UNIVERSIDADE AUTÓNOMA DE LISBOA
LUÍS DE CAMÕES

DEPARTAMENTO DE CIÊNCIAS E TECNOLOGIA
LICENCIATURA EM ENGENHARIA INFORMÁTICA

(Desenvolvimento de um protótipo de automação para *Smart Home* por comando de voz)

Relatório para obtenção do grau de licenciado

Autores: Gonçalo Coelho; Gonçalo Jorge; Miguel Rodrigues e Maria João Elias
Orientador: Professor Héctor Dave Orrillo Ascama
Número dos alunos: 30006613; 30006661; 30006525 e 30006976

Junho de 2023
Lisboa

Resumo

Nos dias que correm, os sistemas inteligentes são progressivamente mais comuns e a sua integração na vida do ser Humano tem um impacto bastante positivo no seu quotidiano. Uma das maneiras imprescindíveis para tal começa na nossa própria casa ao torná-la inteligente. Ao abrigo do conceito de *Smart Home*, implementou-se um protótipo automatizado composto por um estore (prova de conceito) sinalizado por luzes LED, tomadas inteligentes e um sensor de humidade e temperatura, controlado por uma aplicação desenvolvida em openHAB, que fornece aos seus utilizadores controlo manual e por voz (*Google Assistant*) da respetiva infraestrutura. As dificuldades ao sucesso do objetivo foram maioritariamente de ordem monetária e algumas com questões logísticas.

Palavras-chave: Comandos de voz, estores, humidade, LEDs, openHab, sensor, Shellys, *Smart Home*, temperatura, tomadas inteligentes.

Abstract

These days, intelligent systems are progressively more common and their integration into human life has a very positive impact on their daily lives. One of the essential ways to do this starts in our own homes by making it “smart”. Under the *Smart Home* concept, an automated prototype was implemented consisting of roller shutters (proof of concept) signalled by LED lights, smart plugs and a humidity and temperature sensor, controlled by an application developed in openHAB, which provides the manual and voice control (Google Assistant) of the respective infrastructure. The difficulties in achieving the goal were mostly monetary and some related to logistical issues.

Keywords: Commands, Blinds, Home Automation, openHAB, Power Outlets, Sensor, Temperature, Humidity, Shellys, LEDs.

Índice

Lista de Tabelas

Lista de Figuras

Lista de Siglas e Acrónimos

Sigla	Nomenclatura
IOT	*Internet of Things*
API	*Application Programming Interface*
LED	*Light Emitting Diode*
USB	*Universal Serial Bus*
HDMI	*High-Definition Multimedia Interface*

VSCode	*Visual Studio Code*
RAM	*Random Access Memory*
GPIO	*General-Purpose Input/Output*
ARM	*Advanced RISC Machine*
LAN	*Local Area Network*
DC	*Direct Current*
HR	*Humidity Ratio*
MHz	*Megahertz*
URL	*Uniform Resource Loader*
AC	*Alternate Current*
VCA	*Voltage Controlled Amplifier*
VCC	*Voltage at the Common Collector*
IBM	*International Business Machines*
VPN	*Virtual Private Network*
GHz	*Gigahertz*
SD	*Secure Digital*
SSH	*Secure Shell*
IP	*Internet Protocol*
Mbps	*Megabits por segundo*
OFDMA	*Orthogonal Frequency-Division Multiple Access*
ISEP	*Instituto Superior de Engenharia do Porto*

ISTEC	*Instituto Superior de Tecnologias Avançadas*
ISEL	*Instituto Superior de Engenharia de Lisboa*
PLC	*Programmable Logic Controller*
BLE	*Bluetooth Low Energy*
LCG	*Led Control Gear*
TTS	*Text-to-Speech*

1 Introdução

O presente trabalho tem como principal objetivo explorar e desenvolver a temática de *Smart Home* por meio de um projeto e relatório abrangentes. Pretendemos abordar de forma detalhada as metas que almejamos alcançar, justificando e explicando as decisões que foram tomadas no decorrer do desenvolvimento do projeto, bem como a relevância do tema escolhido para os dias de hoje.

1.1. Descrição do problema

No mundo atual, onde a presença de dispositivos e aparelhos inteligentes se torna cada vez mais prevalente, a automação para *Smart Home* torna-se algo indispensável. A *Smart Home* representa uma área em contínuo crescimento e desenvolvimento, envolvendo a integração de dispositivos e sistemas tecnológicos em residências. O objetivo é proporcionar maior comodidade, eficiência energética, segurança e conforto aos seus utilizadores. Com o avanço da Internet das Coisas (IoT), a transformação digital e a crescente conectividade, a *Smart Home* desperta um interesse cada vez maior em todo o mundo.

Neste contexto, a automação *Smart Home* oferece uma solução integrada e simplificada para gerir e controlar remotamente dispositivos como iluminação, climatização, eletrodomésticos e sistemas de segurança. Aplicativos móveis e assistentes de voz, como o Google Assistant ou a Alexa da Amazon, fornecem um controlo mais intuitivo e personalizado da casa, de acordo com as necessidades e preferências individuais, proporcionando praticidade e eficiência aos seus utilizadores.

Além da comodidade, a automação residencial tem o potencial de promover a eficiência energética. Sensores inteligentes monitorizam o consumo de energia e ajustam automaticamente a iluminação, a temperatura e outros dispositivos para minimizar desperdícios. Isto resulta numa redução dos gastos com energia, contribuindo para a preservação do meio ambiente.

Em resumo, a automação *Smart Home* torna-se indispensável no mundo atual, oferecendo praticidade, eficiência energética e segurança aos utilizadores, integrando e simplificando o controlo de dispositivos, e promovendo uma experiência mais sustentável e confortável. Com a possibilidade de controlar remotamente a casa e otimizar o consumo de

energia, a automação *Smart Home* oferece benefícios significativos para os utilizadores, tornando-se uma tendência crescente na busca por um estilo de vida moderno e automatizado

1.2. Objetivos

1.2.1. Objetivo geral

O objetivo geral do projeto consiste em desenvolver um protótipo de automação para *Smart Home*, capaz de controlar dispositivos através de comandos de voz e de forma manual, por meio de uma aplicação *mobile*. De modo a atingir esse objetivo, será necessário configurar e programar os dispositivos envolvidos, possibilitando o controlo remoto por meio de sistemas de reconhecimento de voz e a integração dos mesmos numa aplicação de automação residencial. Esta aplicação permitirá a gestão centralizada e a interação entre diferentes dispositivos, criando um ambiente inteligente e integrado, que poderá ser representado pela figura

Figura 1 Exemplificação da interação entre dispositivos Smart Home

Fonte: https://harakis.com

1.2.2. Objetivos específicos

Os objetivos específicos do projeto desenvolvido são os seguintes:

1. Desenvolver um sistema de automação de estores utilizando um motor bidirecional capaz de realizar a subida e descida de estores.
2. Transformar tomadas convencionais em tomadas inteligentes através de um clique.

3. Implementar um sensor de humidade e temperatura no sistema de automação residencial, responsável por monitorizar as condições ambientais, fornecendo informações precisas sobre a humidade e temperatura.
4. Desenvolver e estabelecer a comunicação e acionamento dos dispositivos *Shelly* por meio de serviços, programados em Java, entre a plataforma/aplicação openHAB e o Raspberry Pi permitindo, assim, o controlo remoto e a integração/controlo dos dispositivos em um único ambiente inteligente.
5. Desenvolver funcionalidades de controlo de voz em conformidade com a aplicação de controlo manual do sistema.
6. Implementar um protótipo à escala reduzida (maquete) para validar o correto funcionamento do sistema de automação residencial. A maquete será constituída por todos os componentes até agora mencionados e permitirá realizar testes e ajustes necessários antes da implementação em escala real.

Os objetivos mencionados visam o desenvolvimento de um protótipo funcional de automação residencial, integrando estores automatizados, tomadas inteligentes, sensor de humidade e temperatura, e proporcionando controlo remoto, por voz e manual aos seus utilizadores. A comunicação entre os dispositivos por meio de uma rede *Wi-Fi* e a validação do sistema por meio de um protótipo à escala reduzida asseguram o correto funcionamento e eficácia do sistema de automação residencial para uma futura implementação real.

1.3. Justificativa

A realização deste projeto de automação residencial em escala reduzida é justificada pelo crescente interesse e procura de soluções que facilitem e aprimorem as tarefas do quotidiano. A automação residencial tem se mostrado uma resposta eficaz às necessidades atuais, proporcionando maior comodidade, eficiência energética, segurança e conforto aos utilizadores.

Ao desenvolvermos um projeto de automação residencial em menor escala, temos como objetivo demonstrar de forma tangível e prática as possibilidades e benefícios desta tecnologia. Através da maquete desenvolvida, podemos ilustrar como a integração de dispositivos inteligentes, o controlo remoto e o uso de tecnologias avançadas podem transformar a forma como interagimos com a nossa casa, otimizando a realização de tarefas domésticas.

Além disso, este projeto serve como uma base sólida para futuras implementações em maior escala, permitindo a identificação de desafios, ajustes e melhorias necessárias. Por meio dessa iniciativa, contribuímos para o avanço e a evolução do campo da automação residencial, fornecendo *insights* valiosos para a indústria e despertando o interesse de potenciais investidores, fabricantes e consumidores.

Em suma, a justificativa para a realização deste projeto reside na necessidade de explorar e demonstrar as vantagens e potencialidades da automação residencial, impulsionando a adoção dessa tecnologia e promovendo um estilo de vida mais conveniente, eficiente e conectado.

1.4. Estrutura do trabalho

O presente relatório encontra-se organizado em 6 tópicos principais, os quais são tratados como capítulos para simplificação logística. Os capítulos são, assim, divididos de acordo com os tópicos que a seguir se referem, e são compostos por vários subtópicos/subtemas, cada um.

O primeiro capítulo, "Introdução", apresenta a problemática e os objetivos, específicos e gerais, daquilo que se pretende fazer durante e até ao término do desenvolvimento do objeto de trabalho, bem como a justificação inerente à iniciativa de investir numa criação destas.

O segundo capítulo, "Fundamentação teórica", tal como o próprio nome indica, engloba subtópicos que se focam na explicação da teoria que fundamenta esta proposta prática. É explicado o conceito de domótica, são apresentados alguns trabalhos relacionados ao tema proposto, definidos os conceitos de microcontrolador e do *hardware* utilizados, explicado o ambiente de desenvolvimento virtual (editor), sobre o mecanismo de comunicação do sistema e componentes físicos do sistema, embora este com um ênfase superior derivado da relevância e centralização que o microcontrolador possui nesta prototipagem.

O terceiro capítulo, "Metodologia do modelo proposto", é dedicado a uma maior aproximação ao projeto real, sendo descrita a metodologia do modelo. O que se pretende nesta seção é fazer uma apresentação mais visual e descritiva, inicialmente geral e posteriormente mais particular, sobre como irá ser montado o sistema que se pretende desenvolver, referindo-se, para isso, pormenores como a composição e a função, quer em termos de *hardware* quer em termos de *software* dos componentes envolvidos no conjunto. Refere-se também como será construída a maquete do protótipo e a aplicação de controlo dos dispositivos.

Segue-se o capítulo 4, “Testes e análises de resultados”, onde são descritos os tipos de testes realizados e feita uma análise acerca dos resultados dos mesmos.

De seguida, apresenta-se um capítulo sobre as “Conclusões”, na qual se explicitam os resultados inerentes ao desenvolvimento do projeto bem como um capítulo adicional sobre as limitações e problemas. Por fim, surge um capítulo acerca dos “Trabalhos futuros”, com recomendações e sugestões de melhorias a implementar e refere como este trabalho poderá influenciar futuros trabalhos na mesma área ou até fora desta. Os últimos capítulos desta ata incidem sobre elaboração da bibliografia, de apêndices e anexos.

2 Fundamentação teórica (Desenvolvimento)

2.1 Domótica

2.1.1 Conceito de Domótica

A domótica é a tecnologia responsável pela gestão e automação de recursos em residências e espaços habitacionais. O termo "domótica" deriva da junção das palavras "*domus*", que significa "casa" em latim, e "robótica", relacionada à automação de ações por meio de dispositivos mecânicos ou eletrônicos.

A domótica engloba a integração de diversos sistemas e dispositivos, como iluminação, climatização, segurança, comunicação, energia e eletrodomésticos, para proporcionar maior conforto, segurança e eficiência energética. Por meio de sensores, controladores e sistemas inteligentes, é possível gerir e controlar esses recursos de forma automatizada e remota.

O objetivo da domótica é criar ambientes residenciais inteligentes, nos quais os moradores possam desfrutar de maior comodidade e praticidade. Por exemplo, é possível programar o acionamento automático das luzes conforme a luminosidade ambiente, regular a temperatura ambiente de forma automática, controlar dispositivos eletrônicos por meio de comandos de voz ou smartphones, monitorizar sistemas de segurança e receber alertas em caso de eventos indesejados, entre outras funcionalidades.

Figura 2 - ilustração do conceito de Smart Home

Fonte: https://casaeficiente.com

2.1.2 Trabalhos relacionados

Neste subtópico, serão apresentados três projetos relacionados e desenvolvidos em função do tema *Smart Home*, que fornecem *insights* valiosos utilizados no desenvolvimento do projeto atual.

O trabalho "Projeto e automação de casa inteligente", desenvolvido por FONSECA e MONTEIRO (s.d.), do ISEP (Instituto Superior de Engenharias do Porto), aborda o desenvolvimento de um projeto para casa inteligente, que pode ser controlado por meio de uma página web pelo utilizador. O objetivo deste projeto é incorporar uma ampla gama de *hardware* e software, permitindo novas abordagens no contexto de Licenciatura em Engenharia Eletrotécnica e de Computadores (FONSECA e MONTEIRO, s.d.).

O segundo trabalho "Domótica com Arduino", desenvolvido por João Pereira (2015), do ISTEC (Instituto Superior de Tecnologias Avançadas), aborda aspetos relacionados à aplicação prática do Arduino na área da domótica, fornecendo *insights* valiosos sobre o uso dessa plataforma em soluções residenciais inteligentes. A partir dos exemplos práticos apresentados, é possível expandir o conhecimento e explorar diferentes abordagens no desenvolvimento de soluções de domótica (PEREIRA, 2015).

Finalmente, o terceiro trabalho "Projeto de sistema de supervisão para habitações", desenvolvido por João Machado (2020), do ISEL (Instituto Superior de Engenharia de Lisboa), apresenta um projeto de um sistema domótico aplicado a uma residência convencional, onde apenas quatro sistemas de controlo foram selecionados para gerenciar a casa. Os dispositivos escolhidos foram destacados pela sua simplicidade, uma vez que todo o processo de configuração e processamento de informações é realizado pelo autómato, resultando numa arquitetura de controlo centralizada. Uma característica relevante deste projeto é o facto do único dispositivo envolvido com capacidade de processamento ser o PLC (Controlador Lógico Programável), o que torna os demais equipamentos mais acessíveis em termos económicos. Essa abordagem permite uma solução mais viável financeiramente para a supervisão e controlo da habitação.

Ao adotar uma arquitetura centralizada, o projeto procura simplificar a implementação e a operação do sistema domótico, oferecendo uma solução eficiente e acessível para o controlo residencial. A concentração do processamento no PLC permite que os demais equipamentos desempenhem funções específicas sem a necessidade de recursos computacionais avançados, o que contribui para a viabilidade económica do projeto. Com a abordagem referida, o trabalho destaca a importância de uma solução domótica acessível em termos monetários, fornecendo

uma alternativa mais econômica para a supervisão e controlo de uma residência. A análise cuidadosa dos requisitos e da arquitetura do sistema demonstra a preocupação em procurar soluções eficientes e adaptadas às necessidades da habitação, sem comprometer a funcionalidade e a usabilidade (MACHADO, 2020).

2.2 Microcontrolador (Raspberry Pi)

O Raspberry Pi é um microcontrolador de baixo custo e tamanho reduzido, projetado para fins educativos, prototipagem e desenvolvimento de projetos de computação em pequena escala. Este surgiu como uma resposta à crescente demanda por plataformas acessíveis e flexíveis para a experimentação e o desenvolvimento de projetos de eletrónica, é baseado numa arquitetura de processador ARM e é capaz de executar um sistema operativo completo, como o Linux, oferecendo uma ampla gama de funcionalidades e recursos.

O Raspberry Pi apresenta uma variedade de interfaces de entrada/saída, incluindo portas USB, HDMI, Ethernet, GPIO (General-Purpose Input/Output), entre outras, que permitem a conexão e controlo de diferentes dispositivos externos. Estas interfaces versáteis permitem que o Raspberry Pi seja utilizado numa ampla gama de aplicações, desde sistemas embarcados até soluções de automação residencial e projetos de IoT.

Com um processador de baixo consumo de energia e recursos de conectividade, o Raspberry Pi oferece flexibilidade para desenvolver soluções personalizadas e adaptáveis, seja para fins educativos, *hobbie*s ou até mesmo para aplicações comerciais. Este também possui uma comunidade ativa de *developers* e entusiastas, o que promove a troca de conhecimentos e o compartilhamento de projetos e recursos adicionais.

Em suma, o Raspberry Pi é um microcontrolador acessível e poderoso que combina capacidades de computação, interfaces de entrada/saída versáteis e um ecossistema de suporte, permitindo a criação e o desenvolvimento de uma variedade de projetos inovadores e soluções personalizadas de maneira eficiente e econômica.

2.2.1 *Hardware*

O microcontrolador que iremos utilizar é um computador Raspberry Pi 400 integrado diretamente num teclado.

Figura 3- Raspberry Pi 400 integrado diretamente num teclado

Fonte: https://mauser.pt

As especificações técnicas do microcontrolador são as seguintes:

- Processador: Broadcom BCM2711 *quad-core* Cortex-A72 (ARM v8) 64-bit SoC @ 1.8GHz
- RAM: 4GB LPDDR4-3200
- Conectividade:
    - Dual-band (2.4GHz and 5.0GHz) IEEE 802.11b/g/n/ac LAN sem fios,
    - Bluetooth 5.0, BLE (Bluetooth Low Energy)
    - Gigabit Ethernet
    - 2 portas USB 3.0 e 1 porta USB 2.0
- GPIO: Conector GPIO horizontal de 40 pins
- Vídeo e som: 2 x portas micro HDMI (suporta até 4Kp60)
- Multimédia:
    - H.265 (descodifica 4Kp60);
    - H.264 (descodifica 1080p60, codifica 1080p30);
    - OpenGL ES 3.0 graphics
- Suporte de cartões de SD: Ranhura para cartão MicroSD para armazenar dados e o SO (sistema operativo)
- Teclado: Compacto de 79 teclas
- Alimentação: 5V DC por conector USB-C
- Temperaturas de operação: ambiente de 0ºC a +50ºC

- Dimensões máximas: 286mm x 122mm x 23mm

2.3 Ambiente de Desenvolvimento (Editor)

O ambiente de desenvolvimento utilizado no decorrer do projeto foi o Visual Studio Code *(*VSCode), um editor *open-source* e gratuito desenvolvido pela Microsoft, lançado oficialmente em novembro de 2015. Desde então, tem ganho popularidade devido à sua interface intuitiva e extensibilidade.

Este ambiente de desenvolvimento avançado e altamente personalizável foi projetado para facilitar e otimizar o processo de criação de software e aumentar a produtividade dos programadores através de recursos poderosos e uma ampla variedade de extensões. Entre esses recursos, destacam-se o realce de sintaxe, o autocompletar do código, a depuração integrada, o controlo de versões Git, a análise de código estático e a refatoração de código, que auxiliam os programadores a produzir uma escrita mais eficiente, precisa e de alta qualidade. Seguidamente estão listadas as funcionalidades descritas, de forma mais íntegra:

- Interface de usuário intuitiva: Possui uma interface de usuário intuitiva que facilita a navegação pelo editor e a realização de tarefas comuns de desenvolvimento de software.
- Completamento de código: Oferece um recurso avançado de preenchimento automático de código que ajuda a acelerar o processo de escrita e reduzir erros.
- Análise de código: Possui um poderoso mecanismo de análise de código que ajuda a identificar possíveis erros e problemas de desempenho antes de executar o código.
- Depuração: Oferece ferramentas avançadas de depuração, permitindo que se encontre e corrija erros no código de forma eficiente.
- Suporte a várias linguagens de programação, incluindo Kotlin, Scala, Python, JavaScript e Java (esta última sendo a que iremos utilizar)
- Gestão de projetos: Possui recursos avançados de gestão de projetos, incluindo integração com sistemas de controlo de versão, gestão de dependências e muito mais.
- Plugins e extensões: Oferece suporte a muitos plugins e extensões que permitem personalizar o editor e adicionar recursos adicionais para aumentar a sua produtividade.
- Integração com outras ferramentas: Pode ser integrado com outras ferramentas de desenvolvimento, como servidores de aplicativos, bases de dados e ferramentas de construção, para facilitar o desenvolvimento e a implantação de aplicações.

Para além de tudo isto, como já mencionado, o VSCode possui ainda uma arquitetura extensível que permite a integração de uma ampla variedade de extensões e *plugins* desenvolvidos pela comunidade. Essas extensões adicionam funcionalidades específicas, suporte a diferentes linguagens de programação, *frameworks* e ferramentas adicionais, personalizando o ambiente de desenvolvimento de acordo com as necessidades individuais de cada projeto.

Uma das vantagens do VSCode é a sua ampla compatibilidade com várias plataformas, incluindo Windows, macOS e Linux, garantindo que os programadores possam usufruir das suas funcionalidades, independentemente do sistema operacional escolhido. Além disso, o facto de ser *open-source*, significa que o código-fonte está disponível publicamente e pode ser acedido, visualizado e modificado por qualquer pessoa, permitindo que qualquer utilizador o examine, perceba o seu funcionamento e até contribua para o seu desenvolvimento. Ademais, esta característica garante uma maior transparência, confiabilidade, colaboração e personalização do software.

Assim, consideramos que a escolha do *Visual Studio Code* como ambiente de desenvolvimento neste projeto proporcionará uma plataforma sólida e flexível para a implementação e execução das tarefas de programação, garantindo a eficiência e produtividade ao longo de todo o processo de desenvolvimento.

Figura 4- Visual Studio Code

Fonte: https://code.visualstudio.com

2.3.1 - Extensão openHab

A extensão openHAB do VSCode é uma ferramenta que agiliza o desenvolvimento e a configuração da automação residencial usando a plataforma openHAB que fornece recursos

e funcionalidades específicas para facilitar a edição de arquivos de configuração e a interação com a plataforma openHAB, como:

1. Destaque de sintaxe: A extensão oferece realce de sintaxe para o arquivo de configurações do openHAB, facilitando a visualização e identificação correta dos diferentes elementos e comandos utilizados.
2. IntelliSenses: O recurso de Intellisense fornece sugestões de preenchimento automático enquanto se digita, com base nas configurações e elementos definidos no ambiente openHAB, agilizando o processo de escrita de código e evitando erros de digitação.
3. Validação de código: ocorre para identificar erros ou problemas de formatação nos arquivos, o que garante que as configurações estejam corretas e evita problemas durante a execução da automação residencial.
4. Navegação rápida: Com esta extensão, é possível navegar facilmente entre os diferentes arquivos de configuração do openHAB, visualizando mais rápida e facilmente os diferentes elementos e configurações relacionadas.
5. Depuração (de configurações do openHAB): permite que os utilizadores verifiquem e solucionem problemas durante a execução da automação residencial.

Essas funcionalidades, fornecidas pela extensão openHAB do VSCode, visam tornar o processo de desenvolvimento e configuração de automação residencial mais eficiente e produtivo, ajudando os utilizadores a criar e gerenciar as suas configurações openHAB com maior facilidade e precisão.

2.4 openHAB

O *Open Home Automation Bus* (openHAB) é um software de automatização doméstica de código aberto que opera em Java e foi lançado em 2010. Segundo o *website Black Duck Open Hub*, o openHAB foi e continua a ser desenvolvido por uma das maiores equipas de desenvolvimento de código *open-source* e possui, também, uma grande comunidade de utilizadores ativa.

O openHAB oferece, assim, uma abordagem flexível e escalável para controlar e monitorizar diferentes aspetos de uma casa, como iluminação, segurança ou a climatização. Uma das suas características distintivas é a sua capacidade de integrar uma ampla variedade de dispositivos e tecnologias, podendo ser executado numa variedade de plataformas, incluindo o Windows, Linux, macOS e Raspberry Pi, e proporcionando uma plataforma centralizada para

gerir todos os recursos habitacionais. De modo a realizar essa integração, utiliza os chamados *bindings*, que são componentes responsáveis por estabelecer a comunicação entre o sistema e os dispositivos específicos. Esses *bindings* atuam como "pontes" que permitem o controlo e a monitorização dos dispositivos, através da plataforma openHAB.

Neste projeto, um dos *bindings* utilizados é o *Shelly Binding. O Shelly Binding* é responsável por estabelecer a conexão entre os dispositivos *Shelly* e a plataforma openHAB, simplificando a integração e permitindo o controlo desses dispositivos através de uma interface unificada no openHAB, e oferecendo uma série de vantagens como:

- A capacidade de integrar uma infinidade de outros dispositivos e sistemas, incluindo sistemas de automação residencial, dispositivos inteligentes e outras tecnologias numa única solução;
- Fornecer uma interface de utilizador uniforme e unificada, bem como uma abordagem comum às regras de automação em todo o sistema, independentemente do número de fabricantes e subsistemas envolvidos;
- É uma ferramenta mais flexível, que permite que os seus utilizadores personalizem e adaptem as configurações de automação residencial de acordo com suas necessidades específicas;
- Não necessita de serviço *cloud* para funcionar, o que ajuda a manter os dados privados e seguros, ao permitir utilizar uma abordagem local à problemática da automatização. No entanto, funciona de uma maneira "*cloud-friendly"*, ou seja, amiga de serviços em nuvem. Um dos serviços disponíveis para o efeito é o serviço *cloud* , um serviço criado pela openHAB Foundation (empresa criadora do openHAB) que permite que os utilizadores se liguem de maneira remota à plataforma.

Figura 5 - openHab

Fonte: https://community.openhab.org

2.5 Mecanismo de comunicação do sistema

2.5.1 openHab (Wi-Fi)

O mecanismo de comunicação do sistema é um elemento essencial para assegurar a conectividade e interação entre os dispositivos e a plataforma openHAB. No contexto da automação residencial, a comunicação sem fios é uma opção amplamente adotada devido à sua conveniência e flexibilidade, sendo um dos meios mais utilizados o Wi-Fi.

O Wi-Fi é uma tecnologia de comunicação sem fios que permite a transmissão de dados através de redes locais. No âmbito da automação residencial, a utilização do Wi-Fi possibilita a conexão e o controlo de dispositivos inteligentes numa casa conectada, permitindo o acesso remoto e a automação dos equipamentos e sistemas.

No caso específico do openHAB, esta foi concebida para suportar a integração com dispositivos e sistemas que utilizam comunicação Wi-Fi. Isto significa que os dispositivos compatíveis com o Wi-Fi podem ser facilmente conectados à plataforma openHAB para fins de controlo e monitorização. Deste modo, ao utilizar o openHAB em conjunto com dispositivos Wi-Fi, os utilizadores podem controlar e interagir com uma vasta gama de equipamentos, como lâmpadas inteligentes, termostatos, fechaduras eletrónicas e/ou câmaras de segurança.

A utilização da comunicação Wi-Fi no âmbito da automação residencial oferece diversas vantagens, tais como a facilidade de instalação, a ampla disponibilidade de dispositivos compatíveis e a capacidade de controlo remoto em tempo real. Com o openHAB e a comunicação Wi-Fi, os utilizadores podem criar cenários personalizados, programar horários de funcionamento, monitorizar o consumo de energia e integrar os seus dispositivos inteligentes com outras tecnologias e serviços residenciais.

Em resumo, o mecanismo de comunicação do sistema no contexto do openHAB envolve a utilização da comunicação sem fios, nomeadamente o Wi-Fi, para ligar e controlar dispositivos inteligentes numa casa conectada. Esta abordagem proporciona aos utilizadores uma maior flexibilidade e conveniência para automatizar e gerir os seus equipamentos residenciais de forma centralizada e remota.

2.5.2 Assistente de voz: *Google Assistant*

A Google iniciou-se unicamente como uma entidade que oferecia pesquisa *online*. No entanto, atualmente, oferece mais de 50 serviços de Internet e produtos, desde *e-mails* a *software* de telemóveis (tal como o sistema operativo Android) e computadores (tal como o *browser* Google Chrome). Além disso, em 2012, ao adquirir a Motorola Mobility, a Google

pôde iniciar-se no mundo do *hardware* móvel, sobre a forma de telemóveis. A Google está, hoje em dia, no topo de empresas de tecnologia, ao lado de outros gigantes, tais como a Microsoft, a IBM e a Apple.

No contexto deste projeto, a API (*Application Programming Interface)* da Google Assistant será utilizada como assistente de voz. Desenvolvida pela Google e lançada em 2018, a API está disponível para plataformas Android, iOS e Wear OS. Ela servirá como base para a implementação da funcionalidade de controlo por voz, complementando a criação de ações e a associação de dispositivos à aplicação desenvolvida pelo grupo.

A Google Assistant possui uma longa lista de funcionalidades e habilidades como a capacidade de responder a perguntas, e torna-se ainda mais útil quando se trata de resultados personalizados. Ao permitir o acesso desta conta a outros serviços, a assistente pode oferecer mais do que informações gerais, como eventos marcados no calendário, a previsão do tempo local e o estado do trânsito. Além disso, a Google Assistant é extremamente conveniente quando se trata de dispositivos de *Smart Home*. Ela é compatível com uma variedade de marcas populares de casas inteligentes, incluindo Philips Hue, SmartThings, Nest, Ring, WeMo e várias outras.

A integração da *Smart Home* com a Google Assistant é facilitada pela aplicação *Google Home*, que se torna o centro principal para o controlo de todos os dispositivos, permitindo uma experiência mais abrangente do controlo da *Smart House*. Através do *Google Home*, os utilizadores podem interagir diretamente com a Google Assistant para controlar uma ampla variedade de dispositivos domésticos inteligentes, como as luzes, termostatos, fechaduras de portas, câmeras de segurança, tomadas inteligentes e muito mais. Contudo, os dispositivos físicos são apenas uma parcela do vasto conjunto de competências da Google Assistant, uma vez que estas podem também conectar-se a aplicações e serviços da web, como o Spotify, o YouTube ou a Netflix.

Além disso, a Google Assistant oferece recursos em smartphones e tablets, como a reprodução de notificações em voz alta, realização de chamadas e envio de mensagens de texto. Mesmo sem altifalantes inteligentes e dispositivos domésticos sofisticados, a Google Assistant permanece uma ferramenta útil.

Figura 6 - Google Assistant

Fonte: https://www.pcguia.pt

2.6 Componentes físicos

2.6.1 Raspberry PI (ver índice 2.2)

2.6.2 Router ASUS RT-AX56U

Como mecanismo de comunicação de sistemas foi utilizado um router da ASUS possuído por um dos constituintes do grupo. Com origem em 1989, a marca ASUS representa uma empresa multinacional amplamente reconhecida por fabricar *motherboards* (placas-mãe), além de computadores pessoais, monitores, placas gráficas, routers e outras soluções tecnológicas. Atualmente, a ASUS posiciona-se como uma marca comprometida em criar e desenvolver tecnologias inteligentes de última geração, com o intuito de oferecer experiências positivas aos seus clientes por meio de seus produtos.

Figura 7-Router ASUS utilizado para o projeto

Fonte: https://www.asus.com

Seguidamente, na tabela 1, expõem-se as especificações técnicas deste produto:

Modelo	• RT-AX56U
Normas de rede	• IEEE 802.11a • IEEE 802.11b • IEEE 802.11g • IEEE 802.11n • IEEE 802.11ac • IEEE 802.11ax • IPv4 • IPv6
Segmento do produto	• AX1800 ultimate AX performance : 1201 Mbps+ 574 Mbps • AX technology → Sim
Taxa de dados	• 802.11a : até 54 Mbps • 802.11b : até 11 Mbps • 802.11g : até 54 Mbps • 802.11n : até 433 Mbps • 802.11ac : até 867 Mbps • 802.11ax (2.4GHz) : até 574 Mbps • 802.11ax (5GHz) : até 1201 Mbps
Antena	• Antena externa x 2
Transmissão/Receção	• 2.4 GHz 2 x 2 • 5 GHz 2 x 2
Processador	• 1.5GHz processador *quad-core*
Conteúdo do pacote	• RT-AX56U • RJ45 x1 • Adaptador x1 • QSG x1 • Cartão de garantia x1
Peso do produto (g)	• 456 g • AiMesh → Sim • Router APP → Sim • Alexa → Sim • IFTTT → Sim
AiProtection	• Proteção Ai Profissional → Sim • Controlo Parental → Sim
Controlo de tráfego	• Regra do limitador de banda-larga máxima: 32 • QoS Tradicional • Máximo de regras QoS tradicionais: 32 • Período de análise de tráfego : Diário, Semanal, Mensal
Wireless	• Encriptação Wi-Fi : Sistema aberto, WPA/WPA2/WPA3-Personal, WPA/WPA2-Enterprise • Regra de rede máxima para convidados: 2,4 GHz x3, 5 GHz x3 • Limite de tempo da ligação de rede do convidado

	• Encriptação de rede para convidados : Sistema aberto, WPA/WPA2-Personal • Máximo de filtros MAC : 64
WAN	• Tipo de ligação à Internet: PPPoE, PPTP, L2TP, IP automático, IP estático • Regra máxima de reencaminhamento de portas : 64 • Regra máxima de acionamento do porto: 32 • Passagem NAT: PPTP, L2TP, IPSec, RTSP, H.323, passagem SIP, retransmissão PPPoE
LAN	• Regra máxima de atribuição manual de endereços IP: 64 • VPN → Sim
Aplicação USB	• Sistema de ficheiros : HFS+, NTFS, vFAT, ext2, ext3, ext4
Administração	• Modo de funcionamento : Ponto de acesso, nó AiMesh, ponte multimédia, repetidor, router • Sistema operativo : ASUSWRT • Filtro máximo de palavras-chave da Firewall : 64 • Filtro máximo de serviços de rede da Firewall : 32 • Filtro máximo de URL da Firewall : 64
Memória	• 256 MB Flash • 512 MB RAM
Aumenta a velocidade	• OFDMA (Orthogonal Frequency-Division Multiple Access; Acesso múltiplo por divisão ortogonal de frequências) • *Beamforming*: *standard-based and universal* (normalizado e universal) • 1024-QAM de alta taxa de dados • Banda-larga de 20/40/80 MHz
Frequências de operação	• 2.4G Hz / 5 GHz
Portas	• RJ45 para Gigabits BaseT para WAN x 1, RJ45 para Gigabits BaseT para LAN x 4 • USB 2.0 x 1 • USB 3.1 Gen 1 x 1 •
Botões	• WPS, Reset (Reiniciar), Power Switch (Interruptor de alimentação)
Indicador LED	• Power (Alimentação) x 1 • 2.4G x 1 • 5G x 1 • LAN x 4 • WAN x 1
Fonte de alimentação	• Entrada AC : 110V~240V(50~60Hz) • Saída DC: 12 V com max. 2 A de corrente

Tabela 1- especificações técnicas do router ASUS

Fonte: https://www.asus.com/pt

2.6.3 Sensor de temperatura/humidade

A *Shelly* é a principal marca da empresa europeia de tecnologia Allterco, que se especializa na inovação, produção e distribuição de produtos relacionados com a *IoT*, tornando-se numa das marcas de crescimento mais rápido no mundo, através dos seus dispositivos e ao fornecer soluções para a automação de casas e edifícios. Com atenção meticulosa aos detalhes e seguindo as tendências tecnológicas mais avançadas, os profissionais da *Shelly* desenvolvem os seus produtos para que seja possível otimizar o consumo de energia através de processos inteligentes, proporcionando aos seus clientes experiências eficientes em termos energéticos. Em apenas alguns anos, a marca conquistou presença em cem mercados ao longo de três continentes, e atualmente mantém três escritórios principais.

Um dos produtos da Shelly é o sensor *Shelly H&T - Wi-Fi,* um sensor de temperatura e humidade que utiliza comunicação por *Wi-Fi*, comunicando diretamente com redes *Wi-Fi,* como o tipo de rede doméstica para o qual este projeto prático é pretendido. Diferenciando-se de outras marcas, o sensor *Shelly H&T* não requer equipamentos adicionais, como *bridges*, que aumentam o custo dos sensores de temperatura e humidade. Este sensor é pequeno, discreto, e funciona com pilhas ou através de cabos USB.

Para além disso, tem compatibilidade com uma grande variedade de plataformas de automação residencial, incluindo o próprio aplicativo da Shelly, a SHELLY APP, que disponibiliza um conjunto de configurações com o propósito de explorar as potencialidades dos seus dispositivos, ao mesmo tempo que permite conectar os dispositivos *Shelly* à *Shelly Cloud*, possibilitando o acesso remoto à automação residencial de qualquer lugar, mesmo que o utilizador esteja fora de casa.

O sensor em si monitoriza, em tempo real, os níveis de humidade e temperatura num determinado local, sendo possível o seu armazenamento gratuito na *cloud*, por 365 dias. Esta funcionalidade é especialmente relevante, considerando-se que o sensor consome uma quantidade mínima de energia, o que resulta numa vida útil estimada de aproximadamente um ano para o dispositivo É, além disso, possível cruzar os dados obtidos pelas medições do sensor H&T com outros dispositivos Shelly para os acionar de maneira automática, criando cenários de reações inteligentes a certos dados.

Figura 8 - Shelly H&T

Fonte: https://mauser.pt

As especificações do sensor ilustrado na figura acima, são as que a seguir se mostram:

- Sensor de temperatura: Sim
- Sensor de humidade: Sim

AMBIENTAIS

- Temperatura de funcionamento: -10 °C a 50 °C / 14 °F a 122 °F
- Sensor de humidade: 30 % a 70 % HR
- Altitude máxima: 2000 m / 6562 pés

ELÉTRICAS

- Alimentação: pilha CR123A 3V DC (não recarregável, não incluída)
- Duração estimada da pilha: Até 18 meses
- Consumo de energia: Estático ≤70uA / Activo ≤250mA
- Alimentação por USB: Opcional (através de uma base – não incluída)

WI-FI

- Protocolo: 802.11 b/g/n
- Banda RF: 2401 – 2495 MHz
- Potência máx. RF: <20 dBm
- Alcance do Wi-Fi: Até 30m em ambientes internos e 50m em ambientes externos (Depende das condições locais)
- MCU
- CPU: ESP8266

- Flash: 2 MB

FUNCIONALIDADES DE FIRMWARE

- Controlo local e remoto: Sim
- Webhooks (acções URL): 5 com 5 URLs por gancho
- Scripting: Não
- MQTT: Não

CARACTERÍSTICAS FÍSICAS

- Tamanho (AxLxP): 35×45 mm
- Peso: 16g sem pilha / 33g com pilha
- Montagem: Sem fios
- Material do invólucro: Plástico
- Cor: Preto ou Branco

2.6.4 Módulos Shelly

2.6.4.1 Módulo Shelly Plus 2PM

O Shelly Plus 2PM é um módulo de relé Wi-Fi que oferece funcionalidades avançadas para a automação residencial. Com este dispositivo, é possível controlar e monitorizar remotamente diversos dispositivos elétricos e equipamentos através de um aplicativo de *smartphone*, ou de uma plataforma de automação residencial.

Equipado com dois relés de saída, o Shelly Plus 2PM permite a ativação e desligamento de dispositivos como lâmpadas, eletrodomésticos e sistemas de aquecimento. A conexão à rede Wi-Fi existente na residência possibilita o controlo remoto dos dispositivos conectados, permitindo uma gestão conveniente a partir de qualquer lugar com acesso à Internet.

Além disso, o Shelly Plus 2PM oferece recursos avançados como monitorização do consumo de energia, programação de horários de funcionamento e integração com assistentes de voz populares, como Amazon Alexa e Google Assistant. Compatível com plataformas de automação residencial, como o Home Assistant, este dispositivo proporciona uma experiência inteligente e conectada no ambiente residencial, permitindo a criação de cenários personalizados e contribuindo para a economia de energia e a comodidade dos utilizadores.

Figura 9- Shelly Plus 2PM

Fonte: https://www.gerafluxo.com

Em seguida, seguem-se as configurações deste dispositivo, para o seu melhor entendimento:

- Conexão por Wi-Fi: conecta-se à sua rede Wi-Fi. Não é necessário HUB
- Bluetooth: Adiciona dispositivos de forma rápida e fácil via conexão Bluetooth, usando o aplicativo Shelly Cloud
- 2 canais na mesma fase
- Medição de potência em cada canal
- Controlo de cobertura (rolo): automatiza e ajusta a posição de persianas, portões, cortinas, toldos, portas de correr, porta de garagem, outros motores bidirecionais
- Ampla gama de suporte de tensão
- *Scripts*, *schedules* e *webhooks*
- Fonte de alimentação AC: 110-240V ±10%, 50/60Hz
- Fonte de alimentação DC: 24 V ±10%
- Canais: 2
- Máx. corrente por canal: 10A
- Máx. corrente total do dispositivo: 16A (pico de 18A)
- Máx. tensão de comutação: 240 VCA / 30 VCC
- Contactos secos: Não
- Proteção de sobrecarga: Sim
- Medição de potência: Sim
- Proteção contra sobretemperatura: Sim
- Trabalha sem linha neutra: Não

- HTTP/HTTPS *webhooks*: Sim
- *Scripts* personalizados (mJs): Sim
- Temperatura de operação: -20°C a + 40°C
- Consumo de energia do dispositivo: < 1,4 W
- Controlo local e remoto: Sim
- Nascer/pôr do sol: Sim
- Horário semanal: Sim
- Opção listada pela UL: Não
- Protocolo Wi-Fi: 802.11 b/g/n
- Frequência de rádio Wi-Fi: 2400 - 2500 MHz
- Potência do sinal de rádio Wi-Fi: 1mW
- Alcance Wi-Fi: até 50 m no exterior e até 30 m no interior (dependendo dos materiais de construção)
- Dimensões: 42 x 38 x 17mm

2.6.4.2 ShellyPlus 1PM

O ShellyPlus 1PM é um módulo de relé Wi-Fi utilizado para a automação residencial e controlo de dispositivos elétricos, desenvolvido pela Shelly, uma empresa especializada em soluções inteligentes para casas conectadas e permite, à semelhança do anterior, controlar e monitorizar dispositivos elétricos através de uma aplicação para *smartphone* ou uma plataforma de automação residencial. Por outro lado, contrariamente ao Shelly Plus2PM, este módulo possui apenas um relé de saída.

Uma das principais vantagens permanece a sua conectividade Wi-Fi, que possibilita o acesso e controlo remoto dos dispositivos a partir de qualquer lugar, desde que haja acesso à Internet, e é ainda igualmente possível a sua integração com assistentes de voz.

Com a sua instalação e configuração simples, o ShellyPlus 1PM é uma solução versátil e acessível para a automação residencial, permitindo aos utilizadores criar cenários personalizados, poupar energia e ter um maior controlo sobre os dispositivos elétricos da habitação.

Figura 10 - Shelly 1PM

Fonte: https://www.luedeke-elektronic

Deste modo, para melhor elucidar as características que compõem este este elemento, estão descritas abaixo as suas componentes:

- Conexão por Wi-Fi - conecta-se à sua rede Wi-Fi. Não é necessário HUB
- Bluetooth - Adiciona dispositivos de forma rápida e fácil via conexão Bluetooth, usando o aplicativo Shelly Cloud
- Medição de energia com armazenamento de dados
- Grande variedade de tensões suportadas
- Processador extremamente rápido - ESP32
- Proteção elétrica - proteção contra sobretensão e sobreaquecimento
- Segurança aprimorada - suporte MQTT e WSS, TLS e suporte a certificados personalizados
- Interface de API aprimorada
- Fonte de alimentação AC: 110-230V ±10%, 50/60Hz
- Fonte de alimentação DC: 24 - 240V
- Contactos secos: Não
- Proteção de temperatura do dispositivo: Sim
- Proteção de sobrecarga: Sim
- Medição de potência: Sim
- HTTP/HTTPS webhooks: Sim
- Scripts personalizados (mJs): Sim
- Canais: 1 canal

- Carga máxima: 16A
- Temperatura de operação: 0°C a + 40 °C
- Consumo de energia do dispositivo: < 1 W
- On/Off Inteligente: Sim
- Controle local e remoto: Sim
- Nascer do sol/ pôr do sol: Sim
- Horário semanal: Sim
- Opção listada pela UL: Não
- Protocolo Wi-Fi: 802.11 b/g/n
- Frequência de rádio Wi-Fi: 2412 - 2484 MHz
- Potência do sinal de rádio Wi-Fi: 1mW
- Alcance Wi-Fi: até 50 m no exterior e até 30 m no interior (dependendo dos materiais de construção)
- Bluetooth: Sim
- Dimensões: 41mm x 36mm x 15mm

2.6.5 Leds (Motor e tomada inteligente)

Embora o objetivo inicial deste projeto fosse simular o movimento dos estores por meio de um motor bidirecional, questões financeiras impediram a sua implementação. Nesse sentido, o grupo optou por utilizar LEDs como uma alternativa mais económica, mas igualmente eficaz, para ilustrar o funcionamento dos estores. Deste modo, o projeto passou a possuir dois tipos de LEDs, um monocromático, que sinaliza apenas a cor branca, e outro com duas cores: vermelho e verde (ver 3.3.2 Implementação Física do Protótipo).

2.6.6 Led Control Gear

O Led Control Gear (LCG) é um componente essencial para o controlo dos LEDs num sistema de iluminação. Ele encontra-se conectado aos módulos Shelly, que por sua vez estão ligados ao interruptor de controlo, que permite ligar e desligar os LEDs conforme necessário, bastando, para isso, conectar o LCG a uma tomada elétrica padrão (ver 3.3.2 Implementação Física do Protótipo).

O interruptor de controlo conectado aos módulos Shelly, permite o acionamento manual das LEDs. Ao ligar ou desligar o interruptor, os comandos são transmitidos para os Shellys, que controlam o funcionamento dos LEDs de acordo com a configuração desejada.

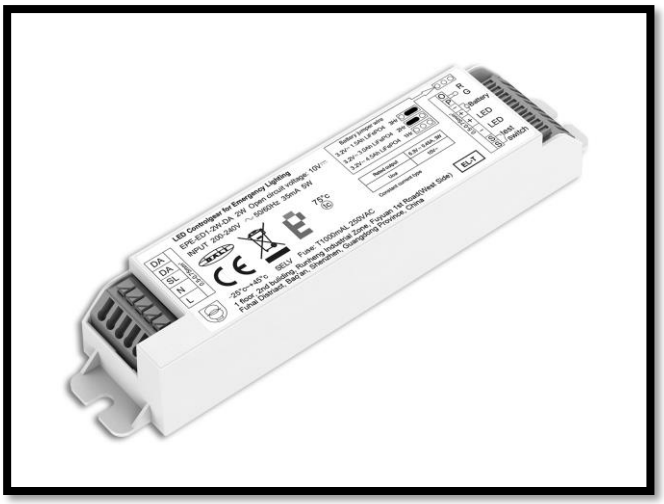


Figura 11 - Led Control Gear

Fonte: https://images.app.goo.gl/VzqiKCECJS1VNi998

3 Metodologia do projeto proposto

3.1 Apresentação geral do modelo proposto

O projeto que se pretende desenvolver engloba conhecimentos em várias áreas, tais como Eletrónica, Redes, Telecomunicações e Informática, sendo necessários alguns conhecimentos dentro dessas áreas para compreender o modelo proposto.

De um modo geral, o modelo centra-se no microcontrolador Raspberry Pi 400 com uma instalação do openHABian 3.4.4. O microcontrolador, que tem a função de servidor, interliga os dispositivos Shelly, os quais estão ligados a lâmpadas LED, ao sensor de temperatura/humidade e ao dispositivo móvel, através de Wi-Fi (sendo possível utilizar ligação Ethernet para conectar o microcontrolador à Internet). O seu objetivo consiste em utilizar este conjunto de componentes para, através de comandos de voz e/ou de comandos executados por botões virtuais, criados na aplicação openHAB, mostrar uma prova de conceito.

O openHAB irá aceder a dados partilhados por cada componente que serão, de seguida, descritos individualmente:

- Sensor de temperatura e humidade, cujos dados poderão ser acedidos e exibidos na aplicação, e servem apenas para consulta em tempo real.
- Módulos Shelly, que serão utilizados para o controlo das funcionalidades dos estores (subir e descer) e para ligar/desligar tomadas inteligentes.
- Lâmpadas LEDs, com cores específicas, que servirão o propósito de simular ações de subir e descer o motor de estores e ligar e desligar tomadas inteligentes.
- Um *Router*, que servirá o propósito de permitir a interligação, comunicação e interação entre os componentes envolvidos neste projeto.

Considerando que, através desta configuração seria apenas possível controlar os dispositivos na mesma rede, foram aproveitados os servidores disponibilizados pela própria openHAB Foundation, para que fosse possível aceder em segurança, e sem a necessidade de criação de redes VPN complexas, ou da abertura de portas (*port forwarding)* no *router* utilizado para ligar o Raspberry à Internet, de maneira totalmente remota. Através do *smartphone* é então possível aceder à aplicação openHAB, onde são desempenhadas as funcionalidades listadas em relação a *Smart Home*, através de comandos manuais ou comandos de voz. O diagrama a seguir ilustra a arquitetura do modelo proposto.

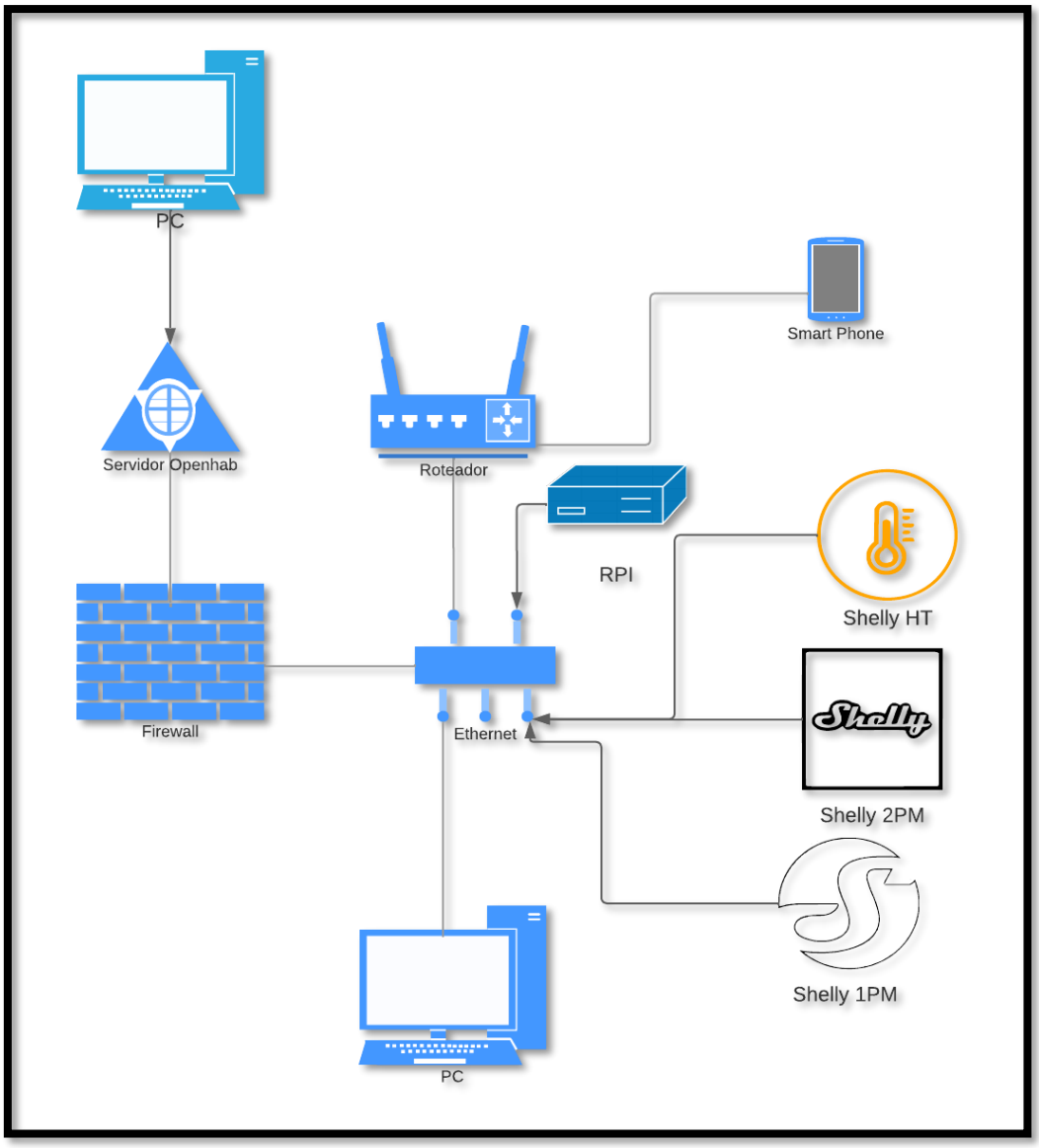


Figura 12- ilustração do modelo proposto

Fonte: autores do trabalho

3.2 Descrição do Funcionamento

3.2.1 Sensor de temperatura e humidade

O sensor de temperatura/humidade sem fios que se pretendia utilizar era o *Wi-Fi ZigBee - Sonoff*, no entanto, devido a constrangimentos monetários, tal não foi possível. Alternativamente, foi adquirido um sensor H&T da marca Shelly (ver 2.6.3 Sensor de temperatura/humidade), para que, a partir da respetiva aplicação, se pudessem configurar os dispositivos Shelly.

Para conectar o dispositivo Shelly a uma rede, basta pressionar o botão localizado ao lado da pilha, colocando assim o Shelly em modo de busca por redes Wi-Fi disponíveis para

conexão. Após o dispositivo se conectar à rede Wi-Fi fornecida pelo router utilizado no projeto, este é integrado à aplicação openHAB por meio de um *binding* específico.

Na aplicação Shelly, é necessário configurar o endereço IP e a porta de comunicação para estabelecer a ligação com a aplicação openHab. Essa configuração permite que a aplicação openHAB se comunique com o Shelly H&T e obtenha os dados de temperatura e humidade fornecidos pelo dispositivo. Esses dados são depois utilizados para visualização na aplicação openHAB, permitindo a monitorização precisa da temperatura e humidade de certa divisão.

3.2.2 Sensores Shelly 1PM e 2PM

No âmbito do nosso projeto, foi feita uma adaptação do Shelly 1PM para desempenhar o papel de botão de acionamento das tomadas inteligentes, enquanto o Shelly 2PM foi utilizado para controlar a subida e descida dos estores. Esta escolha foi motivada pela capacidade do Shelly 2PM de oferecer duas portas de saída, o que permite controlar o movimento dos estores de maneira conveniente.

A escolha de utilizar o Shelly 1PM como botão de acionamento das tomadas inteligentes foi baseada na sua versatilidade e capacidade de controlo. O Shelly 1PM é um dispositivo compacto e de fácil integração, que oferece uma porta de saída que pode ser configurada para controlar diferentes tipos de cargas elétricas, incluindo tomadas. Ao adaptar o Shelly 1PM para essa função, aproveitamos a sua capacidade de atuar como um interruptor remoto, permitindo ligar e desligar as tomadas remotamente.

3.2.3 Sistema central de processamento

O sistema central de processamento escolhido foi o microcontrolador Raspberry Pi 400, um controlador de alta capacidade de processamento, portabilidade e versatilidade, ideal para o desenvolvimento de aplicações *Smart Home*.

Ao utilizar o Raspberry Pi como controlador central e após a instalação do openHabian, um sistema operativo de automação residencial de código aberto, cujo objetivo é ajudar a automatização de um ambiente doméstico, foi possível desenvolver um sistema integrado para *smartphone* que permite a comunicação entre todos os dispositivos Shelly, fornecendo assim controlo remoto e automatizado de estores, tomadas inteligentes, e controlo de temperatura e humidade.

3.2.4 Aplicação de Controlo

O aplicativo de controlo desenvolvido em openHab proporciona uma interface intuitiva e eficiente para gerir diversos dispositivos de uma casa, incluindo estores, tomadas inteligentes e sensores de temperatura e humidade. Em seguida, será apresentado um guia passo a passo sobre a configuração e o funcionamento da aplicação.

Configuração de Acesso:

- Após a instalação do openHab no Raspberry Pi, é configurado o acesso à aplicação, o que envolve definir um nome de usuário e senha para proteger o acesso aos dispositivos e dados da casa.
- Além disso, é necessário configurar a integração com serviços de nuvem, permitindo o acesso remoto aos dispositivos através da internet.

Adição de Dispositivos:

- Para os estores, tomadas inteligentes e sensor de temperatura/humidade, os dispositivos são conectados ao Raspberry Pi através de uma rede sem fios e protocolos de comunicação específicos, como Zigbee ou *Shelly bindings.*

Configuração das Funções:

- Após adicionar os dispositivos ao sistema, são configuradas as suas funções específicas.
- Para os estores, foram programados os comandos de abertura e fecho na aplicação e por comandos de voz.
- Para as tomadas inteligentes, foi programada a função de ligar/desligar.
- Os sensores de temperatura e humidade foram configurados para atualizar as leituras em intervalos regulares e com variações de temperatura e humidade de 0,5 valores, permitindo a obtenção de dados em tempo real.

Funcionamento da Aplicação:

- Com a configuração concluída, o aplicativo de controlo openHab oferece uma interface amigável para gerir os dispositivos.
- Através do aplicativo, é possível abrir e fechar os estores com um simples clique, controlar as tomadas inteligentes, monitorizar a temperatura e humidade em diferentes ambientes da casa.

3.3 Desenvolvimento do modelo proposto

3.3.1 Construção da maquete

Durante a realização deste projeto, de modo a representar fisicamente a parte do trabalho relacionada ao desenvolvimento dos estores automatizados e das tomadas inteligentes, foi necessário desenvolver uma maquete em pequena escala que permitisse ilustrar o funcionamento da aplicação. Para esse propósito, foram utilizadas lâmpadas LED como uma representação simbólica dos estores automatizados. As lâmpadas LED encarregues de realizar o controlo dos estores e das tomadas inteligentes, estão conectadas aos módulos Shelly que, por sua vez, estão conectados a um controlador e, por fim, a uma tomada que fornece a energia necessária para o funcionamento dos dispositivos. Para os estores, quando a luz verde acende, representa a abertura dos estores, enquanto que a luz vermelha indica o fechamento dos mesmos. Estes podem ainda ser controlados manualmente através de um interruptor que permite testar a subida e descida dos estores. Relativamente às tomadas inteligentes, quando a luz branca acende, quer dizer que a tomada está ligada, e quando está desligada, não recebe energia.

3.3.2 Implementação Física do Protótipo

Podemos verificar, na figura seguinte, o resultado final do desenvolvimento do protótipo quando este se encontra completamente desligado, ou enquanto espera por um comando, verificando-se assim as ligações entre o equipamento.

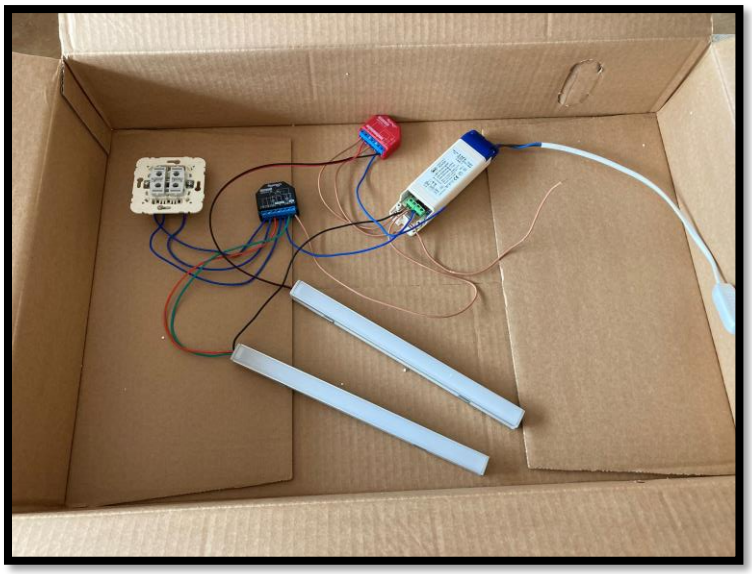


Figura 13 - representação física da maquete

Fonte: autores do trabalho

Em seguida, podemos verificar o seu funcionamento quando este recebe um comando, com a luz verde a simbolizar a subida dos estores e o vermelho a representar a descida.

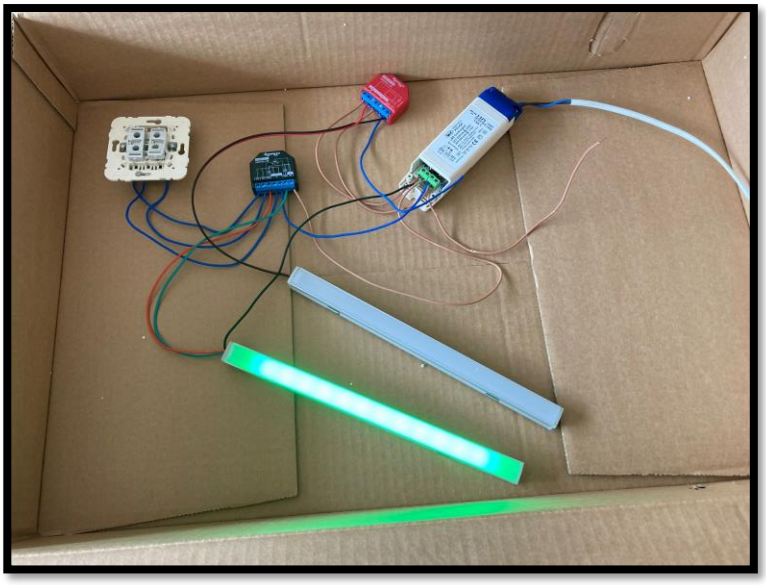


Figura 14 - Representação simbólica da subida dos estores

Fonte: autores do trabalho

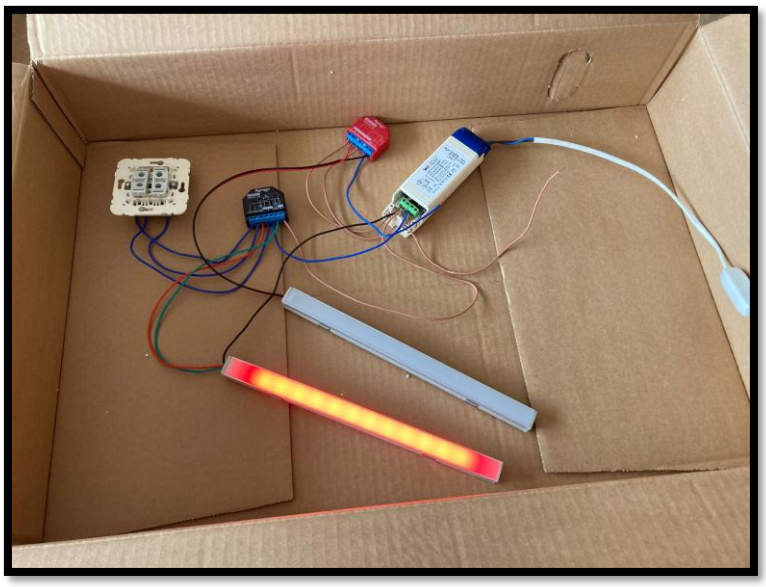


Figura 15- Representação simbólica da descida dos estores

Fonte: autores do trabalho

3.3.2.1 Central de processamento

3.2.2.1.1 Sistema operativo

O openHABian é um sistema operativo *open source* (de código aberto), gratuito, cujo objetivo é auxiliar a automatização de um ambiente doméstico. Este sistema operativo tem

como inspiração o nome da própria plataforma openHAB, e toma por base o sistema operativo Raspbian. O openHABian cinge-se a correr *scripts* por cima do Raspbian, que instalam *packages* e configuram o sistema de maneira específica e otimizada para hospedar a plataforma openHAB. Neste projeto, o Raspberry Pi 400 é utilizado para alojar o openHAB, visto ser o microcontrolador escolhido para a realização do protótipo desenvolvido.

O Raspberry possui uma ranhura de inserção de cartões SD e será, preferencialmente, num cartão SD que se armazenará o sistema operativo selecionado, de modo a evitar potenciais problemas, nomeadamente questões relativas à segurança, que muitas vezes existem em boots de sistemas operativos em pens, por exemplo.

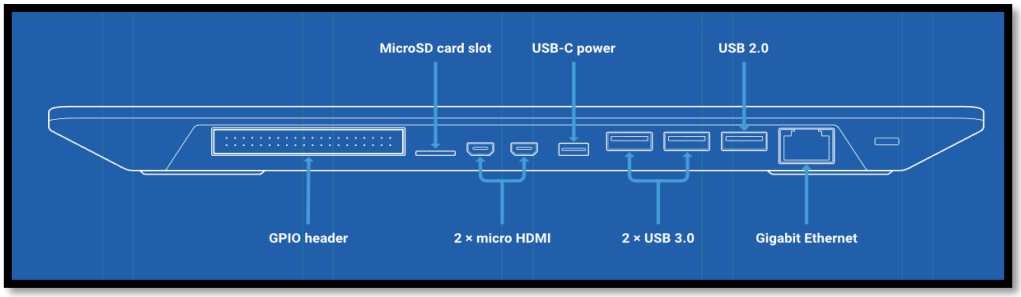


Figura 16 - Perspetiva das entradas/saídas do Raspberry Pi 400

Fonte: https://www.raspberrypi.com

Para instalar o *boot* do openHABian dispôs-se de um programa de criação de imagem de sistema operativo utilizado para a instalação do mesmo. Seguidamente, inseriu-se o cartão SD no computador onde foi executado o programa de criação de imagens (recomenda-se a aplicação oficial Raspberry Pi Imager, por motivos de acessibilidade, intuição, bom funcionamento e eficácia, através do *link* Raspberry Pi OS – Raspberry Pi). Fez-se o *download* da imagem do openHABian (neste caso o 3), através do endereço Download openHAB | openHAB para que, posteriormente, se pudesse criar a imagem de openHABian no cartão SD que seria inserido no Raspberry Pi.

Figura 17 - Website para download da imagem de openHABian

Fonte: https://www.openhab.org/download

Executando a aplicação, carregou-se na opção “CHOOSE STORAGE” (debaixo de “Storage”) e selecionou-se o cartão SD, além disso, carregou-se na opção “CHOOSE OS” (debaixo de “Operating System”), escolheu-se “Use custom” e, finalmente o ficheiro openHABian descarregado anteriormente.

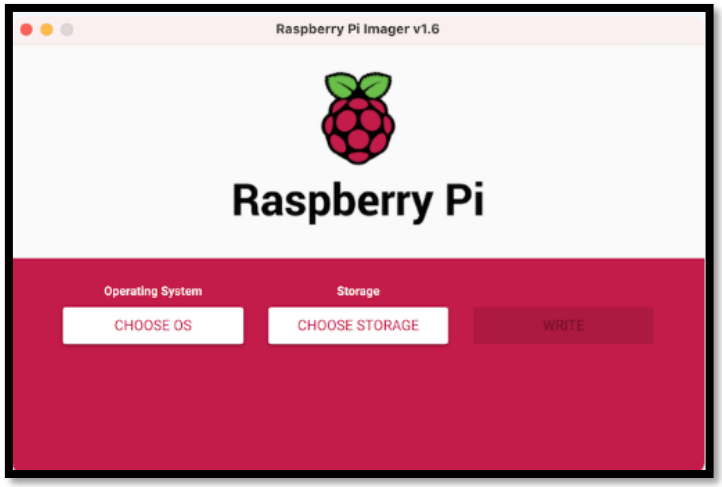


Figura 18- Raspberry Pi Imager

Fonte: autores do trabalho

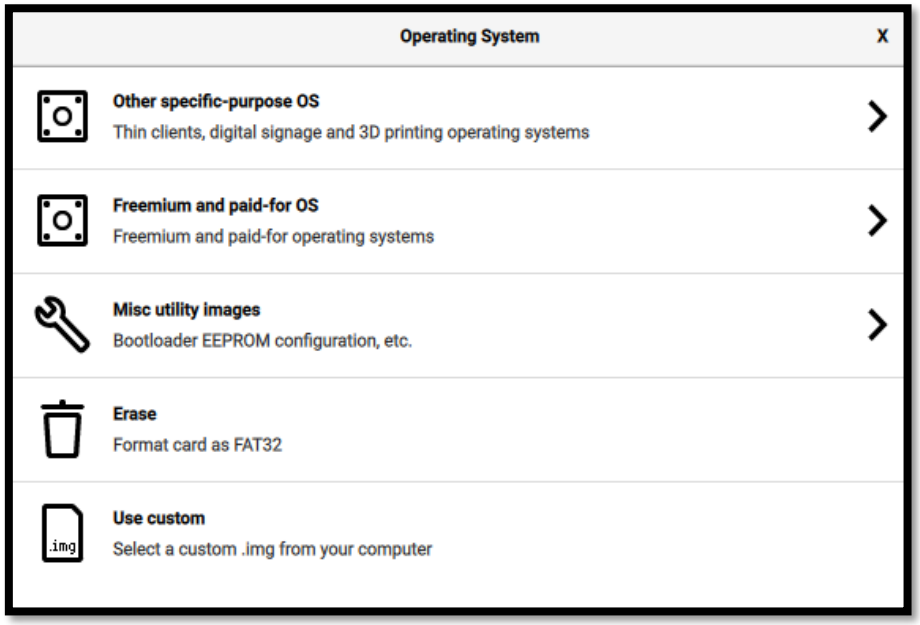


Figura 19- Opções de instalação do SO

Fonte: autores do trabalho

Após a criação da imagem do openHABian 3.4.4 (recomenda-se, aliás, a instalação desta iteração específica por ser, aquando do desenvolvimento deste protótipo, a última iteração estável de openHAB 3 lançada), retirou-se o cartão SD do computador, inseriu-se no

Raspberry, ligou-se o Raspberry através do cabo de alimentação e conectou-se o dispositivo à Internet através de Ethernet, para que este fizesse a configuração inicial do sistema operativo (mais tarde configurou-se a conexão do dispositivo à Internet através de *Wi-Fi*, visto este possui um módulo de *Wi-Fi* integrado). A conexão à Internet foi necessária, visto que o openHABian necessita de descarregar os *packages* e correr os *scripts* necessários para configurar o sistema operativo e otimizar o próprio para efeitos de automação. Adicionalmente, viu-se essencial vigiar e acompanhar atentamente o terminal virtual do microcontrolador e verificar qualquer código de erro que pudesse ocorrer na configuração/instalação inicial do sistema operativo.

Visto que o Raspberry não possui um ecrã próprio, fator que implica a necessidade de ligar o microcontrolador a um ecrã externo, quer através de uma das portas HDMI existentes no próprio computador, quer através de uma ligação SSH. Será então utilizada a aplicação PuTTY (disponível oficialmente para Windows e algumas versões de Unix) de modo a facilitar a configuração e acesso aos ficheiros e logs do Pi.

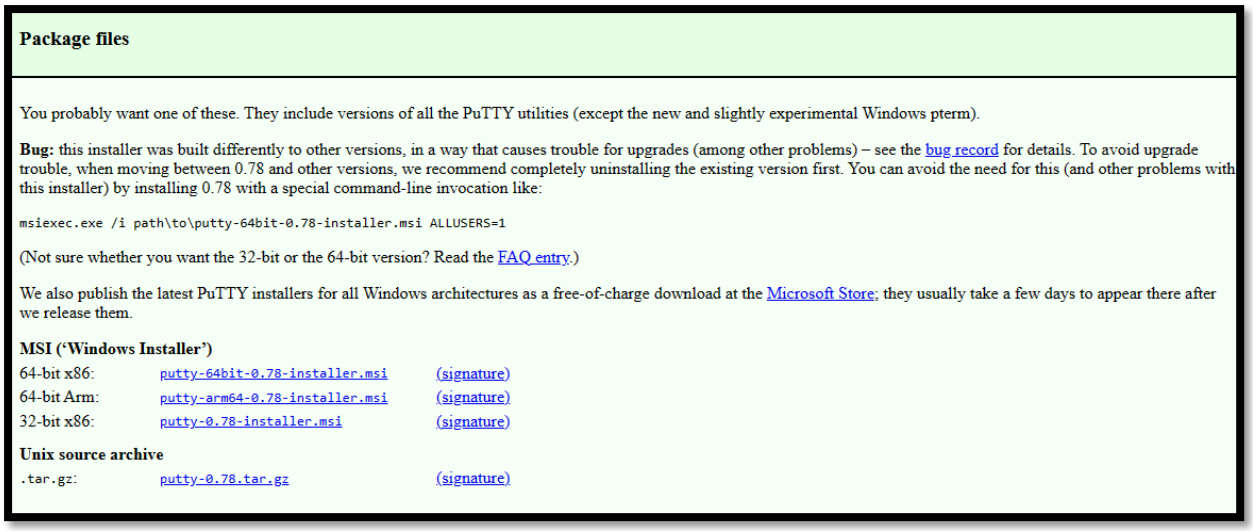
Após esperar aproximadamente 15 minutos de instalação, tornou-se possível começar a trabalhar no openHABian. Recomenda-se, finalmente, utilizar o comando *sudo openhabian-config* para aceder às configurações e atualizar os *packages*, à semelhança do que foi concretizado pelo grupo.

3.2.2.1.2 Cliente SSH

Figura 20 - PuTTY

Fonte: https://pensandolinux.com.br

Para utilizar a aplicação PuTTY, é necessário dirigir-se ao *website* Download PuTTY: latest release (0.78) (greenend.org.uk) e escolher a opção relativa ao sistema operativo pretendido. Neste caso será instalado a versão para Windows 10 Home de 64 bits, visto ser o sistema operativo hospedeiro presente nas máquinas dos constituintes do trabalho.



Package files

You probably want one of these. They include versions of all the PuTTY utilities (except the new and slightly experimental Windows pterm).

Bug: this installer was built differently to other versions, in a way that causes trouble for upgrades (among other problems) – see the bug record for details. To avoid upgrade trouble, when moving between 0.78 and other versions, we recommend completely uninstalling the existing version first. You can avoid the need for this (and other problems with this installer) by installing 0.78 with a special command-line invocation like:

```
msiexec.exe /i path\to\putty-64bit-0.78-installer.msi ALLUSERS=1
```

(Not sure whether you want the 32-bit or the 64-bit version? Read the FAQ entry.)

We also publish the latest PuTTY installers for all Windows architectures as a free-of-charge download at the Microsoft Store; they usually take a few days to appear there after we release them.

MSI ('Windows Installer')

64-bit x86:	putty-64bit-0.78-installer.msi	(signature)
64-bit Arm:	putty-arm64-0.78-installer.msi	(signature)
32-bit x86:	putty-0.78-installer.msi	(signature)

Unix source archive

.tar.gz:	putty-0.78.tar.gz	(signature)

Figura 21 - Página de dowload do PuTTY

Fonte: https://www.chiark.greenend.org.uk

Após descarregar e correr o programa, acedeu-se a "Session" e inseriu-se o nome de utilizador openhabian no campo "Host Name (or IP address)". Posteriormente recomenda-se (não sendo, no entanto, imprescindível) criar um nome de sessão personalizado para evitar ter de inserir os dados de autenticação cada vez que se pretender usar a aplicação PuTTY para aceder ao Raspberry Pi. Para tal, após se inserir o nome de utilizador no campo correspondente, carrega-se no campo "Saved Sessions" e insere-se um nome, o qual é de livre escolha. Neste caso, criou-se uma seção com o nome "Raspberry Pi".

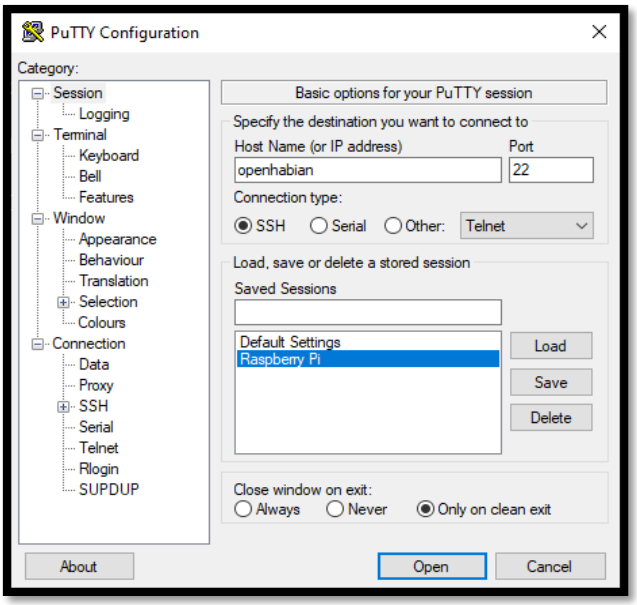


Figura 22 - Configuração do PuTTY

Fonte: autores do trabalho

Carregando na opção “Open” e, o terminal Linux do Raspberry abre e permite a inserção dos das credenciais de acesso definidas para o openHabian que, após inserção, fornece o controlo do dispositivo ao utilizador.

3.2.2.1.3 Aplicação de controlo

A aplicação de controlo é, como já referido, a plataforma openHAB. Após a instalação e configuração do sistema operativo openHABian, torna-se agora possível aceder à interface *web* da plataforma e, consequentemente, a configurações de alto nível da aplicação que se pretende desenvolver. Os botões com as funcionalidades que se irão desenvolver encontrar-se-ão, ainda antes do desenvolvimento da aplicação móvel propriamente dita, disponíveis nesta interface *web*, a qual permite o acesso a privilégios de administração relacionados com o desenvolvimento da aplicação.

3.3.2.2 Sistema de sensores (Shellys)

Durante a montagem da maquete, realizou-se a conexão dos fios e ligou-se a ficha a uma tomada para fornecer energia aos dispositivos Shelly. Após a ativação do Shelly 1PM e 2PM, uma luz indicadora acendeu confirmando que o relé estava ligado. A partir desse momento, foi possível utilizar a aplicação móvel (Shelly) para adicionar o dispositivo à rede Wi-Fi, seguidamente foi executado o mesmo procedimento para configurar o sensor H&T. Além disso, os endereços IP dos Shellys foram configurados como estáticos por meio da aplicação Shelly, evitando assim possíveis alterações de IPs que poderiam ocorrer com o desligamento do relé ou do dispositivo de medição de temperatura e humidade. Com o auxílio do binding do openHAB, foi possível detectar os sensores na rede e proceder à respetiva configuração na plataforma. Para realizar as configurações dos dispositivos utilizou-se o VSCode, que permitiu programar as "*Things*" (dispositivos) em java, de maneira que estes conseguissem comunicar com a plataforma e aplicação openHab.

```
// Shelly dos Estores
Thing shelly:shellyplus2pm-roller:b8d61a896c34 "Shelly WindowShutter" @ "Bed Room" [ deviceIp="192.168.1.207", userId="", password=""]

// Shelly da tomada inteligente
Thing shelly:shellyplus1pm:3c6105729b9c "Shelly SmartPlug" @ "Bed Room" [ deviceIp="192.168.1.206", userId="", password=""]

// Shelly sensor de temperatura e humidade
Thing shelly:shellyht:c9a763 "Shelly Hygrometer" @ "Bed Room" [ deviceIp="192.168.1.208", lowBattery=15, eventCoIoT=true, userId="", password=""]
```

Através do código apresentado, é possível verificar que foi criada uma variável do tipo shelly para cada um dos componentes, com a respetiva configuração do dispositivo, tipo de dispositivo, *tag* indicativa do local da casa e configurações de acesso e programação dos sensores, como o ip dos sensores e credenciais de acesso, que neste caso não se encontram definidas.

Com a plataforma openHab (administração) procedemos à verificação da criação das *Things* referidas e verificamos também o estado dos dispositivos.

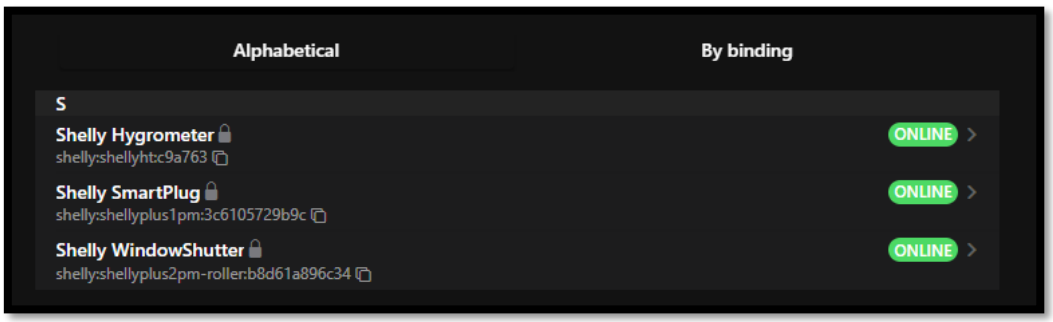


Figura 23 - UI openHab Things

Fonte: autores do trabalho

Após a criação das *Things*, foi necessário associar estas a *Items*, que permitem que sejam atribuídas e alteradas funcionalidades do dispositivo através de funções. No fundo estes funcionam como uma variável do tipo *Thing*, que dependendo do tipo de componente (ex: *switch*, *roller shutter*) terá funções associadas que permitem manipular a função dos sensores.

```
// Item da tomada inteligente
Switch PlugSwitch "Tomadas" <switch> {ga="Outlet", channel="shelly:shellyplus1pm:3c6105729b9c:relay#output"}

// Item dos Estores
Rollershutter BlindsRollerShutter "Estores" <blinds> {ga="Shutter" [checkState=true], channel="shelly:shellyplus2pm-roller:b8d61a896c34:roller#control"}

// Sensor TH
Group THSensor {ga="ClimateSensor" [useFahrenheit=false]}

// Item do sensor de Temperature
Number TemperatureSensor "Temperatura" <temperature> (THSensor) {ga="TemperatureSensor", channel="shelly:shellyht:c9a763:sensors#temperature" }

// Item do sensor de Humidade
Number HumiditySensor "Humidade" <humidity> (THSensor) {ga="HumiditySensor", channel="shelly:shellyht:c9a763:sensors#humidity" }
```

Através do código apresentado, é possível verificar que foram criadas variáveis de vários tipos diferentes. O primeiro item refere-se à tomada inteligente, representada pelo switch "PlugSwitch". O ícone "<switch>" indica que se trata de um interruptor. A configuração "ga" especifica a categoria do dispositivo (Outlet) para integração com assistentes de voz, e o canal "shelly:shellyplus1pm:3c6105729b9c:relay#output" estabelece a conexão com o relé de saída do dispositivo Shelly Plus 1PM.

O segundo item é relacionado aos estores (blinds), representado pelo "Rollershutter" chamado "BlindsRollerShutter". O ícone "<blinds>" indica que se trata de um estore. A configuração "ga" especifica que é um dispositivo de controle de estores (Shutter) com a opção "checkState" ativada para verificar o estado atual. O canal "shelly:shellyplus2pm-roller:b8d61a896c34:roller#control" estabelece a conexão com o controlo do estore do dispositivo Shelly Plus 2PM.

Os três itens seguintes referem-se aos sensores de temperatura e humidade. O item "TemperatureSensor" é um número que representa a temperatura, com o ícone "<temperature>". O item "HumiditySensor" é um número que representa a humidade, com o ícone "<humidity>". Ambos são associados ao grupo (grupo de itens) "THSensor" para

organização. As configurações "ga" especificam as categorias do dispositivo (TemperatureSensor e HumiditySensor), e os canais "shelly:shellyht:c9a763:sensors#temperature" e "shelly:shellyht:c9a763:sensors#humidity" estabelecem a conexão com os sensores de temperatura e umidade do dispositivo Shelly HT. Através da plataforma openHab (administração) procedemos à verificação da criação dos itens referidos.

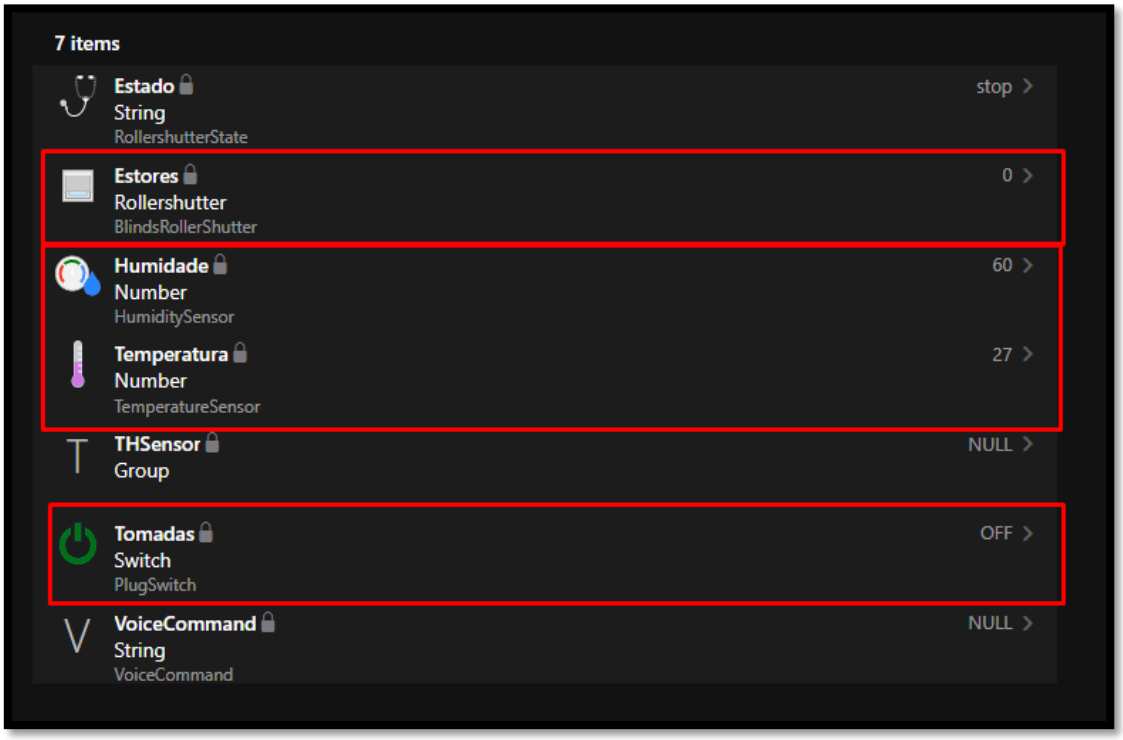


Figura 24 - UI openHab Items

Fonte: autores do trabalho

3.3.2.2 Aplicação de controlo

Após a criação da maquete e configuração dos dispositivos (*things e items*) no VSCode. Será necessário criar regras (funções) para o correto funcionamento dos estores, visto que, as funcionalidades de um motor não são as mesmas dos LEDs.

```
var Timer myTimer = null
rule "Blinds"

when
Item BlindsRollerShutter received command UP or Item BlindsRollerShutter received command DOWN
then
	if(myTimer !== null) {
		myTimer?.cancel;
		myTimer = null;
		logInfo("Timer", "Timer reset");
	}
	myTimer = createTimer(now.plusSeconds(5), [ |
		BlindsRollerShutter.sendCommand(STOP);
		 logInfo("Timer", "Timer activated");
	])
end
```

Através do código apresentado, é possível verificar que foi criada uma regra ou função chamada “Blinds”, que é ativada após a ocorrência do envio de um comando “UP” ou “DOWN” no item BlindsRollerShutter, ou seja, nos estores.

Dentro desta regra, há uma verificação do objeto Timer chamado "myTimer", criado como variável global. Se o timer já estiver em execução, ou seja, se os estores estiverem a descer ou a subir, então este é cancelado e o valor é resetado. Isso é feito para evitar que múltiplos comandos de subida e descida sejam criados simultaneamente e alterem o tempo de subida e descida dos estores, que é simulado devido ao uso de LEDs em vez do motor bi-direcional.

Em seguida, um novo timer é criado utilizando a função createTimer e agendado para ser executado após 5 segundos (now.plusSeconds(5)). Quando o timer é ativado, o comando STOP é enviado para o item BlindsRollerShutter, interrompendo o movimento dos estores. Além disso, é registrado um log com informações, acessível através do terminal do Raspberry Pi, indicando que o timer foi ativado.

Esta regra é útil para controlar o movimento dos estores de forma automatizada, garantindo que, após um comando de subir (UP) ou descer (DOWN), os estores parem de se mover após 5 segundos, evitando assim que ocorra algum problema durante o processo.

Por último, para implementar os dispositivos criados numa aplicação android foi criado o seguinte sitemap onde foram associados os dispositivos da app.

```
sitemap teste label="Projeto Final" {
        Frame label="Quarto" {
        Default item=RollershutterState
        Default item=BlindsRollerShutter
        Default item=PlugSwitch icon="poweroutlet"
        Default item=TemperatureSensor
        Default item=HumiditySensor
        }
}
```

Visto que os itens criados têm uma estrutura definida e icons associados, a criação do sitemap é bastante fácil e intuitiva como pudemos ver pelo código fonte. O resultado final foi o seguinte.

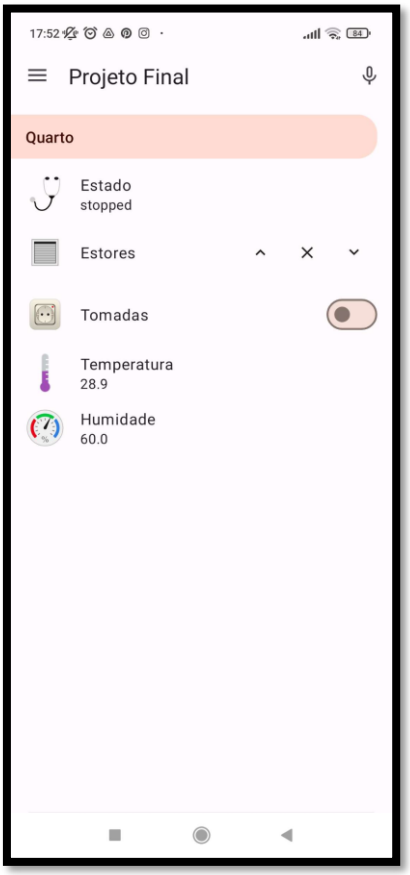


Figura 25- Aplicação openHab

Fonte: autores do trabalho

3.3.2.2.1 Controlo por voz

Em relação aos comandos de voz da aplicação, foram desenvolvidos 2 tipos de implantação, uma para a própria aplicação utilizando o reconhecimento de voz da Google e outra para o Google Assistant da Google Home.

Visto que o openHab possui integração de voz do google assistant (API) para aplicações móveis, foi então possível realizar integração de maneira simplificada tanto na aplicação como no Google Home.

Na aplicação desenvolvida no VSCode, primeiramente foi criada a variável do tipo String, responsável por captar os comandos de voz do utilizador (telemóvel).

```
// Voice commands
String VoiceCommand "VoiceCommand"
```

Após a variável ter sido definida, é então criada a regra (função) que executa as funcionalidades desenvolvidas no projeto, de acordo com os valores recebidos.

```
// Voice commands
String VoiceCommand "VoiceCommand"
var String command
rule "VoiceControl"
when
    Item VoiceCommand changed
then
    command = VoiceCommand.state.toString.toLowerCase;
    createTimer(now.plusSeconds(1), [|
    switch (command) {
        case "abrir estores",
        case "subir estores": {
            logInfo("Executing ", "VoiceCommand executing " + command)
            BlindsRollerShutter.sendCommand(UP)
        }
        case "descer estores",
        case "fechar estores": {
            logInfo("Executing ", "VoiceCommand executing " + command)
            BlindsRollerShutter.sendCommand(DOWN)
        }
        case "parar estores": {
            logInfo("Executing ", "VoiceCommand executing " + command)
            BlindsRollerShutter.sendCommand(STOP)
        }
        case "ligar tomadas": {
            logInfo("Executing ", "VoiceCommand executing " + command)
            PlugSwitch.sendCommand(ON)
        }
         case "desligar tomadas": {
            logInfo("Executing ", "VoiceCommand executing " + command)
            PlugSwitch.sendCommand(OFF)
        }
        default: {
             logInfo("Error", "VoiceCommand " + command + " doesn't exist")
        }
    }
    ])
end
```

O código ilustrado, consiste em receber o conteúdo captado pelo controlador de voz e executar ações com base nesse conteúdo. Os comandos que são possíveis utilizar são os seguintes.

1. **“Abrir estores” ou “subir estores”-** Executa a função de abertura dos estores.
2. **“Fechar estores” ou “descer estores”-** Executa a função para fechar os estores.
3. **“Parar estores”-** Para os estores e cancela as funções de subir ou descer estores.
4. **“Ligar tomadas”-** Liga as tomadas inteligentes.
5. **“Desligar tomadas” -** Desliga as tomadas inteligentes.

Visto que o sensor de temperatura e humidade tem os seus valores em exibição na aplicação, não fez sentido desenvolver comandos de voz para obter os respectivos valores.

Relativamente aos comandos de voz do Google Assistant, para estes serem integrados com a aplicação openHab, foi necessário a configuração do Google Home e dos respectivos dispositivos que comunicam com o Google Assistant.

Primeiramente, os items e utilizados para controlador de voz são definidos com a tag “ga”(google assistant) da API da Google e o respetivo equipamento, como mostra o exemplo.

```
// Item da tomada inteligente
Switch PlugSwitch "Tomadas" <switch> {ga="Outlet", channel="shelly:shellyplus1pm:3c6105729b9c:relay#output"}
```

Figura 26 - Definição da implementação do ga

Para configurar e usufruir dos comandos de voz para os dispositivos associados ao google assistant, basta seguir os seguintes passos:

Configurar o Google Home

1. Certifique-se de ter a aplicação do Google Home instalado no seu smartphone.
2. Abra a aplicação do Google Home aos dispositivos.
3. Se não possuir uma casa criada, crie uma (é possível ignorar a maioria dos passos)
4. Clique na opção "Adicionar dispositivo" ou "Configurar dispositivo".
5. Escolha a opção "Funciona com o Google".
6. Selecione "openHAB" na lista de serviços disponíveis.
7. Faça login na sua conta do openHAB quando solicitado .
8. Volte para a aplicação e selecione novamente "openHAB" na lista de serviços.
9. Clique em Allow para permitir a autenticação na conta openHab
10. Após a conexão ser bem sucedida, adicione os dispositivos que aparecem no ecrã a uma divisão da sua casa e está feito.

Após a instalação será possível pedir ao google assistant para executar os comandos da aplicação, como subir e descer estores, ligar ou desligar as tomadas inteligentes ou fornecer a temperatura e humidade da divisão.

Alguns dos comandos de voz disponíveis encontram-se listados a seguir.

1. **“Abre os estores” ou “sobe os estores”-** Executa a função de abertura dos estores.
2. **“Fecha os estores” ou “desce os estores”-** Executa a função para fechar os estores.
3. **“Para os estores”-** Para os estores e cancela as funções de subir ou descer estores.
4. **“Liga as tomadas”-** Liga as tomadas inteligentes.
5. **“Desliga as tomadas” -** Desliga as tomadas inteligentes.
6. **“Diz-me a temperatura do quarto”** - Obtém a temperatura do quarto
7. **“Diz-me a humidade do quarto”** - Obtém a humidade do quarto

4. Testes e análises de resultados

4.1 Teste do protótipo

- **Teste de desempenho:** Avalia o desempenho do sistema em termos de tempo de resposta, estabilidade e confiabilidade das operações realizadas e verifica se as ações de acionamento das tomadas e movimentação dos estores são executadas de forma rápida e consistente.
- **Teste de comunicação entre os dispositivos Shelly e o openHAB:** Verifica se os dispositivos Shelly estão a comunicar corretamente com o openHAB, transmitindo informações de estado, comandos de controlo e se recebem atualizações das configurações. Isso pode ser feito verificando se as informações de medição de temperatura e/ou humidade estão a ser atualizadas e se o estado das *Things* estão online no openHAB conforme esperado.
- **Teste de controlo dos estores:** Verifica se o Shelly 2PM está a enviar corretamente os comandos de subida e descida para controlar os estores, isto é, se os estores se movimentam de acordo com os comandos enviados pelo Shelly 2PM e se param efetivamente nos pontos desejados.
- **Teste de confiabilidade da comunicação:** Executar testes de longa duração para verificar se a comunicação entre os dispositivos Shelly e o openHAB é estável e confiável ao longo do tempo e se não ocorrem falhas na transmissão de dados ou na execução dos comandos durante um período prolongado de uso.
- **Teste de funcionamento dos LEDs:** Verificar se todos os LEDs estão a funcionar corretamente, emitindo a cor esperada bem como se os comandos no openHAB para ligar, desligar os LEDs respondem conforme o esperado.
- **Teste de posicionamento dos estores:** Verificar se é possível posicionar os estores em diferentes níveis de abertura. Acione os comandos no openHAB para controlar a posição dos estores e verifique se eles param no ponto desejado, permitindo um ajuste preciso da abertura.

4.2 Análise de resultados

Apesar de, duma forma geral, os resultados dos nossos testes terem sido positivos, deparamo-nos com alguns erros durante a testagem do nosso protótipo, como explicamos na tabela abaixo.

ERRO:COMM	Este tipo de erro sugere que houve um problema de comunicação, que requer alguma investigação, uma vez que se pode dever à conexão física (cabos e extensões), à conexão de rede (configurações do IP), problemas derivados dos mecanismos de comunicação, ou até relacionados com a alimentação elétrica.
ONLINE	ONLINE indica que o dispositivo pode ser acedido e está a responder corretamente. Os dispositivos alimentados por bateria também permanecem ONLINE quando estão no modo de suspensão. A ligação tem um temporizador *watchdog* integrado que supervisiona o dispositivo. A *Thing* muda para o estado OFFLINE quando ocorre algum tipo de erro de comunicação.
OFFLINE	A comunicação com o dispositivo falhou. Verifique o estado da *Thing* na UI e o registo do openHAB para obter uma indicação do erro. Tente reiniciar o openHab ou apagar e voltar a descobrir a *Thing*.

Tabela 2- Tipos de erros encontrados

Fonte: https://www.openhab.org/addons/bindings/shelly/

Seguem abaixo as análises mais pormenorizadas de cada teste realizado:

- **Teste de desempenho:** O sistema demonstrou um desempenho satisfatório, com tempo de resposta rápido (5 segundos) e operações estáveis. As ações de acionamento das tomadas inteligentes e movimentação dos estores foram executadas de forma consistente e dentro dos padrões esperados.
- **Teste de comunicação entre os dispositivos Shelly e o openHAB:** A comunicação entre os dispositivos Shelly e o openHAB foi estabelecida com sucesso. As informações de medição de temperatura e/ou humidade foram transmitidas corretamente para o openHAB, permitindo o acompanhamento em tempo real desses dados.

- **Teste de controlo dos estores:** O Shelly 2PM demonstrou eficiência no envio dos comandos de subida e descida dos estores (representados pelas LED). Os estores responderam de forma adequada, movendo-se de acordo com os comandos enviados, e.g., quando solicitamos ao Google Assistant que subisse os estores.
- **Teste de confiabilidade da comunicação:** Durante os testes de longa duração (4 horas), a comunicação entre os dispositivos Shelly e o openHAB manteve-se estável e confiável. Não foram observadas falhas na transmissão de dados ou na execução dos comandos ao longo do período de teste.
- **Teste de funcionamento dos LEDs:** Todos os LEDs funcionaram corretamente, emitindo a luz de acordo com a cor e "ordem" esperada, ou seja, verde quando o comando fosse de subir/abrir e vermelho quando o comando fosse de descer/fechar.
- **Teste de posicionamento dos estores:** tendo em conta que não foi possível utilizar o motor bi-direcional, este teste acabou por não ser bem-sucedido. O Shelly 2PM é capaz de enviar comandos de subida e descida para os estores, mas a utilização dos LEDs impossibilitou a realização da sincronização do dispositivo, visto que este não possui percentagens e, portanto, não foi possível obter dados acerca do posicionamento específico dos estores.

Como se verifica, os resultados obtidos através do protótipo foram satisfatórios, sendo possível efetuar os vários comandos implementados. Com base nessa análise, os resultados dos testes indicam que o protótipo do sistema de automação residencial utilizando os dispositivos Shelly e o openHAB funciona conforme o esperado, apresentando bom desempenho, comunicação confiável e controlo efetivo dos dispositivos, sendo capaz de proporcionar uma experiência com qualidade aos seus utilizadores.

Limitações e/ou problemas

À semelhança daquilo que é comum na realização de qualquer projeto, o grupo deparou-se com alguns obstáculos que condicionaram tanto o estado geral do protótipo, como alguns aspetos mais particulares. Foram necessárias alterações face aos planos iniciais para poder, dentro das condicionantes apresentadas, concretizar a ideia de uma prova de conceito que mostrasse, numa escala reduzida e *low cost*, as potencialidades da implementação de estores automatizados. Deste modo, neste tópico da presente ata, serão colocados em discussão os problemas e as limitações ao qual o grupo foi exposto.

A primeira limitação encontrada no decorrer deste trabalho prático resultou do facto de existir uma falta de *stock* de Raspberry Pi a nível nacional, o que condicionou a aquisição atempada de um microcontrolador dessa série, que fosse simultaneamente acessível em termos financeiros, e que cumprisse o requisito de possuir 2 GB de memória RAM (ou mais), dadas as recomendações do openHAB ser uma aplicação que requer, no mínimo, essa quantidade de memória. Eventualmente, o grupo conseguiu localizar uma loja onde se vendia um modelo de Raspberry Pi que possuía 4 GB de RAM com um preço razoável.

O facto de o Raspberry Pi, o nosso microcontrolador, ser uma componente física, implicou que tivessem de ser pesquisadas, treinadas e utilizadas ferramentas alternativas e muitas vezes alheias ao conhecimento do grupo, para ultrapassar barreiras relativas ao deslocamento e à sua inconveniência subjacente. Para tal, foram implementados os hábitos de utilizar a aplicações de terceiros como o Parsec, cujo intuito seria o de obter acesso remoto ao computador que estaria conectado à mesma rede doméstica que o Raspberry Pi para, quer através da interface *web* e aplicação do VSCode do openHAB quer através da aplicação PuTTY, qualquer membro do grupo pudesse aceder, configurar e programar o Raspberry Pi ou o openHAB de maneira remota.

Após a obtenção do equipamento necessário ao desenvolvimento do protótipo, o grupo deparou-se com a imperatividade de utilizar o cartão SD de 16 GB fornecido com o próprio Raspberry Pi, não tendo conseguido utilizar, como seria preferencial, uma *pen* USB de 32 GB, pois a instalação do sistema operativo openHABian 3.4.4 no microcontrolador não era efetuada de maneira suave e fiável, o que iria comprometer, logicamente, a fiabilidade do *software* e, assim, do projeto como um todo. Para além desse fator, surgiu uma outra questão adjacente à instalação do sistema operativo, nomeadamente a necessidade de utilizar a aplicação oficial Raspberry Pi Imager para poder criar, através de um PC cujo sistema operativo hospedeiro fosse o Windows 10 Home de 64 bits (versão 22H2), a imagem de openHABian 3.4.4 no cartão

SD, onde se encontra, de momento, este sistema operativo instalado. O facto deu-se após a tentativa de criar a mesma imagem através da aplicação balenaEtcher a qual, por motivos alheios ao grupo, criava uma imagem com erros.

Outra limitação encontrada foi o facto de o PuTTY, aplicação de cliente SSH e *telnet* utilizada pelo grupo para aceder ao terminal Linux do sistema operativo openHABian, apenas permitir o acesso de maneira semi-remota, i.e., utilizando um computador externo ao próprio Raspberry, mas interno à rede local à qual esse estaria ligado. Deste modo, optámos por utilizar a aplicação Parsec para aceder remotamente a um computador presente na rede local do próprio Raspberry e poder, assim, aceder a configurações e outros aspetos mais técnicos do openHAB.

Uma das decisões difíceis, mas imprescindíveis para o bom funcionamento do projeto, foi a de sacrificar a implementação de um motor por um conjunto de luzes LED que, de maneira pré-planeada, se acendem ou apagam de acordo com a intenção de subir ou baixar. Neste sentido, a intenção das luzes LED é, portanto, a mesma que teria o estore em si, integrando-as na nossa prova de conceito, de forma prática e numa escala muito mais acessível e de dimensões reduzidas, mas ainda assim fiel ao que seria um efetivo estore elétrico automatizado. Uma vez que as ligações elétricas dos Shelly às luzes LED foram configuradas e programadas de maneira idêntica à de um motor de estore, com algumas modificações claro, o dito funcionaria da maneira pretendida. Contudo, o Shelly 2PM não se encontra calibrado por percentagens e logo não é possível obter a posição do estore, visto que o Shelly não se encontra ligado a um motor.

Em suma, um grande fator decisivo entre a montagem de Shellys ligados a luzes LED, ou de um motor verdadeiro ligado a um estore é a diferença acentuada nos preços, de aproximadamente 9€ no caso das LED e 70€ no caso do motor de estore.

Outra problemática com que nos viríamos a deparar no decorrer do trabalho, estaria relacionada com o sensor de temperatura e humidade. Após adquirirmos um sensor Sonoff com a capacidade de leitura dos dados de temperatura e humidade do ambiente ao seu redor, pôde-se constatar que o mesmo requer a aquisição adicional, de uma *bridge* oficial da mesma marca, que possuísse compatibilidade com o protocolo ZigBee. Deste modo,estando este custo adicional já fora do âmbito albergado pelo instituto universitário, o grupo incumbiu-se de adquirir a referida peça, para permitir conectar o sensor à aplicação openHAB através do *router* e, portanto, por *Wi-Fi*. Assim, o grupo foi obrigado a reiniciar uma nova procura por um sensor de humidade e temperatura com ligação *Wi-Fi*, que fosse economicamente viável, sem sacrificar a qualidade e funcionalidade desejada. Afortunadamente, o grupo conseguiu adquirir

um sensor H&T da marca Shelly, muito mais económico financeiramente, e igualmente eficaz no cumprimento do propósito pretendido.

Outro dos problemas com que nos deparámos foi com a programação dos comandos de voz dentro da aplicação openHAB, visto que as extensões disponíveis para TTS (*Text to Speech*) ou eram pagas ou não possuíam suporte para português. Acabámos por resolver o problema que encontrámos utilizando o controlo de voz do Google Assistant (Google Home).

Por fim, um dos maiores desafios enfrentados foi lidar com a limitação do distanciamento físico entre os componentes do grupo e a necessidade de interação física para atingir os objetivos estabelecidos, baseando-se o nosso projeto no acesso remoto aos dispositivos, através do Parsec. Compreendemos que, em projetos futuros, devemos ter este aspeto em consideração e pesquisar ferramentas ou métodos alternativos.

Conclusões

Ao longo do desenvolvimento deste projeto colaborativo, várias são as conclusões obtidas que podem ser destacadas. Em primeiro lugar, através deste projeto e da sua prova de conceito conseguimos demonstrar que é possível automatizar o processo de subida e descida dos estores utilizando LEDs como indicadores visuais. Isto veio a confirmar a viabilidade da automação dos estores de forma alternativa, mesmo que não tenha sido alcançada a funcionalidade completa esperada.

De um modo geral, podemos afirmar que o nosso projeto demonstrou resultados bastante satisfatórios, tendo sido alcançados quase todos os objetivos inicialmente propostos. Similarmente, todos os testes realizados, à exceção do relativo ao posicionamento dos estores, foram bem sucedidos, revelando a eficácia e confiabilidade das soluções implementadas e sugerindo que, no geral, os resultados dos testes validaram a funcionalidade básica do protótipo, destacando a capacidade de controlo remoto, a clareza das indicações visuais e a estabilidade da comunicação entre os dispositivos.

Conforme mencionado anteriormente, porém em desacordo com as nossas expectativas, não conseguimos alcançar o objetivo específico de utilizar um motor bidirecional devido a restrições financeiras e limitações de prazos. Em vez disso, adotámos uma abordagem alternativa, utilizando um LEDs da maneira a simular o abrir e fechar dos estores, associando cada uma das funções a cores, o que proporcionou uma funcionalidade semelhante, embora tenha afetado outros componentes, como o posicionamento dos estores. Nesse contexto, não foi viável determinar uma posição intermediária dos estores, nem simular de forma precisa o movimento dos mesmos, ou avaliar o tempo necessário para a conclusão das funções de descida e subida em diferentes posições. Portanto, a nossa única alternativa foi definir *timers* de 5 segundos para os processos de abertura e fechamento dos estores, independentemente da direção. Adicionalmente, consideramos que o facto do protótipo não proporcionar uma demonstração visual idêntica a um estore físico real, pode afetar a experiência do utilizador. No entanto, ficámos positivamente surpreendidos ao perceber que marcas como a Shelly são alternativas surpreendentemente acessíveis a outras mais conhecidas como a Google ou a Amazon, tanto em termos financeiros, como em termos de simplicidade de configuração e equipamentos, para a criação de casas inteligentes. Tal competência ressuscita o prolongado debate sobre a eficácia, qualidade ou confiabilidade dos serviços e produtos prestados pelas

grandes empresas face há também muito eficaz, e muitas vezes até superior, qualidade geral dos produtos e serviços de empresas/entidades mais pequenas ou não tão conhecidas.

Com todos os problemas e imprevistos com que nos deparamos, consideramos fundamental que, num próximo projeto, seja feita uma pesquisa mais aprofundada sobre o tipo de produtos que se pretendem adquirir. Embora o grupo tenha conseguido, eventualmente, arranjar um sensor de qualidade para medição da temperatura e humidade, é inegável o prejuízo, quer monetário quer temporal, que o sensor originalmente utilizado causou. Para se optar por uma solução aparentemente menos dispendiosa, teve de se abdicar, de qualquer modo, de tempo e mesmo até dinheiro que não teria sido necessariamente gasto caso a pesquisa inicial pelo sensor tivesse sido conduzida com maior grau de minuciosidade.

Ainda assim, o investimento de tempo realizado pelo grupo neste esforço coletivo evidenciou a utilidade e importância da área da Domótica, bem como o potencial que dela advém. O protótipo desenvolvido permitiu abordar temas, ferramentas e ideias que inicialmente se encontravam fora da zona de conforto de qualquer um dos constituintes do grupo, mas que, após pesquisa sobre esta temática e pelo interesse generalizado que a proposta do projeto nos causou, foi possível desenvolver competências, conhecimentos, ideias e opiniões com sentido crítico oportuno sobre a área e sobre a indústria da automação e das casas inteligentes.

A plataforma openHAB demonstrou ser uma opção bastante vantajosa, não só pela sua abordagem *open-source* ao código, gratuidade e pelo esforço da equipa de desenvolvedores altamente motivados e dedicados, mas também pela sua notável acessibilidade. A criação de um ambiente de casa inteligente, diretamente no conforto da nossa própria casa, permite reforçar o enorme potencial que a Domótica possui como parte integrante do quotidiano da mais banal das famílias.

Por fim, vale a pena mencionar que todos os conhecimentos e aprendizagens adquiridos no decorrer deste projeto académico estão devidamente registados e documentados e poderão ser utilizados futuramente como referência para trabalhos em áreas que por este sejam albergadas. Esta experiência, além de nos ter proporcionado conhecimentos acerca deste tema, proporcionou o desenvolvimento de competências de resolução de problemas, procura de alternativas e gestão de recursos, características fundamentais para projetos de engenharia e tecnologia, tendo um impacto direto no desenvolvimento de competências dos envolvidos e fornecendo uma experiência prática em engenharia e tecnologia, através da composição da maquete.

Trabalhos futuros

Face às limitações encontradas pelo grupo no decorrer deste projeto, são apresentadas, em seguida, algumas sugestões de aspetos a colmatar em trabalhos futuros relacionados com o tema, como:

1. Utilizar um motor bidirecional. A incorporação de um motor bidirecional permitiria um controlo mais preciso do posicionamento dos estores, proporcionando a capacidade de os ajustar em diferentes níveis de abertura, como posições intermediárias, permitindo uma personalização ainda maior do ambiente e um controlo mais refinado da entrada de luz.
2. Desenvolvimento de funcionalidades avançadas de controlo e personalização. Seria interessante explorar a possibilidade de criar funcionalidades mais avançadas na aplicação mobile, permitindo aos utilizadores personalizar e agendar diferentes configurações para os dispositivos, de acordo com as suas preferências e rotinas diárias. Isso incluiria, por exemplo, agendar uma hora para abertura e fecho dos estores de acordo com a altura do dia e o tipo de luminosidade exterior.
3. Testes em escala real. Embora neste projeto tenha sido possível desenvolver uma prova de conceito, reconhecemos as limitações dos recursos e infraestruturas que impediram a implementação de uma maquete de maiores dimensões, ou com um potencial de utilização equiparadas às situações diárias e mundanas. Expandir o protótipo desenvolvido para uma implementação em escala real, instalando o sistema de automação numa residência real e avaliando o seu desempenho, usabilidade e eficiência num ambiente mais complexo e dinâmico permitiria a identificação de desafios adicionais e a realização de ajustes e melhorias com base na experiência do mundo real. No entanto, é importante reconhecer que a implementação em escala real envolve desafios adicionais, como a necessidade de recursos financeiros, acesso a uma residência real para a instalação e realização de testes, e a colaboração dos utilizadores para participarem no projeto. Assim, no contexto deste trabalho, as limitações de recursos tornaram inviável a realização de testes em escala real.
4. A acrescentar ao fator monetário, sendo o molde que se desenhou, projetou e concretizou um projeto de *Smart Home*, assume-se que a implementação numa situação e à escala real implicaria a prévia posse, por parte de hipotéticos clientes, de equipamentos mais dispendiosos, tal como o próprio estore elétrico.

Bibliografia

ASUS Portugal. (s.d.). ASUS Portugal. https://www.asus.com/pt/networking-iot-servers/wifi-routers/asus-wifi-routers/rt-ax56u/

Castro Electronica. (s.d.). Switch Module for Autom. Wi-Fi/Bluetooh w/ Med. Cons. 110/230VAC; 24-240VDC 16A - ShellyPlus 1PM. Retirado de https://www.castroelectronica.pt/product/modulo-interruptor-p-autom-wi-fibluetooh-c-med-cons-110230vac-24-240vdc-16a--shellyplus-1pm

Castro Electronica. (s.d.). Switch Module w/ 2 Channels for Autom. Wi-Fi/BT w/ Med. Cons. and Cont. Coverage - Shelly Plus 2PM. Retirado de https://www.castroelectronica.pt/product/modulo-interruptor-c-2-canais-p-autom-wi-fibt-c-med-cons-e-contr-cobertura--shelly-plus-2pm

FONSECA, Filipe e MONTEIRO, Ana — LABSI: Projecto e automação de uma casa inteligente. Disponível em: http://ave.dee.isep.ipp.pt/~lbf/LABSI/Proj%202019_2020/34%20Automacao%20de%20uma%20Casa%20Inteligente/

MACHADO, João Pedro Valadas — Projeto de sistema de supervisão para habitações. Lisboa, 2020. Trabalho Final de Mestrado para obtenção do grau de Mestre em Engenharia Mecânica. Apresentado ao Instituto Superior De Engenharia De Lisboa. Disponível em: https://repositorio.ipl.pt/handle/10400.21/12438 [Consult. Em 10-06-2023].

Mauser. Monitorizador ambiental de Temperatura e humidade - Branco - Shelly H&T (s.d.). Retirado de https://mauser.pt/catalog/product_info.php?products_id=096-7167&utm_source=google&utm_medium=cpc&utm_campaign=shopping_shelly_pt_012&utm_content=feed_shelly&gclid=Cj0KCQjw7uSkBhDGARIsAMCZNJuCL29nFEyTxeVmSHEykP09WeRfvQODqAUFY62GT-CjkfjLZHWzgm0aAldYEALw_wcB

Mauser. Kit computador Raspberry Pi400 - teclado e livro português - 1.8GHz 4GB - com WiFi 2.4/5GHz + BT 5.0 - Raspberry Pi Pi400 KIT-PT (s.d.) Retirado de

https://mauser.pt/catalog/product_info.php?products_id=096-9501&utm_source=google&utm_medium=cpc&utm_campaign=shopping_catalog_pt_011&utm_content=feed&gclid=Cj0KCQjw7uSkBhDGARIsAMCZNJtKbN-OwoslNMTBY9dYxPLEtsomWwbHoR9bQNtlf48bmTMa3kfOvjUaAofIEALw_wcB

PEREIRA, João Francisco Oliveira — Domótica com Arduino. Lisboa, 2015. Trabalho final de curso. Apresentado no Instituto Superior de Tecnologias Avançadas. Disponível em: http://www.projetos.tech/wp-content/uploads/2019/01/1918-Jo%C3%A3o-Pereira.pdf [Consult. Em 10-06-2023].

OpenHAB Empowering the Smart Home (s.d.). Retirado de https://www.openhab.org/docs

OpenHab. Empowering the Smart Home (s.d.). Retirado de https://community.openhab.org

OpenHAB 3 + Visual Studio Code Extension Setup Guide. *Youtube*. Retirado de https://www.youtube.com/watch?v=Qw8Feci7igI

Stackoverflow (s.d.). Retirado de https://stackoverflow.com