa fotografia é tirada através de uma camara USB acoplada no Raspberry Pi,

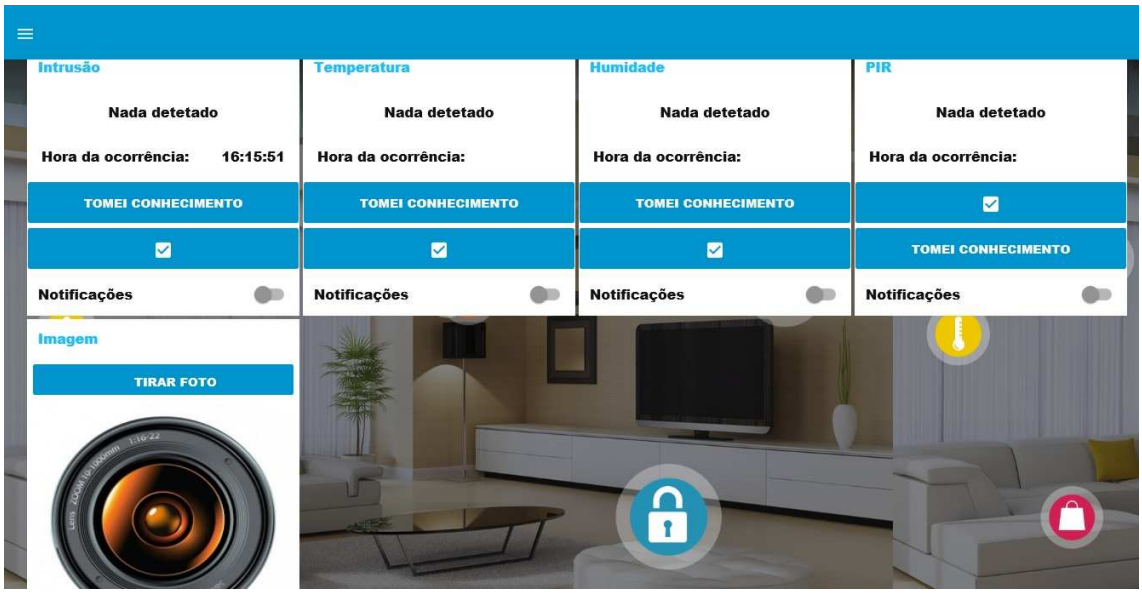


**Figura 4.17** – Página dos Alarmes.

no qual é ativada através do nó `exec` do Node-Red, no qual é possível executar o comando - `fswebcam/home/pi/Pictures/imagerec_.jpg`, que tira uma fotografia com o diretório e o nome tal como indicado dentro do comando.

Também neste bloco existe um botão, que pressionando envia essa imagem, ou a última guardada no Raspberry Pi, para a plataforma MySense. Na sua implementação, esse botão está conectado a outro nó `exec`, que envia o comando para executar o código em PHP, para o envio de imagens. A Figura 4.18 ilustra esse bloco e uma captura de imagem:

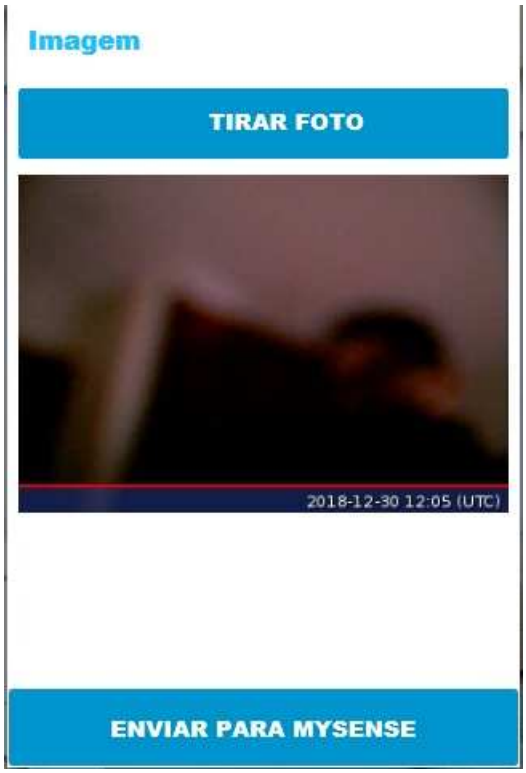


**Figura 4.18** – Exemplo do bloco imagem com a captura de uma imagem.

Em cada grupo é possível também visualizar, aquando da receção de um alerta, informações relevantes quanto ao alarme, por exemplo, no caso, da temperatura e humidade, receber os valores correspondentes, ou simplesmente a notificar que é detetado um alarme nos outros casos. Essa notificação aparece no canto superior direito, durante um período de cinco segundos e surge sobreposta a qualquer página da interface web que o utilizador se encontre, de modo a ficar bem visível. Essas notificações podem ser desligadas em todos os alarmes, utilizando o botão "Notificações"para o efeito. É também possível, receber mensagens áudio através do nó `audio out`, as informações que chegam aos nós `mqtt`, para uma maior brevidade no alerta ao utilizador e consequente ação sobre o alarme. Na figura 4.19 é possível verificar a mensagem que surge no campo de texto, a hora em que ocorreu o alarme e o sinal sonoro no separador web.

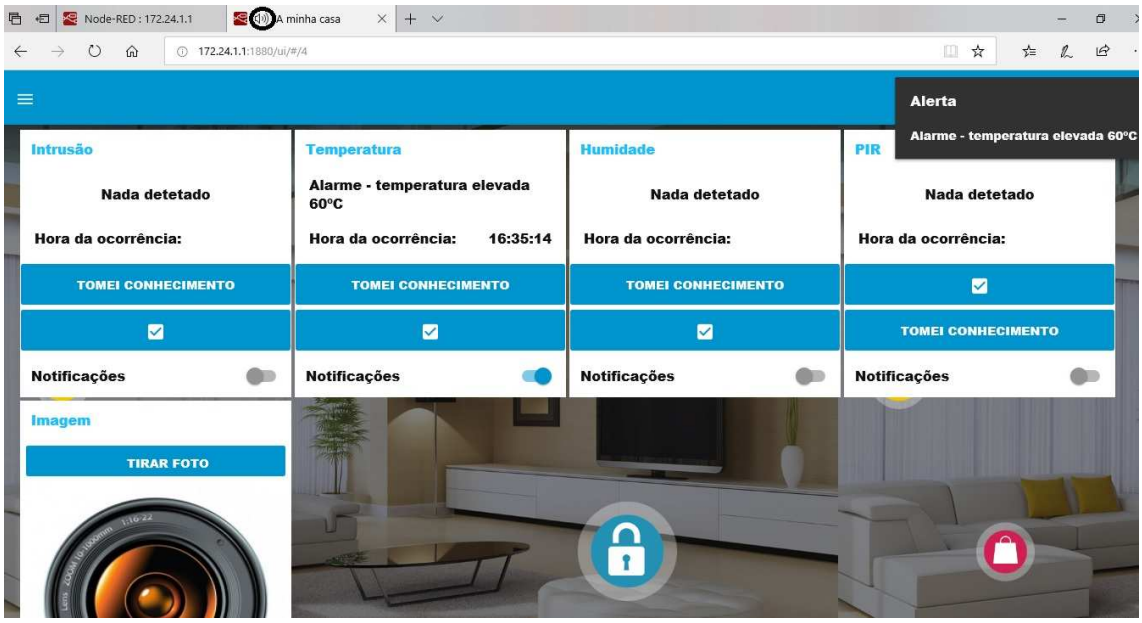


**Figura 4.19** – Notificações de alarmes.

Cada grupo, contém também dois botões que têm a função de limpar o conteúdo gerado pela deteção de algum dos alarmes, neste caso, o texto e a hora da ocorrência.

## 4.5 Aplicação com a plataforma mySense ioT++

Foi feito ainda uma aplicação de forma a enviar as imagens capturadas pela interface, na secção da Imagem, para a plataforma mySense ioT++. Esta plataforma representa um ambiente de agregação e integração de dados, desenvolvida na UTAD e que pretende dar suporte a vários projectos académicos. Dados georeferenciados e imagens podem ser armazenados e disparar ações em tempo real usando protocolos M2M e, ao mesmo tempo, alimentar sistemas de apoio à decisão. Cada dispositivo de aquisição de dados colocado em funcionamento em campo é visto como um objeto mySense, contendo canais (transdutores) de entrada (sensores) e saída (atuadores). Se um dispositivo recebe dados de outros, pode ser definido como um concentrador (gateway). Um canal é a unidade mínima com identificador único na plataforma mySense. Os dados gerados por um transdutor podem ser públicos, ser propriedade

privada ou de um grupo específico de utilizadores. Para facilitar, existe uma livraria de sensores prontos a serem usados (Morais, 2016).


**Figura 4.20** – Logótipo da plataforma mySense.

Com isto, foi então necessário a criação de um objeto mySense, para o envio das imagens. O objeto foi criado com o nome "Raspberry Casa", que com o tipo de conta disponível, as imagens capturadas ficam guardadas durante uma semana. Para o desenvolvimento da aplicação, foi necessária a implementação de um código em PHP (*Hypertext Preprocessor*), uma linguagem de código aberto amplamente utilizada, especialmente desenhada para desenvolvimento web podendo ser incorporada em HTML. Nesse código, e tendo como referência a documentação disponibilizada em `https://mysense.utad.pt/`, é necessário incluir o número do canal que deve ser definido no nível do objeto mySENSE como um dispositivo de imagem, as credenciais para uso da API para o envio de imagens para a plataforma, onde estão incluídas a chave API, o canal e o ID do objeto, da seguinte forma:

```
$apikey = 'YRWpyG4Sxs1HbSx6GWiQ1LwCjb8zkIeX';
$imgch = 1;
$objectfeed = 'bsisIr5w2B7mZPqB';
```

É necessário também a inserção do caminho para o ficheiro a ser enviado, através de :

```
$imgfile = '/home/pi/Pictures/imagerec_.jpg';
```

e o caminho para a API mySense, disponível em `http://mysense.utad.pt/api/imagein/`.

Executando o comando para a execução do código, surge uma resposta HTTP por parte do servidor com os dados da imagem. Nesta altura, basta aceder ao mySense, no grupo de objetos e uma imagem já estará disponível para visualização, como mostra a Figura 4.21.

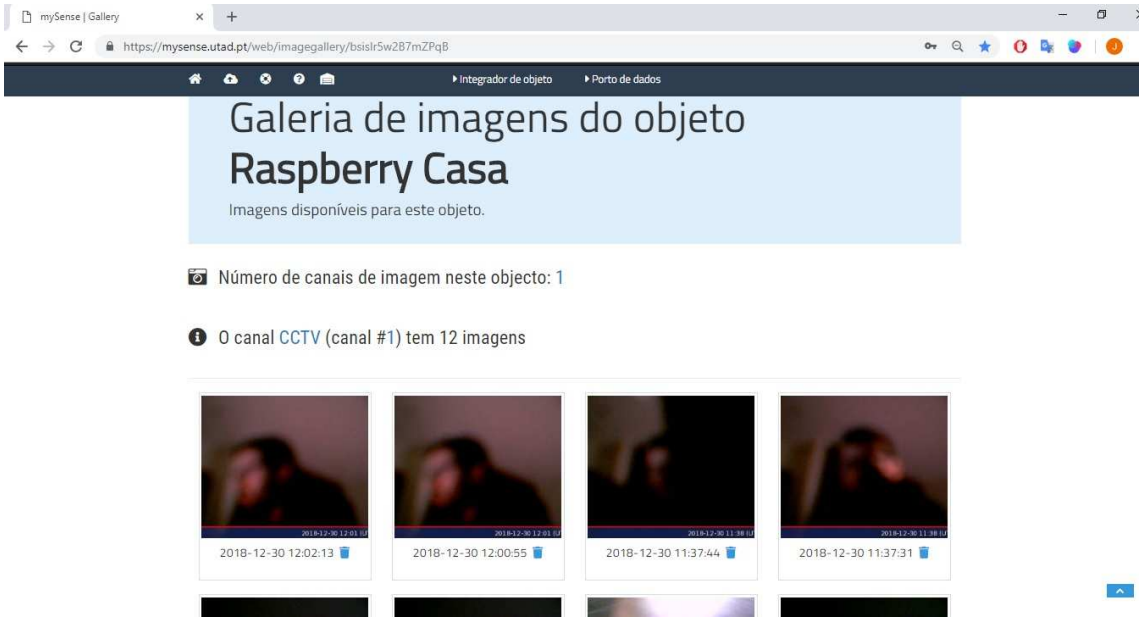


**Figura 4.21** – Imagens capturadas na plataforma mySense.

## 4.6 Registo de eventos

O processo de registo de eventos ou log de dados num sistema computacional é importante para uma melhor compreensão do sistema. Para tal, foi criado uma pasta de logs no Raspberry Pi. Dentro dessa pasta existem duas pastas, uma para o registo de dados dos sensores e outra para o registo de alarmes. Os logs de dados encontram-se guardados em ficheiros de texto e neles são registados os valores lidos pelos sensores, disparo de alarmes e respetiva data e hora dos acontecimentos. Estes dados são apagados, pelo sistema, de 24 em 24 dias. Na Figura 4.22 encontram-se como são apresentados os dados nos ficheiros, neste caso, com o exemplo de dados lidos pelo sensor de temperatura e com o registo de alarmes devido a humidade elevada.

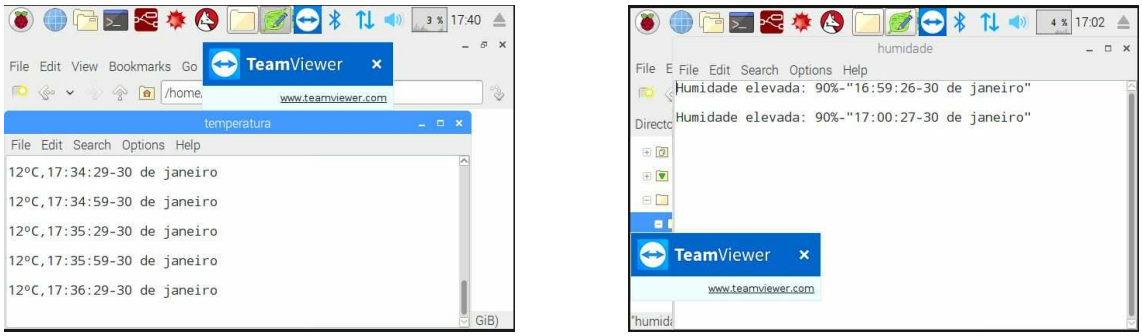


**Figura 4.22** – Registo de eventos em ficheiros de texto.

# 5 Testes e resultados

Neste capítulo serão apresentadas as operações e resultados da implementação do sistema. Os ensaios efetuados, bem como os resultados obtidos, são meramente ilustrativos das potencialidades, capacidade e expansão que se pode obter com uma rede deste tipo e dispositivos E/S utilizados. Ao nível de visualização, o uso de uma *smartTv* permite uma visualização cómoda num dispositivo já existente numa habitação.

## 5.1 Funcionamento da interface nos diferentes dispositivos

Um dos aspetos mais importantes em termos de interface e interação com o utilizador é a facilidade e migração dessa interface entre vários dispositivos, com diferentes tipos de sistemas operativos, e diferentes tamanhos de ecrã. Para tal, no `Node-RED` foi necessário fazer alguns ajustes na dimensão dos blocos, de forma a que a visualização em dispositivos móveis fosse também ela aprazível.

Como se pode verificar nas Figuras 5.1 a 5.3, os blocos da interface são apresentados

de formas diferentes, como era de esperar, mas permitindo no entanto, realizar as mesmas operações.

De notar que, inicialmente, foi usado o Google Chrome como *browser* para a construção e visualização da interface, mas que, mais tarde deixaria de ser possível ser apresentada neste *browser* em computadores, por razões alheias e desconhecidas. Para ultrapassar o problema, bastou apenas a troca do Google Chrome, pelo Microsoft Edge, *browser* nativo do Windows 10. Em telemóveis, tal erro não se registou.

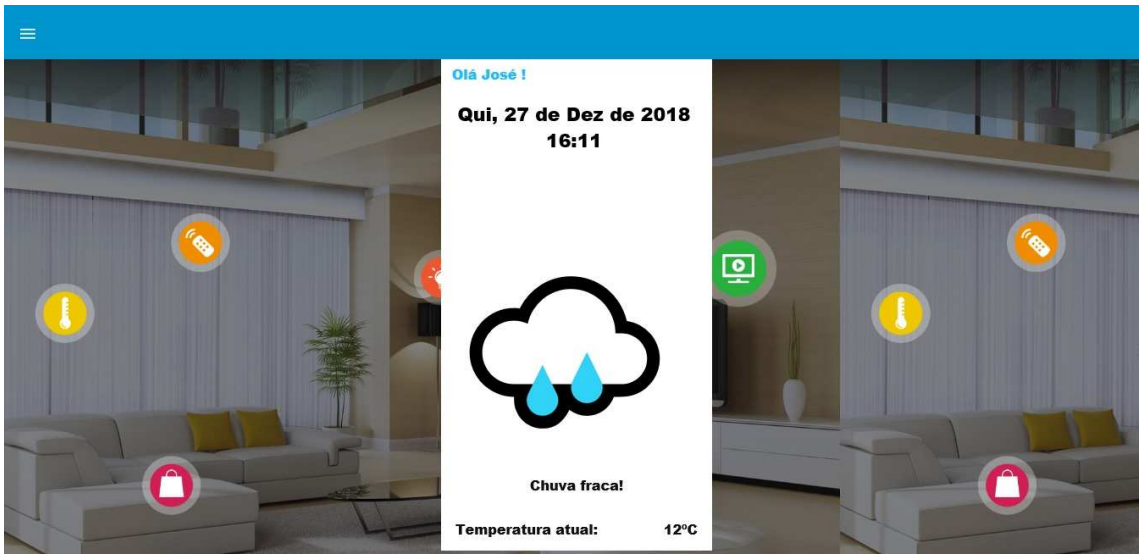


**Figura 5.1** – Imagem no PC.

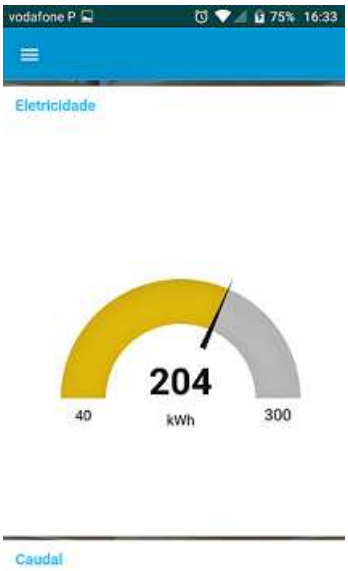


**Figura 5.2** – Imagem num telemóvel.

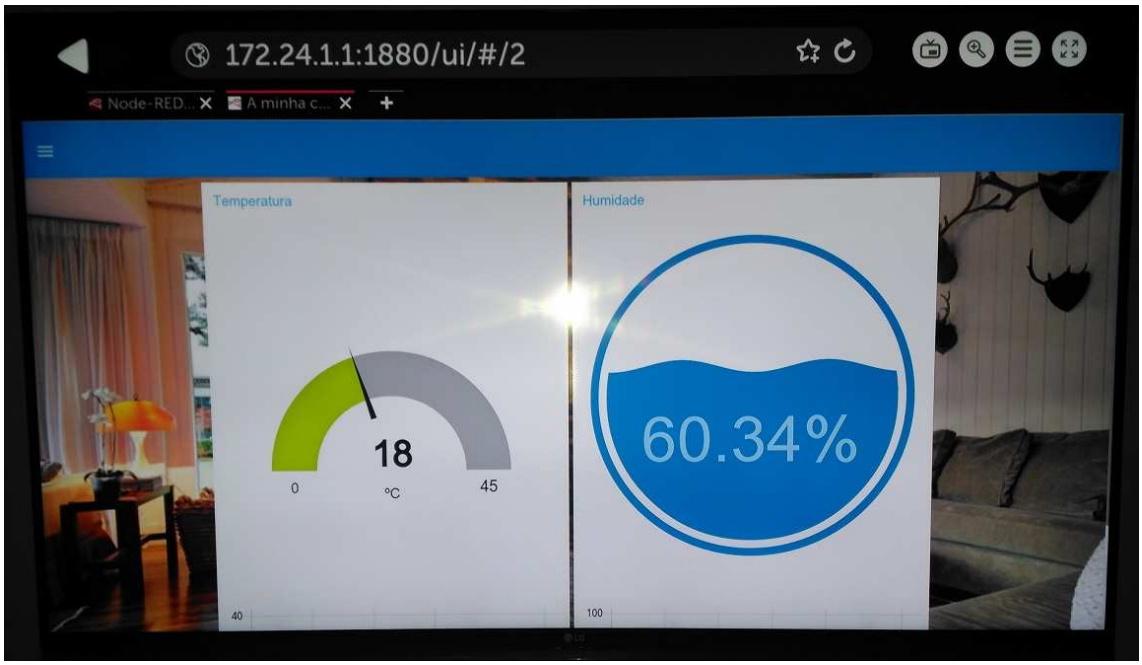


**Figura 5.3** – Dados de um dispositivo E/S visualizados numa *SmartTv*.

Por falta de compatibilidade nos diferentes sistemas operativos, não é possível mostrar a imagem ilustrativa do tempo, que se encontra numa formatação que impede a demonstração das figuras. Também de referir, que na *SmartTV*, a barra lateral, onde podemos aceder às páginas da interface web, não aparece sempre visível, apenas surge no topo das páginas.

## 5.2 Execução de testes e resultados

De forma a abranger todas as funcionalidades do sistema e visto que que no projeto são realizadas operações tanto para controlo de atuadores como para receção de dados, dividiu-se essas operações em diferentes tipos de testes. Numa primeira fase, foram analisadas as operações sobre os atuadores e, numa segunda fase, as operações que permitem a recolha de dados dos sensores.

### 5.2.1 Operações sobre atuadores

- **Ligar/desligar atuador**:

Este teste pretende verificar a possibilidade do utilizador executar ações sobre o atuador na rede criada, neste caso de o ligar ou desligar, através da interface. Para a realização deste teste, é necessário estar na página da interface com o bloco destinado ao atuador e a partir do *switch* ativar ou desativar o LED, consoante a função efetuada. É esperado que o LED ligue ou desligue, de acordo com a mensagem enviada. Em todos os testes realizados, não se verificaram falhas.

- **Temporizador**:

Este teste pretende verificar a capacidade do sistema executar funções através de um temporizador definido pelo utilizador, através da interface implementada. Para tal, no bloco do Atuador, na página "Quarto"da interface, tem-se que habilitar através

do *switch* a opção "Temporizador"e definir a data e hora para executar a ação. Espera-se que ao terminar a contagem decrescente o atuador seja ativado, podendo ser desligado no *switch* Ligar/Desligar. Também tem a possibilidade de habilitar notificações, para informar o utilizador do término da temporizaçãor. Sempre que se realizou este teste, não foram verificadas falhas.

- **Tirar fotografia**:

Este teste pretende verificar a capacidade do sistema em capturar uma imagem e exibi-la na interface. Para despoletar a ação de tirar uma fotografia, basta no bloco correspondente, carregar no botão "Tirar foto", é enviada essa informação para o Raspbberry Pi e é assim tirada uma foto através da câmara USB. Espera-se no final deste processo que a imagem capturada seja visualizada na interface, apresentando a data e hora (na própria fotografia) do momento da captura da imagem. Sempre que se realizou o teste, não foram verificadas falhas.

- **Enviar imagens para o mySense**

Neste teste, é pretendido verificar o envio das imagens para a plataforma mySense, segundo uma ordem do utilizador, clicando no botão para o efeito. Assim, para realizar esta ação, basta pressionar o botão e a última imagem guardada no diretório (já mencionado anteriormente) do Raspberry Pi, será enviada através da execuçâo de um *script* em PHP. É esperado que no final deste processo a imagem seja guardada e apresentada na plataforma. Neste teste, sempre que realizado, não se verificaram falhas.

### 5.2.2 Operações sobre informação recolhida pelos sensores

- **Leituras do sensor DHT**:

Neste teste é esperado que o utilizador seja capaz de visualizar os valores de temperatura e humidade, provenientes da leitura por parte do sensor DHT22, na interface

e de ser notificado no caso do aumento da temperatura ou humidade de acordo com o definido, no subcapítulo sobre o *software* implementado. Para isso, basta o microcontrolador estar ligado para o sensor começar a registar valores e enviá-los para a interface, periodicamente, como visto anteriormente. Para simular valores elevados de temperatura ou humidade, foi enviado manualmente um comando através da linha de comandos do Raspberry Pi, com valores superiores aos definidos no código, para fazer acionar o alarme. Neste teste, por vezes, quando não havia a possibilidade do Raspberry Pi de se ligar à Wi-Fi, os dados não são atualizados na interface, apesar disso, a maior parte do sistema sempre teve ligado à rede, havendo sempre sucesso no registo dos valores de temperatura e humidade. Quanto aos alarmes, o teste teve sempre sucesso, com a interface a notificar o utilizador do alerta.

- **Sensor de movimento PIR**:

Neste teste pretende-se mostrar a capacidade deste sensor em detetar movimento e enviar uma notificação ao utilizador através da interface. Para realizar o teste, é necessário estar ligado o microcontrolador com o sensor PIR acoplado, para iniciar de imediato o registo de movimentos. Pela passagem de uma pessoa ou de uma mão, podemos despoletar a ação pretendida. Sempre que se realizou o teste, não se verificaram falhas, com o utilizador a ser capaz de receber sempre as notificações.

- **Alarmes de intrusão**:

Neste teste pretende-se mostrar a capacidade do sistema notificar o utilizador sempre que surge uma intrusão na casa. Essa intrusão foi simulada através do corte de um circuito fechado. Ao fazer o corte desse circuito espera-se que o utilizador seja notificado. Sempre que realizado este teste, não se verificaram falhas.

### 5.2.3 Exemplo prático de uma ação sobre o sistema

Nesta secção é apresentado um exemplo prático da interação entre o utilizador e o sistema, mais concretamente, a atuação sobre o LED. Se todas as operações no

sistema forem efetuadas com sucesso, o sistema deverá encontrar-se tal como representado na Figura 5.4, isto é, a alteração do estado do LED de desligado para ligado.

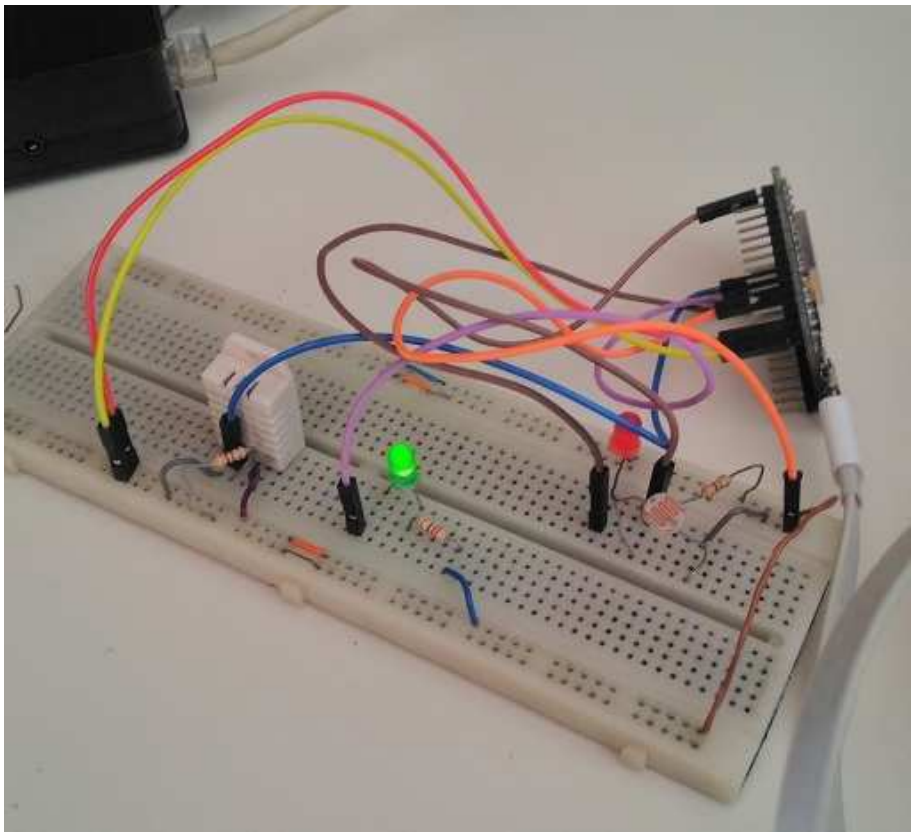

**Figura 5.4** – Exemplo prático de ligação de um sensor e de um atuador ao dispositivo E/S.

O exemplo prático escolhido, foi então, a atuação sobre o atuador do sistema, neste caso o LED que se encontra ligado. O outro LED que se pode ver na imagem é o que se encontra conectado ao sensor LDR, que neste caso e com a incidência da luz naquela altura, permanece desligado.

Para atuar sobre o LED, o utlizador realiza um pedido através da interface web, que comunica com o servidor do sistema, neste caso, o Raspberry Pi, que por sua vez comunica com o microcontrolador. O microcontrolador, como já visto, faz pedidos GET através de uma ligação HTTP ao servidor, e após essa comunicação e analisar o estado que se pretende, volta a enviar um "ON"ou "OFF"para o servidor e que

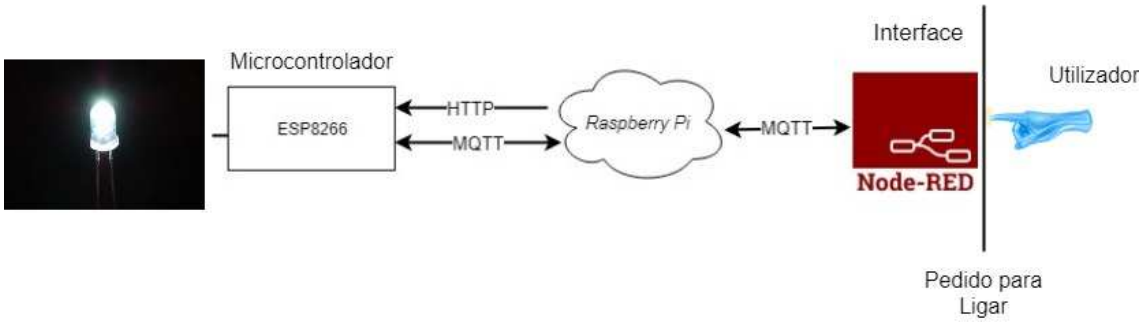


**Figura 5.5** – Diagrama funcional de um pedido para ligar um atuador da rede.

por sua vez faz ligar o LED. A Figura 5.5 mostra de forma simplificada como são executados os processos.

# 6 Conclusão e trabalhos futuros

Nos dias que correm e com a evolução de tecnologias, como as que suportam o paradigma da IoT, a evolução de pequenos, mas cada vez mais poderosos microcontroladores e sensores, faz com que a domótica seja uma área em constante desenvolvimento. Apesar disso e de existir já uma grande variedade de soluções comerciais no mercado, os custos para a implementação de um sistema desse tipo ainda apresenta custos elevados, não existindo uma norma global para sistemas domóticos.

Foi nesse sentido, que este trabalho se idealizou, na perspetiva de criar uma sistema amigável de automação residencial, onde os utilizadores poderão ter uma comunicação aprazível com o sistema onde é possível o controlo e monitorização de vários aspetos de uma habitação através de uma interface web.

No final, os objetivos propostos inicialmente foram cumpridos, sendo possível, para um utilizador, interagir com o sistema, através da interface criada, tal como apresentado no capítulo 5, sendo possível a visualização de aspetos importantes, tais como a humidade e temperatura. Apesar disso, o sistema, ainda permite uma grande evolução, de forma a ficar com um nível superior e ser apresentado como um produto comercial. Dessa forma, propõem-se algumas funcionalidades e objetivos para trabalhos futuros:

- Inserção da interface criada num servidor/nuvem, onde poderia ser acessível externamente à rede criada.
- Captura automática de imagens e vídeo.
- Mais opções de autenticação no sistema, havendo a possibilidade da criação de várias contas para diferentes utilizadores.
- Enviar uma notificação através de e-mail ao utilizador no caso de algum alarme surgir na habitação.
- Desenvolver os dispositivos de E/S com múltiplas funções com encapsulamento e alimentação elétrica adequados.

# Referências bibliográficas


Arduino. External interrupts. https://www.arduino.cc/reference/en/language/functions/external-interrupts/attachinterrupt/. Acedido a 27.12.2018. 58

Arduino. Interrupts. https://www.arduino.cc/reference/en/language/functions/interrupts/interrupts/. Acedido a 27.12.2018. 51

ASecureLife (2018). No Title. https://www.asecurelife.com/. Acedido a 14.11.2018. 27

Assistant, H. No Title. https://www.home-assistant.io/. Acedido a 14.11.2018. xix, 32

Baker, J. (2017). 6 open source home automation tools. https://opensource.com/tools/home-automation. Acedido a 27.12.2018. 31

Banggood. Geekcreit® NodeMcu Lua WIFI Internet Things

Development Board Based ESP8266 CP2102 Wireless Module. https://www.banggood.com/NodeMcu-Lua-WIFI-Internet-Things-Development-Board-Based-ESP8266-CP2102-Wireless-Module-p-1097112.html?rmmds=myorder&cur_warehouse=CN. Acedido a 12.09.2018. xx, 50

Bticino (2016). GUIDELINES FOR DESIGN AND INSTALLATION. Acedido a 13.11.2018. 27

BUYECOMPONENTS. LDR 5MM. https://buyecomponents.com/shop-electronic-components-in-dubai/sensors/product-70/. Acedido a 15.11.2018. xx, 53

Chen, G., Feng, H., Xie, H., Zhan, R., Wu, Q., Guan, X., Wang, A., Takasuka, K., Tamura, S., Wang, Z., and Zhang, C. (2004). Characterizing diodes for RF ESD protection. *Electron Device Letters, IEEE*, 25(5):323–325. 11

da Eletrônica, B. (2018). NodeMcu ESP-12E. http://blog.baudaeletronica.com.br/esp8266/fig4-2/. Acedido a 06.10.2018. xx, 50

Datafloq. Internet of things (iot): Security, privacy and safety. https://datafloq.com/read/internet-of-things-iot-security-privacy-safety/948. Acedido a 20.12.2018. 5

Domoticz (2015). Domoticz. (February). Acedido a 14.11.2018. xix, 32

Espressif (2018). Esp8266. https://www.espressif.com/en/products/hardware/esp8266ex/overview. Acedido a 27.12.2018. 40

Foundation, J. Node-red - flow-based programming for the internet of things. https://nodered.org/. Acedido a 22.12.2018. 60

Friedewald, M., Da, O., and Punie, Y. (2005). Perspectives of ambient intelligence in the home environment q. 22:221–238. 10

Gewiss. Domotics. https://www.gewiss.com/ww/pt/products/experience-catalogue/catalogs/domotics. Acedido a 13.11.2018. 23

Gordon, A. W. (2010). MQTT-S – A Publish / Subscribe Protocol for Wireless Sensor Networks. (February):1–19. xix, 13, 16, 17

GorillaBuilderz. HC-SR501 PIR Sensor. https://www.gorilladistribution.com.au/product/hc-sr501-pir-sensor/. Acedido a 15.11.2018. xx, 53

Gubbi, J., Buyya, R., Marusic, S., and Palaniswami, M. (2013). Internet of Things (IoT): A vision, architectural elements, and future directions. *Future Generation Computer Systems*, 29(7):1645–1660. 9, 10

Hager. Instalações inteligentes tebis KNX. Acedido a 13.11.2018. 25

HiveMQ. Mqtt essentials part 4: Mqtt publish, subscribe unsubscribe. https://www.hivemq.com/blog/mqtt-essentials-part-5-mqtt-topics-best-practices/. Acedido a 21.12.2018. 15

HiveMQ. Mqtt essentials part 5: Mqtt topics best practices. https://www.hivemq.com/blog/mqtt-essentials-part-5-mqtt-topics-best-practices/. Acedido a 21.12.2018. 14

In, R. (2011). M2M : From Mobile to Embedded Internet. (April):36–43. 12

Karagiannis, V., Chatzimisios, P., Vazquez-Gallego, F., and Alonso-Zarate, J. (2015). A Survey on Application Layer Protocols for the Internet of Things. *Transaction on IoT and Cloud Computing*, 3(1):11–17. xvii, 18, 19

Kelley, S. (2018). Dnsmasq. http://www.thekelleys.org.uk/dnsmasq/doc.html. Acedido a 22.12.2018. 38

Kim, J., Lee, J., Kim, J., and Yun, J. (2014). M2M service platforms: Survey, issues, and enabling technologies. *IEEE Communications Surveys and Tutorials*, 16(1):61–76. 11

Li, S., Xu, L. D., and Zhao, S. (2015). The internet of things: a survey. *Information Systems Frontiers*, 17(2):243–259. xix, 6, 7

Light, R. mqtt — mq telemetry transport. https://mosquitto.org/man/mqtt-7.html. 13

Livingdam. Home automation. https://livingdam.pt/pt/home-automation. Acedido a 28.12.2018. 30

Miorandi, D., Sicari, S., De Pellegrini, F., and Chlamtac, I. (2012). Internet of things: Vision, applications and research challenges. *Ad Hoc Networks*, 10(7):1497–1516. 3, 8

Mittal, S. (2016). IoT Ecosystem: How the IoT Market will Explode by 2020. https://blog.beaconstac.com/2016/03/iot-ecosystem-iot-business-opportunities-and-forecasts-for-the-iot-market/. Acedido a 15.09.2018. xix, 11

Morais, R. (2016). mysense iot++ - uma plataforma para agricultura de precisão. https://mysense.utad.pt/main. Acedido a 27.12.2018. 73

MySensors. openHAB. https://www.mysensors.org/controller/openhab. Acedido a 14.11.2018. xx, 34

Nerokas. DHT22 AM2302 Temperature and Humidity sensor. https://store.nerokas.co.ke/index.php?route=product/product&product_id=249. Acedido a 15.11.2018. xx, 52

Node.js. About node.js. https://nodejs.org/en/about/. Acedido a 21.12.2018. 48

NodeMcu. Nodemcu connect things easy. http://www.nodemcu.com/index_en.html. Acedido a 21.12.2018. 49

OpenHAB. No Title. Acedido a 14.11.2018. xx, 33

OpenWeather. Openweathermap. https://openweathermap.org/. Acedido a 22.12.2018. 63

Pedro Pinto (2017a). MEO lança MEO Smart Home: A sua casa vai ser inteligente. https://pplware.sapo.pt/informacao/meo-vai-tornar-casa-inteligente-meo-smart-home/. Acedido a 12.11.2018. xix, 20

Pedro Pinto (2017b). Meo Smart Home: Vamos conhecer a plataforma Web. https://pplware.sapo.pt/gadgets/hardware/meo-smart-home-conhecer-plataforma-web/. Acedido a 12.11.2018. 22

Pellegrino, P., Bonino, D., and Corno, F. (2006). Domotic house gateway. *Proceedings of the 2006 ACM symposium on Applied computing SAC 06*, page 1915. 2

Pi, R. Raspberry pi. https://www.raspberrypi.org/. Acedido a 22.12.2018. 46

PubSubCLient. Api documentation for mqtt arduino client. https://pubsubclient.knolleary.net/api.html. Acedido a 27.12.2018. 58

Realinho, V. (2015). Low Cost Domotic System based on Open Hardware and Software. *In Proceeding of The Eighth International Conference on Advances in Human-oriented and Personalized Mechanisms, Technologies, and Services-oriented and Personalized Mechanisms, Technologies, and Services (CENTRIC 2015)*, (c):13–16. 1

Saraswathi, E., Kumar, A., Singh, J., Mohanty, J., and Mishra, Y. (2018). Arduino Based Home Automation System Using MQTT Protocol Incorporating Internet of Things (IOT). 8(5):3. 13

Singh, M., Rajan, M. A., Shivraj, V. L., and Balamuralidhar, P. (2015). Secure MQTT for Internet of Things (IoT). *Proceedings - 2015 5th International Conference on Communication Systems and Network Technologies, CSNT 2015*, pages 746–751. 17

Thangavel, D., Ma, X., Valera, A., Tan, H. X., and Tan, C. K. Y. (2014). Performance evaluation of MQTT and CoAP via a common middleware. *IEEE ISSNIP 2014 - 2014 IEEE 9th International Conference on Intelligent Sensors, Sensor Networks and Information Processing, Conference Proceedings*, (April):21–24. xvii, 18, 19

Townsend, C. and Arms, S. (2004). Wireless Sensor Networks:. pages 439–450. 9

Zhang, Y., Yu, R., Xie, S., Yao, W., Xiao, Y., and Guizani, M. (2011). Home M2M networks: Architectures, standards, and QoS improvement. *IEEE Communications Magazine*, 49(4):44–52. 11

# A Requisitos do sistema

Neste anexo serão apresentados alguns requisitos do sistema, entenda-se por sistema o conjunto nó principal mais os nós secundários - atuadores/sensores, que poderão ou não ser totalmente implementados. Os principais componentes do sistema em questão serão o Raspberry Pi 3, os microcontroladores, sensores, a interface, o tempo e o próprio utilizador (estes dois últimos os atores do sistema). Tratando-se de um sistema domótico é esperado a total ou parcial automatização da habitação. Vamos ter, portanto, controlo de iluminação, aparelhos domésticos, sistemas de rega, segurança, alarmes técnicos, entre outros. Tendo por base estes fatores comecemos então por enumerar alguns dos requisitos funcionais:

1. Nó principal
   (a) Deve criar uma rede Wi-Fi.
   (b) Deverá ser o responsável pela gestão das mensagens entre dispositivos (*broker*), armazenamento das mensagens bem como responsável pela atuação de alguns nós.
2. Nós secundários - Atuadores/Sensores
   (a) Os nós deverão ser capazes de se ligar à rede Wi-Fi

(b) Os nós deverão ser capazes de se registar no sistema de mensagens.

(c) Os nós deverão obter dados de sensores e publicá-los no sistema.

(d) Alterar o estado dos atuadores.

(e) O utilizador será capaz de controlar aspetos gerais do sistema através de comando de vozes, como, controlo de luzes, estores, eletrodomésticos, televisão e realizar chamadas.

(f) Alerta de inundação - Reação: desligar electroválvula; envio de mensagem ao utilizador

(g) Alerta de fumos nocivos - Reação: ligar ventoinhas ou mecanismo de sucção de ar; envio de mensagem ao utilizador.

(h) Alerta de intrusão - Reação: ativar barulhos sonoros e ligação de luzes; envio de SOS as autoridades competentes e ao utilizador.

(i) O sistema deverá ser capaz de detetar humidade na superfície de alguma divisão (principalmente na casa-de-banho) através de um sensor e alertar ao utilizador.

(j) Alerta de incêndio - Reação: ligar aspersores

(k) Em caso de incêndio o sistema deverá desligar todos os dispositivos de cozinha ou suscetíveis a combustão.

(l) O utilizador poderá ser capaz de operar equipamentos como micro-ondas, forno através da interface do sistema

(m) O sistema devera incorporar câmaras, capazes de enviar imagens ao utilizador, de modo a ver o que existe na sua despensa.

3. Sensores

(a) O sistema deverá ser capaz obter sinais de entrada de detetores de fumo.

(b) Deverá ser capaz de obter sinais de entrada de sensores de movimento.

(c) Obter sinais de entrada de temperatura.

(d) Obter sinais de entrada de humidade.

(e) Obter sinais de entrada de luminosidade.

4. Tempo

   (a) Terá a responsabilidade de promover alterações nos atuadores mediante uma data.

   (b) Os temporizadores do sistema poderão ser ciclos ou não cíclicos (únicos).

5. Interface com o utilizador

   (a) Deverá apresentar informações num browser.

   (b) A aplicação deverá ser um diagrama com a planta da casa.

   (c) Ter a capacidade se ligar a redes de telemóveis (GSM por exemplo).

   (d) Capacidade de receber e enviar SMS.

   (e) O utilizador poderá através da aplicação registar elementos no sistema.

   (f) Inserir novos eventos e tarefas.

   (g) Ler dados do nó principal do sistema.

   (h) Solicitar dados a um nó especifico.

   (i) Editar as configurações dos nós do sistema, tais como, horários de atuação, valores dos sensores para posterior atuação (exemplo temperatura e ação do ar condicionado).

   (j) Registar e apagar temporizações, por exemplo, na hora da rega e fecho de estores.

   (k) A aplicação devera alertar o utilizador sobre alarmes e notificar o utilizador para eventos do sistema através de SMS ou email.

   (l) O sistema deverá permitir comandos de voz via telemóvel.

   (m) O sistema poderá enviar informações através do telemóvel.

6. Eventos

   (a) Luz

i. O sistema será capaz de controlar as luzes através de um temporizador.
ii. Através de comandos de voz ligar/desligar e regular a intensidade luminosa.
iii. Através de comandos de sons para ligar/desligar e regular a intensidade luminosa.
iv. Através de sensores de luminosidade regular a intensidade mediante as leituras do sensor.
v. Através da deteção de movimentos.
vi. Permitir a sua interação sem necessitar do sistema, isto é, deverá ser autónomo ao nó principal.

(b) Ambiente da casa

i. O sistema será capaz de controlar a temperatura através de um temporizador.
ii. Através de comandos de voz ligar/desligar e regular a temperatura.
iii. Através de comandos de sons para ligar/desligar e regular a temperatura.
iv. Ajuste automático (para valores predefinidos pelo utilizador) com a recolha de dado de sensores de temperatura.
v. Ligação automática da rega com a ajuda de um temporizador.
vi. Ligação automática da rega através de um sensor de humidade.
vii. Ligação automática da rega através de sensor e temporizador.

(c) Segurança

i. Através de sensores na porta de entrada contar as pessoas que entram e saem.
ii. O utilizador poderá configurar esses mesmos sensores para não atuar enquanto está em casa por exemplo.
iii. Configurar os sensores para atuar em determinado período em que ninguém estará na habitação por exemplo.

iv. Enviar esses dados periodicamente para o nó principal e para a interface.

v. Caso sejam ativados os sensores nesses períodos enviar informação ao utilizador através de e-mail ou SMS e ativar alarmes técnicos de intrusão.

# B Objeto JSON

Este anexo encontra-se o objeto JSON usado na implementação do sistema, onde todos os pares - chave/valor - poderão ou não estar totalmente implementados no sistema.

```
  {
"id":"000",
"nome": "casa",
"ndivisoes": 6,
"nos":
[
  {
    "id":"001",
    "nome":"no1",
    "tipo":"misto",
    "divisao":"sala",
    "canal": [
                {
                  "nome": "atuador",
                  "ncanal":"1",
                  "tipo":"output",
                  "estado":"ativo",
```

```
                "valorAtual":[{"desligado":"off"},
                              {"ligado":"on"}
                              ],
                "topicoMQTT":"casa/sala/atuador",
                "timestamp":"hora"
              },
              {
                "nome":"temperatura",
                "ncanal":"2",
                "tipo":"output",
                "estado":"ativo",
                "valorAtual":[
                               {
                                 "temperatura": "String(t)"
                               }
                              ],
                "topicoMQTT":"casa/sala/temperatura",
                "timestamp":"hora"
              },

              {
                "nome":"humidade",
                "ncanal":"3",
                "tipo":"output",
                "estado":"ativo",
                "valorAtual":[
                               {
                                 "humidade": "String(h)"
                               }
                              ],
                "topicoMQTT":"casa/sala/humidade",
                "timestamp":"hora"
              }
            ]
  },
```

```
{
  "id":"002",
  "nome":"no2",
  "tipo":"misto",
  "divisao":"quarto2",
  "canal": [
            {
               "nome": "ldr",
               "ncanal":"4",
               "tipo":"output",
               "estado":"ativo",
               "valorAtual": [
                               {
                                 "ldr":"analogRead(A0)"
                               }
                             ],
               "topicoMQTT":"casa/sala/ldr",
               "timestamp":"hora"
             }
          ]
},

{
  "id":"003",
  "nome":"no3",
  "tipo":"misto",
  "divisao":"geral",
  "canal": [
             {
               "nome": "consumoEletrico",
               "ncanal":"5",
               "tipo":"output",
               "estado":"ativo",
               "valorAtual": [
                               {
                                 "eletricidade": "String(e)"
```

```
                                }
                            ],
                "topicoMQTT":"casa/geral/consumos/eletricidade",
                "timestamp":"hora"
              },

              {
                "nome": "consumoAgua",
                "ncanal":"6",
                "tipo":"output",
                "estado":"ativo",
                "valorAtual": [
                                {
                                  "caudal":"String(c)"
                                }
                            ],
                "topicoMQTT":"casa/geral/consumos/caudal",
                "timestamp":"hora"
              }
            ]
},

{
  "id":"004",
  "nome":"alarmes",
  "tipo":"misto",
  "divisao":"geral",
  "canal": [
              {
                "nome":"intrusao",
                "ncanal":"7",
                "tipo":"output",
                "estado":"ativo",
                "valoratual":0,
                "topicoMQTT":"casa/geral/alarmes/intrusao",
                "timestamp":"hora"
```

```
                    },

                    {
                      "nome":"temperatura",
                      "ncanal":"8",
                      "tipo":"output",
                      "estado":"ativo",
                      "valoratual":0,
                      "topicoMQTT":"casa/geral/alarmes/temperatura",
                      "timestamp":"hora"
                    },

                    {
                      "nome":"humidade",
                      "ncanal":"9",
                      "tipo":"output",
                      "estado":"ativo",
                      "valoratual":0,
                      "topicoMQTT":"casa/geral/alarmes/humidade",
                      "timestamp":"hora"
                    },

                   {
                      "nome":"movimento",
                      "ncanal":"10",
                      "tipo":"output",
                      "estado":"ativo",
                      "valoratual":0,
                      "topicoMQTT":"casa/geral/alarmes/pir",
                      "timestamp":"hora"
                    }
                  ]
    }
  ]
}
```