INSTITUTO
UNIVERSITÁRIO
DE LISBOA

Sistema para Monitorizar Animais de Estimação Remotamente

Filipe Manuel Nogueira Afonso

Mestrado em Engenharia de Telecomunicações e Informática

Orientador:
Professor Doutor Nuno Manuel Branco Souto, Professor Auxiliar,
ISCTE-IUL

Co-Orientador:
Professor Doutor Pedro Joaquim Amaro Sebastião, Professor Auxiliar,
ISCTE-IUL

Outubro, 2020

Resumo

Com o crescimento acelerado da tecnologia Internet das Coisas (IoT), o custo dos componentes eletrónicos orientados para este tipo de aplicações tem vindo a descer consideravelmente. Desta forma, torna-se viável a sua utilização nas mais diversas áreas. Dentro do largo espectro de aplicações possíveis, a utilização de sensores de temperatura, humidade, distância, etc., que permitem uma monitorização em tempo real e constante das instalações de animais de estimação, tendo em vista o bem-estar dos mesmos. Neste contexto a proposta foca-se no desenvolvimento de um sistema de monitorização com compatibilidade IoT.

Nesta dissertação é apresentado um sistema flexível e de baixo custo que permite monitorizar animais de estimação remotamente e em tempo real, utilizando uma abordagem orientada para a IoT e impressão 3D.

Para atingir este objetivo será desenvolvida uma interface de controlo, aquisição e visualização de sinais em tempo real e uma camera.

As principais contribuições deste estudo passam pelo desenho e implementação de toda a solução, incluindo uma demonstração prática através de um protótipo impresso em 3D, que funciona com dispensador de alimentação para gatos, tendo sido realizado um estudo para escolher e decidir qual o *hardware* e *software* mais adequado a utilizar, de forma atingir os objetivos pretendidos.

A principal vantagem do sistema desenvolvido, e que o distingue de outras soluções, é a capacidade de ser escalável, quer ao nível do número de sensores utilizados, quer ao nível de dispositivos controlados pela aplicação *Mobile*. Para além disso, todos os componentes utilizados são eficientes e de baixo custo. A utilização de peças desenhadas e impressas em 3D torna o sistema totalmente diferenciador e único dos restantes.

Palavras-Chave: Internet das Coisas; Impressão 3D; Sensores; ESP32, Arduino, Raspberry Pi.

Abstract

With the accelerated growth of Internet of Things (IoT) technology, the cost of electronic components oriented to this type of applications has been falling considerably. This way, it is possible to use them in the most diverse areas. Within the broad spectrum of possible applications, the use of temperature, humidity, distance sensors, etc., allows real-time and constant monitoring of the facilities, with a view to the welfare of pets. Within this context, the proposal focuses on the development of a monitoring system with IoT compatibility.

This dissertation presents a flexible and low cost system that allows monitoring pets remotely and in real time, using an approach oriented to IoT and 3D printing.

To achieve this objective, a control, acquisition and visualization interface for real time signals and a camera will be developed.

The main contributions of this study are the design and implementation of the entire solution, including a practical demonstration through a 3D printed prototype, which works with a cat food dispenser. A study was carried out to choose and decide which hardware and software is most suitable to use in order to achieve the intended objectives.

The main advantage of this system, which distinguishes it from other solutions, is its scalability, both in terms of the number of sensors used, and in terms of devices controlled by the Mobile application. In addition, all the components used are efficient and low cost. The use of parts designed and printed in 3D makes the system totally different and unique from the rest.

Keywords: Internet of Things; 3D Printing; Sensors; ESP32; Arduino; Raspberry Pi.

Agradecimentos

Em primeiro lugar, gostaria de agradecer ao Professor Nuno Souto pela sua orientação e conselhos para a elaboração da minha dissertação. Foi um prazer trabalhar com o Professor, que sempre teve uma palavra de incentivo e um grande sentido de ajuda durante os últimos meses. O seu conhecimento e experiência foram fundamentais para superar todos os obstáculos que foram surgindo ao longo do tempo.

Quero também agradecer ao Professor Pedro Sebastião pelas suas contribuições, ideias e disponibilidade para me ajudar neste projeto.

Agradecimento ao Instituto de Telecomunicações (IT).

À minha família, em especial a minha mulher Sandra e aos meus filhos, Martim e Maria, pelo vosso apoio incondicional ao longo destes últimos meses.

A todos os que enumerei o meu sincero “Obrigado”.

Índice Geral

Índice de Tabelas

Índice de Figuras

Glossário de Abreviaturas e Siglas

3GPP – **3**rd **G**eneration **P**artnership **P**roject

ABS – **A**crylonitrile **B**utadiene **S**tyrene

BLE – **B**luetooth **L**ow **E**nergy

BSS – **B**asic **S**ervice **S**et

CSMA/CA – **C**arrier **S**ense **M**ultiple **A**ccess with **C**ollision **A**voidance

CSS – **C**hip **S**pread **S**pectrum

CSV – **C**ommand **S**eparated **V**alues

ESS – **E**xtend **S**ervice **S**et

GPIO – **G**eneral **P**urpose **I**nputs **O**utputs

HQ – **H**igh **Q**uality

HTTP – **H**ypertext **T**ransfer **P**rotocol

HTTPS – **H**ypertext **T**ransfer **P**rotocol **S**ecure

IA – **I**nteligência **A**rtificial

IBSS – **I**ndependent **B**asic **S**ervice **S**et

IEEE – **I**nstitute of **E**lectrical and **E**lectronics **E**ngineers

IOT – **I**nternet **o**f **T**hings

IP – **I**nternet **P**rotocol

LAN – **L**ocal **A**rea **N**etwork

LPWA – **L**ow **P**ower **W**ide **A**rea

MQTT – **M**essage **Q**ueuing **T**elemetry **T**ransport

PC – **P**oly**c**arbonate

PETG – **P**oly**e**thylene **T**erephthalate **G**lycol

PLA – **P**oly**l**actic **A**cid

RPi – **R**aspberry **Pi**

SD – **S**ecure **D**igital

SLA – **S**tereo**l**ithography **A**pparatus

SMAER – **S**istema para **M**onitorizar **A**nimais de **E**stimação **R**emotamente

SSH – **S**ecure **Sh**ell

SSL – **S**ecure **S**ockets **L**ayer

TCP – **T**ransmission **C**ontrol **P**rotocol

TLS – **T**ransport **L**ayer **S**ecurity

TPU – **T**hermoplastic **P**oly**u**rethane

VOIP – **V**oice **o**ver **IP**

WI-FI – **W**ireless **F**idelity

WLAN – **W**ireless **L**ocal **A**rea **N**etwork

Capítulo 1 – Introdução

Nos dias de hoje, existe cada vez mais a necessidade de conectar qualquer dispositivo à Internet, não apenas para enviar os dados para os servidores remotos, de forma a processar e armazenar a informação, mas também para ser controlado em qualquer parte do mundo, através da Internet. Todos os dias e cada vez estamos a ligar novos equipamentos à WEB e estima-se que em 2025, o número de dispositivos IoT (*Internet of Things*) alcance o valor de 77.44 biliões [1].

As casas inteligentes, onde é possível controlar remotamente através do *smartphon*e as luzes de uma casa, o ar condicionado ou até outro equipamento menos inteligente ligado a uma simples tomada 220V, são exemplos deste novo paradigma que chamamos a Internet das Coisas.

A IoT foi projetada com o objetivo de ganharmos tempo, melhorando a nossa qualidade de vida e está presente em muitas experências do nosso quotodiano, como nos transportes, saúde e automação industrial entre outros.

Por outro lado a Inteligência Artificial também designada de IA, corresponde a um ramo da ciência da computação que se dedica ao estudo e investigação sobre formas de replicar a inteligência do ser humano em máquinas, tornando-as capazes de realizar tarefas, que normalmente, exigem competências humanas. Os algoritmos mais utilizados em sistemas de IA incluem aprendizagem, raciocínio lógico, planeamento, solução de problemas e tomadas de decisão.

Os assistentes virtuais e os *chatbots* são alguns exemplos de serviços que nos dias de hoje já usam IA. Este tipo de tecnologia utilizam os dados de milhões de pessoas e geram todos os dias melhores respostas às nossas perguntas.

Se combinarmos estas duas vertentes, a IoT como obtenção dos dados e tendências, e IA para analise e tratamento dos dados sem a necessidade do ser humano, obtemos uma "máquina" que se molda de acordo com as informações que recebe ao longo do tempo, seguindo um processo evolutivo de aprendizagem.

É por este motivo que quanto mais dispositivos ligados à rede Internet usamos, mais inteligentes eles se tornam.

1.1. Motivação

O facto de vivermos numa Sociedade em que cada vez mais as pessoas possuem os seus animais de estimação, leva-nos a ter algumas preocupações quando vamos de férias ou de fins-de-semana mais prolongados. Existe sempre a preocupação de saber como estão os animais, o que fazem num determinado momento, e sobretudo que não falte comida a estes membros da família.

Apesar de já existirem alguns dispensadores automáticos de Ração no mercado, são bastante limitados em termos de funcionalidades. Em regra, os tipos de dispositivos existentes são apenas mecânicos e sem qualquer tipo de inteligência ou sensores embutidos. Acabam por funcionar de um modo isolado por não possuírem qualquer conectividade com as redes.

Se juntarmos a estes dispositivos menos inteligentes o conceito de IoT, passamos a ter o controlo sobre estes equipamentos em qualquer parte do mundo [2]. Para além de que, será possível implementar rotinas, envio de alertas e transferência de dados em tempo real. O que permite potencializar novas funcionalidades e serviços [3] [4].

A principal motivação é fazer algo diferenciador, barato, totalmente *open-source*, com materiais que já possuo e sobretudo acessível a todos, pois normalmente estas soluções são bastante caras [5].

1.2. Objetivos

O objetivo principal desta dissertação consiste no desenvolvimento e implementação de um sistema de monitorização de animais de estimação. Para tal é necessário atingir os seguintes sub-objetivos:

- Desenhar e fabricar um protótipo com recurso à impressão 3D.
- Avaliar e selecionar o *hardware* mais adequado ao sistema, preferencialmente de baixo custo.
- Estudar soluções para a implementação da comunicação entre os dispositivos utilizados.
- Desenvolver o *software* que permita a realização das funcionalidades desejadas.

1.3. Estrutura e Organização da Dissertação

Esta dissertação está estruturada em cinco capítulos que pretendem refletir as diferentes fases do estudo.O primeiro capítulo introduz o tema da investigação e objetivos da mesma bem como uma breve descrição da estrutura do trabalho.

O segundo capítulo reflete o enquadramento teórico, designado por Revisão da Literatura.

O terceiro capítulo é dedicado à descrição da Arquitetura do Sistema.

O quarto capítulo apresenta a Análise dos Testes e Resultados Obtidos.

No quinto e último capítulo apresentam-se as Conclusões deste estudo bem como as Recomendações, Limitações e Trabalhos Futuros.

Capítulo 2 – Revisão da Literatura

Este capítulo tem por base uma revisão das possíveis abordagens que podem ser adotadas para o desenvolvimento de um sistema inteligente aplicado à monitorização e controlo de animais de estimação.

Na secção 2.1 é descrita a importância que a IoT tem nos dias de hoje e os seus principais componentes. A secção 2.2 descreve os protocolos de comunicação sem fios mais utilizados.

A secção 2.3 descreve métodos de comunicação com o servidor.

Na secção 2.4 são apresentadas algumas placas controladoras adequadas para projetos de IoT.

Por último, na secção 2.5 é abordado o tema da modelagem e impressão 3D, onde são apresentados alguns tipos de materiais utilizados em impressoras 3D.

2.1. Internet das Coisas

Uma rede de Sensores tem sem dúvida nenhuma um papel muito importante no desenvolvimento de um sistema IoT, pois reúnem os dados dos sensores e de seguida toma algumas decisões. Tipicamente as Redes de Sensores são compostas por três tipos de nós [5] [6]:

A. ***Sensor Node*** – Compõem o nível mais baixo da rede de sensores e são responsáveis por transmitir os dados para outro nó da rede. Tipicamente este nó não armazena nem manipula os dados.

B. ***Data Node*** – São Sensores que armazenam dados além de enviar para outro nó da rede. O armazenamento de dados pode ser usado para realizar tarefas autónomas ou manter os dados guardados enquanto as comunicações encontram inoperacionais. Por terem estas características, eles exigem mais recursos de *hardware*, mais concretamente ao nível do armazenamento.

C. ***Aggregation Node*** – Estes nós encontram-se no fim da cadeia da rede, são responsáveis por coletar os dados dos restantes sensores (*Data* e *Sensor Nodes*), e de seguida através de uma *gateway* enviar esses dados para um servidor ou *Cloud*.

Na figura 1 são representados os três tipos de nós.

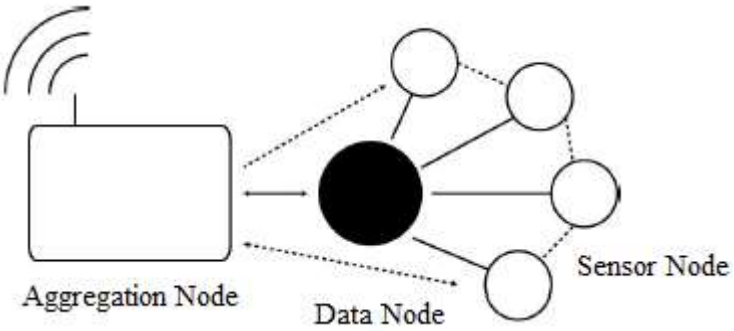


Figura 1 – Tipo de Sensores

2.2. Protocolos de Comunicação

Em conjunto com os Sensores, a comunicação [7] é também uma componente bastante importante em qualquer sistema IoT, pois possibilidade o envio dos dados entre os nós e o servidor. Sem os protocolos de comunicação, seria impensável criar sistemas IoT. De seguida serão apresentados as tecnologias sem fios mais utilizadas nos sistemas IoT, por permitirem uma maior flexibilidade de utilização e implementação em qualquer sistema em relação às tecnologias com fios. Também serão abordados os protocolos mais usados do lado do servidor.

A tecnologia de redes sem fios permite que os dispositivos comuniquem entre si sem a utilização de cabos. É sem dúvida uma grande vantagem pois possibilita a utilização de dispositivos de pequena dimensão em qualquer lugar, sem a preocupação do espaço, nem com os fios.

Na tabela 1 são apresentadas algumas das características mais importantes das redes sem fios mais utilizadas nos dias de hoje. Mais à frente, serão alvo de maior detalhe cada uma dessas tecnologias [5].

Tabela 1 – Principais características dos protocolos sem fios

Característica	***Wi-fi***	***ZigBee***	***Bluetooth***	***LoRaWAN***
Velocidade [Mbps]	11	0.25	1	0.011
Frequência [GHz]	2.45	2.45	2.45	0.433
Distância [m]	1-100	1-100	10	20000
Nós	32	65540	1-35	15000
Consumo [mA]	100-350	1-35	1-10	1-10
Complexidade	Alta	Média	Baixa	Média
Segurança	WPA/WPA2	128 bit	128 bit	128 bit

2.2.1 IEEE 802.11 (*Wi-fi*)

O *Wi-fi* é um protocolo de comunicação bidirecional que permite aos dispositivos trocarem informação entre sim numa rede local desginada WLAN. Suportada numa tecnologia de rádio, o padrão 802.11 foi aprovado pelo IEEE em 1997 e até aos dias de hoje tem sofrido bastantes melhorias que ao nível das velocidades atingidas, quer ao nível de funcionalidades presentes. Inicialmente surgiu com uma largura de banda 1 Mbps, seguidamente com as normas b e g foi possível atingir os 54Mbps nos 2.4GHz. Atualmente com as normas Europeias h e ac já é possível atingir velocidades na ordem 1Gbps [8] na faixa de frequência dos 5GHz.

A arquitetura IEEE 802.11 é composta por diversos componentes que permitem a comunicação e mobilidade dos dispositivos dentro de uma rede local, através da utilização de um ponto central - *Basic Service Set* (BBS). Esta norma também permite que os dispositivos se conectem entre si diretamente sem a utilização de um BSS (designado por IBSS). A esta operação designa-se ligação *adhoc* e dispensa um ponto de acesso (*Access Point*).

A utilização de ESS (*Extend Service Set*) para controlo dos diversos BSS numa rede, permite criar uma rede de grande dimensão, com elevada cobertura e complexidade.

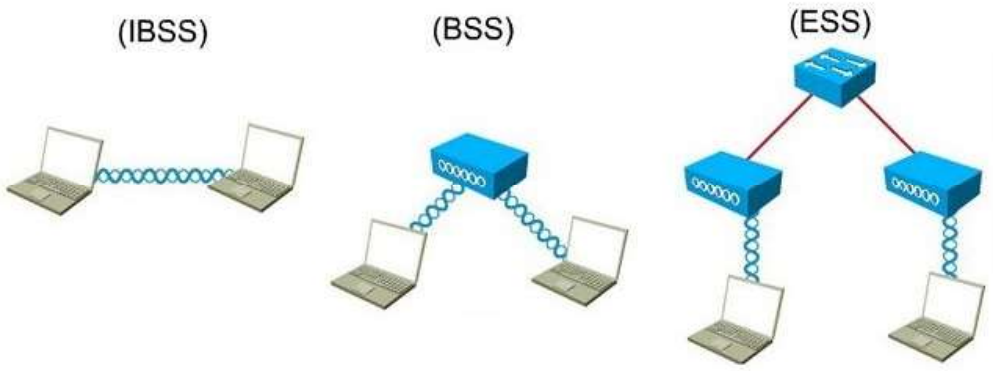


Figura 2 – IBSS e ESS nas redes Wi-fi

Na Tabela 2 são descritas algumas das vantagens e desvantagens da norma IEEE 802.11 (*Wi-fi*). De referir que a principal desvantagem é o consumo de energia um pouco elevado para uma rede de sensores.

Tabela 2 – Vantagens e Desvantagens da Norma IEEE 802.11 (Wi-fi)

Vantagens	Desvantagens
• Segurança, permite encriptação de 128-Bit AES; • Fácil de adicionar / remover dispositivos; • Permite atravessar obstáculos como parades.	• Consumo energético um pouco elevado.

2.2.2 IEEE 802.15.1 (*Bluetooth*)

Tecnologia desenvolvida pela empresa *Ericsson Mobile Communications* e apresentada em 1994 do tipo WPAN. Consiste num sistema de rádio de curto alcance, desenhado para substituir cabos e ligar periféricos. Foi também projetado para a transferência de informação entre os dispositivos de forma fácil, segura e rápida.

Este padrão não define apenas a parte rádio mas também toda a pilha de comunicação, que permite aos dispositivos se encontrarem e anunciarem entre si. Um dispostivo *Bluetooth* pode operar em dois modos: modo Escravo ou modo Mestre, sendo assim vai depender da função que tem em cada rede que se conecta.

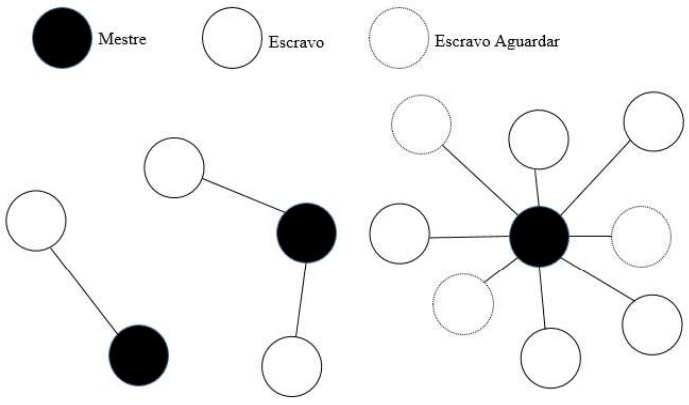


Figura 3 – Topologia Bluetooth

A tecnologia *Bluetooth* atualmente encontra-se na versão 5 também designada por *Bluetooth Low Energy* (BLE). Consiste numa versão com menor consumo energético, o que permite operar por uns períodos mais longos em relação às versões anteriores. Além do baixo consumo, outra característica importante é que tem dez vezes mais alcance que o *Bluetooth* tradicional, chegando mesmo aos 100 metros de distância entre dispositivos, tornando-se assim num bom candidato para sistemas IoT.

Na tabela 3 são apresentadas algumas vantagens e desvantagens da tecnologia *Bluetooth*, a destacar o elevado número de dispositivos com esta tecnologia e a capacidade de suportarem diversas ligações.

Tabela 3 – Vantagens e Desvantagens da Norma Bluetooth

Vantagens	Desvantagens
• Suporta um elevado número de dispositivos; • Um coordenador consegue controlar diversos escravos.	• Emparelhar toda a rede pode ser um pouco complexo de gerir; • Limite no número de nós; • Apenas permite arquitetura em estrela.

2.2.3 IEEE 802.15.4 (*ZigBee*)

Surgiu em 2002 com base na norma IEEE 802.15.4. Foi desenvolvida sobretudo para redes privadas (LR-WPAN), sendo uma tecnologia bastante utilizada nos sistemas IoT, devido ao baixo consumo de energia, baixo custo de *hardware* e alto desempenho na transferência de mensagens.

Pode ainda fornecer maior segurança, com a utilização de serviços de criptografia, autenticação, possibilitando a manipulação até 65.000 nós de diferentes plataformas. Funciona em três frequências de rádio, sendo a faixa dos 2.4 GHz mais comum, com uma taxa de transferência de pico na ordem dos 250 kbps. Para garantir e evitar colisões, utiliza o protocolo CSMA/CA [9].

Na tabela 4 são apresentadas algumas vantagens e desvantagens da tecnologia *ZigBee*, onde se destaca o baixo consumo energético e a capacidade de se adaptar à entrada e saída de dispositivos da rede.

Tabela 4 – Vantagens e Desvantagens da Norma IEEE 802.15.4 (ZigBee)

Vantagens	Desvantagens
• Segurança, permite encriptação de 128-Bit AES; • Reorganização da rede automaticamente; • Baixo consumo energético.	• Tamanho máximo do pacote 127 bytes por mensagem (250 kbps); • Incompatível com outros protocolos; • Necessita de *hardware* adicional.

2.2.4 LoRa

O termo LoRa [10] significa *Long Range*, baseia-se em técnicas de modulação de propagação do espectro, oriundas da tecnologia *Chirp Spread Spectrum* (CSS). Utiliza uma topologia em estrela, reduzido a complexidade da rede e utilizando o mecanismo de saltos entre nós. A LoRa apenas define a camada física da rede e é proprietária.

Por sua vez, o LoRaWAN é um padrão livre, de comunicação bidirecional baseado nos protocolos da camada MAC garantido assim a segurança dos sensores LoRa, modificando o uso de chaves simétricas criptográficas. Este protocolo permite comunicações a longas distâncias e de baixa potência.

Na tabela 5 são descritas algumas vantagens e desvantagens da tecnologia *Lora*, onde se destaca o baixo consumo energético e a capacidade de transmissão.

Tabela 5 – Vantagens e Desvantagens da Norma LoRa

Vantagens	Desvantagens
• Baixo consumo energético; • Funciona em longas distâncias; • Segura com codificação 128-bit AES *encryption*.	• Tamanho do pacote bastante reduzindo, no máximo 55 bytes por mensagem.

2.2.5 NB-IoT

O *NarrowBand-Internet of Things* é uma tecnologia baseada nos padrões do LPWA (*Low Power Wide Area*) [11], desenhada e projetada para dispositivos e serviços IoT. O NB-IoT melhora significativamente o consumo de energia dos dispositivos, sendo mesmo possível que uma bateria chegue a 10 anos de vida útil. O parqueamento inteligente, a localização de ativos e pessoas, telemetria da água são alguns exemplos de soluções que podem usar esta tecnologia de rede. Em Portugal, alguns operadores nacionais já começaram a investir neste tipo de serviços, pois como sabemos, será o futuro.

2.2.6 LTE Cat-M1

O LTE Cat-M1 [12] também é uma tecnologia com padrões LPWA. Opera numa faixa de frequência 1,4 Mhz com potência de transmissão de 20 dBm, oferecendo velocidades de *upload* entre os 200 e os 400kbps em média.

O baixo consumo de energia, bem como o modo de poupança, pois o Cat-M1 só é ativado quando é necessário transmitir informação, caso contrário é desligado, torna este sistema bastante apelativo para o IoT.

A sua utilização poderá ser direcionada para diferentes aplicações, como por exemplo na agricultura onde pode ajudar atingir uma ampla área geográfica com sensores e/ou atuadores de rega entre outros.

2.3. Comunicação com o Servidor

Regra geral, os dados dos sensores são coletados e enviados para um nó agregador através de uma rede sem fios, como as redes descritas na secção anterior. Posteriormente, é necessário enviar esses dados para um servidor que tipicamente encontra-se na *Cloud*, de forma a esta informação ser analisada e/ou tratada. Assim, é necessário possuir um protocolo que seja compatível com a Internet, para que seja possível comunicar entre o nó agregador e o servidor.

2.3.1 Hyper Text Transfer Protocol

Hyper Text Transfer Protocol (HTTP) é o protocolo mais comum e utilizado na Internet e pode ser utilizado em sistemas IoT para enviar informações para o servidor, através de uma arquitetura de mensagens do tipo pedido e reposta. Este protocolo permite o envio de um grande número de mensagens em pequenos pacotes, o que poderá provocar algum congestionamento em ligações com pouca largura de banda disponível [13].

O HTTP opera sobre o protocolo TCP/IP – orientado à conexão, sendo um protocolo síncrono e unidirecional. Desta forma, o cliente primeiro precisa de se ligar ao servidor e após a conexão estar estabelecida, o cliente pode começar a enviar os dados, conforme descrito nas figuras 4 e 5.

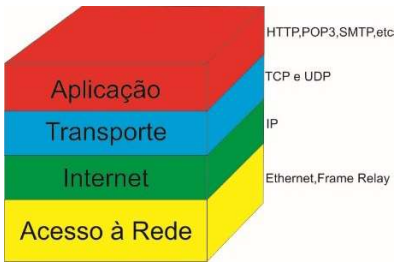


Figura 4 – Protocolos de comunicação de redes baseadas em TCP/IP

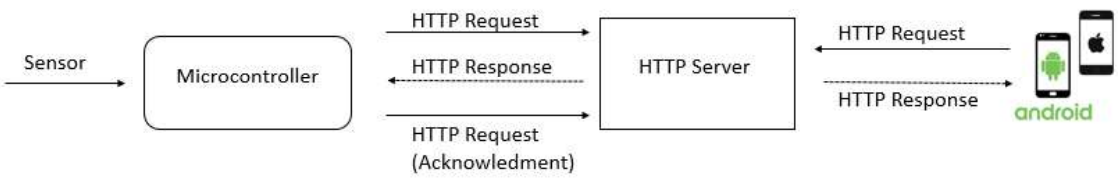


Figura 5 – Protocolo HTTP: Mensagens de Pedido / Resposta

Se ao HTTP, adicionarmos uma camada de segurança como o protocolo TLS/SSL, garantimos que os dados são transmitidos através de uma conexão criptografada, ou seja, mais segura e onde é verificada autenticidade do servidor e do cliente, através de certificados digitais. Tipicamente a porta utilizada para o protocolo HTTPS é a 443.

Num sistema IoT, este processo pode-se tornar um pouco longo e demorado, uma vez que cada nó pode ter várias mensagens a enviar, e requer para cada uma delas, estabelecer primeiro a conexão antes de enviar a mensagem.

2.3.2 Message Queuing Telemetry Transport

É um protocolo de comunicação de dados construído sobre o protocolo TCP mas com uma complexidade bastante inferior. Desenvolvido pela IBM e a *Eurotech* na década de 90 [14], permite a comunicação de muitos dispositivos para muitos, com três componentes principais: *Broker*, *Publisher*, *Subscriber*.

O *Publisher* envia uma mensagem para o *Broker* (Servidor) com dois elementos (*topic* e *info*). Posteriormente, o *Broker* é responsável por enviar a mensagem para os *Subscribers*, que por sua vez subscreveram o respetivo tópico, ou seja, apenas recebem as mensagens dos tópicos subscritos e nada mais.

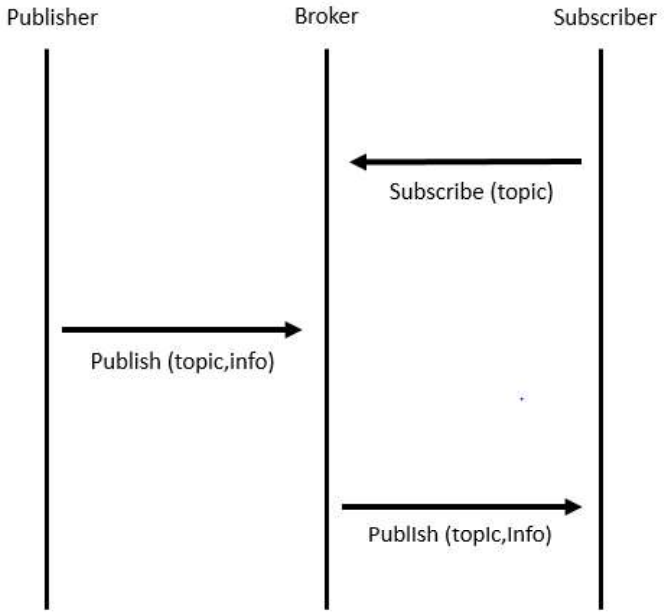


Figura 6 – Protocolo MQTT

Ao contrário do HTTP, com esta arquitetura de *Publisher/Subscriber* é possível que vários *Publishers* comuniquem ao mesmo tempo sem a necessidade de encerrar a conexão com o servidor.

O protocolo MQTT foi desenhado e projetado para os sistemas IoT, por se tratar de um protocolo leve e de fácil implementação. Proporciona o envio de mensagens para um ou mais dispositivos com uma menor latência e menor largura de banda necessária, permitindo assim, a sua utilização em dispositivos com poucos recursos de *hardware* e com um menor consumo energético.

2.3.3 Transport Layer Security

O protocolo TLS [15] atualmente na versão 1.3, foi proposto pela *Internet Engineering Task Force* [16]. É um protocolo de segurança largamente adotado e desenhado para simplificar a privacidade e segurança da informação na Internet. O TLS permite criptografar as comunicações entre aplicativos da *web* e os servidores. Adicionalmente também pode ser usado para criptografar outras comunicações como correio eletrónico, mensagens de voz sobre IP (VoIP). O TLS garante:

1. **Encriptação:** ocultação dos dados antes de serem transferidos pela rede;
2. **Autenticidade:** garante que as partes que trocam informações são quem afirmam ser;
3. **Integridade:** verifica se os dados não foram falsificados ou adulterados.

O TLS utiliza criptografia simétrica, onde as chaves criptográficas são geradas exclusivamente para cada conexão e previamente negociadas entre o servidor e o cliente antes da transmissão dos dados. A este processo designa-se de *Handshake* [17].

2.4. Plataformas de Controlo

As placas de controlo são o cérebro de qualquer projeto IoT. Utilizando um microcontrolador como um dos seus principais componentes, são responsáveis por controlar todo o sistema, uma vez que são capazes de recuperar, armazenar e analisar os dados, bem como integrar totalmente uma arquitetura de comunicação.

As soluções mais comuns utilizam microcontroladores autónomos e os sistemas mais avançados usam dispositivos do tipo SoC (*System on a Chip*) [18].

2.4.1. Arduino

O Arduino é uma plataforma de *hardware* em código aberto (*open-source*), criada em Itália em 2005. A placa Arduino UNO (uma das placas da família Arduino) é baseada no microcontrolador ATMega328 com 32 kB de memória, com porta USB que permite carregar ou programar o Arduino.

Tem o seu próprio Ambiente de Desenvolvimento [19], compatível com diversos sistemas operativos, como *Linux*, *Windows* e *MacOS*. Com a utilização de bibliotecas, os utilizadores podem criar os seus projetos através da linguagem de programação C ou C++. Possui também diversos pinos GPIO, onde é possível ligarem diversos sensores, motores ou outros dispositivos compatíveis com a placa.

Apesar de acessível, a versão UNO tem a desvantagem de não poder executar diversas tarefas em simultâneo, ou seja, apenas permite uma linha de execução, e em termos de memória interna é um pouco limitada.

Figura 7 – Placa Arduino UNO

Na tabela 6 são apresentadas as principais especificações técnicas da placa Arduino UNO.

Tabela 6 – Principais Características Técnicas do Arduino UNO

Arduino Uno
• ATmega328P • 32 KB Flash Memory • 2 KB SRAM • 1 KB EEPROM • Clock Speed 16Mhz • Digital I/O Pins 14 (6 PWM output) • Analog Input Pins 6

2.4.2. Raspberry Pi

Esta placa foi devolvida pela *Raspberry Pi Foundadion* no Reino Unido em 2009 [20]. Trata-se de um pequeno computador, de baixo custo mas com alguma capacidade de processamento e memória integrada. Permite a programação das portas entrada/saída, designadas de GPIOs através das linguagens de programação *C++*, *Python* [21].

Desde a sua criação, que os modelos Raspberry Pi têm vindo a sofrer sucessivas melhorias, quer ao nível do *hardware* quer *software*. Atualmente, este projeto encontra-se na versão 4 [22]. Existe também uma versão com um tamanho mais reduzido, designado de Pi Zero W [23], oferecendo praticamente as mesmas funcionalidades do Pi 4 mas com um menor consumo energético.

Ambas as placas utilizam o *Raspberry Pi OS*, um sistema operativo adaptado a este tipo placas, e é instalado num cartão MicroSD.

Na figura 8 é apresentado o RPi Zero e na figura 9 o RPi versão 4.

Figura 8 – Zero

Figura 9 – Raspberry Pi 4

O baixo custo, o tamanho reduzido, a facilidade de programação para os diversos pinos, as capacidades de comunicação através de *Wi-fi* por exemplo, são as principais vantagens do Raspberry Pi. A única desvantagem será não suportar portas analógicas.

Na tabela 7 são descritas as principais especificações técnicas destas duas placas.

Tabela 7 – Características Técnicas do Raspberry Zero e Raspberry Pi 4

Raspberry Zero	Raspberry Pi 4
• 1GHz single-core CPU • 512MB RAM • Mini HDMI port • Micro USB OTG port • Micro USB power • HAT-compatible 40-pin header • Composite video and reset headers • CSI camera connector (v1.3 only)	• Broadcom BCM2711, Quad core Cortex-A72 (ARM v8) 64-bit SoC @ 1.5GHz • 2GB, 4GB or 8GB LPDDR4-3200 SDRAM (depending on model) • 2.4 GHz and 5.0 GHz IEEE 802.11ac wireless, Bluetooth 5.0, BLE • Gigabit Ethernet • 2 USB 3.0 ports; 2 USB 2.0 ports. • Raspberry Pi standard 40 pin GPIO header (fully backwards compatible with previous boards) • 2 × micro-HDMI ports (up to 4kp60 supported) • 2-lane MIPI DSI display port • 2-lane MIPI CSI camera port • 4-pole stereo audio and composite video port • H.265 (4kp60 decode), H264 (1080p60 decode, 1080p30 encode) • Micro-SD card slot for loading operating system and data storage • 5V DC via USB-C connector (minimum 3A*) • 5V DC via GPIO header (minimum 3A*) • Power over Ethernet (PoE) enabled (requires separate PoE HAT) • Operating temperature: 0 – 50 degrees C ambient • A good quality 2.5A power supply can be used if downstream

2.4.3. ESP8266

É uma placa de baixo custo com módulo *Wi-fi* incorporado, com a capacidade de se ligar a uma rede através do protocolo TCP/IP. O ESP8266 apresenta todas as capacidades da placa Arduino Uno, com a vantagem de se poder ligar à *Internet*, o que o torna num dispositivo bastante apelativo para os sistemas IoT. Existem algumas versões deste *chip*, no entanto a mais comum é o ESP-12 [24], que possui 64kB de memória RAM e 4MB de memória *flash*. Para além das características apresentadas, permite também a ligação de sensores através dos pinos de entrada e de saída da placa. Conta também com uma porta micro USB para alimentação ou programação do *chip*.

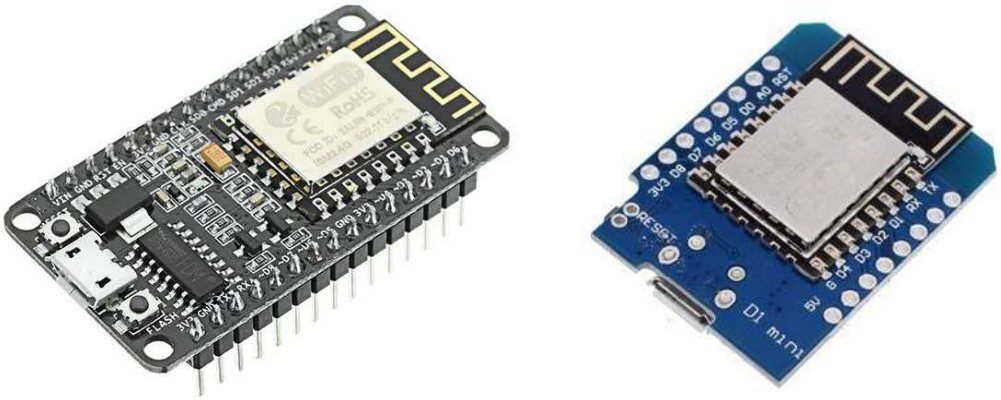


Figura 10 – Placas ESP8266

Para além do seu tamanho reduzido e consumo energético também baixo, é uma ótima solução quando é necessário ligar o dispositivo à rede, pois já possui *Wi-fi*.

Na tabela 8 são apresentadas as principais especificações técnicas do *chip* ESP8266.

Tabela 8 – Principais Características Técnicas do ESP8266

ESP8266
• Microcontroller: Tensilica 32-bit RISC CPU Xtensa LX106 • Operating Voltage: 3.3V • Input Voltage: 7-12V • Digital I/O Pins (DIO): 16 • Analog Input Pins (ADC): 1 • UARTs: 1 • SPIs: 1 • I2Cs: 1 • Flash Memory: 4 MB • SRAM: 64 KB • Clock Speed: 80 MHz • USB-TTL based on CP2102 is included onboard, Enabling Plug n Play • PCB Antenna • Small Sized module to fit smartly inside your IoT projects

2.4.4. ESP32

Esta placa é muito semelhante ao ESP8266 apresentado anteriormente, sendo mesmo considerado como o seu sucessor.

Combinando todas as vantagens do ESP8266 e levando em consideração algumas desvantagens, o ESP32 surge com microcontrolador mais avançado e com um melhor desempenho. Não inclui apenas o *Wi-fi* mas também BLE e com um *chip* de dois núcleos. As portas de entrada e saída também foram melhoradas, agora com 32 disponíveis, incluindo 12 portas analógicas e atualização da memória para 520kB.

Outra funcionalidade que está presente e é ideal para os sistemas IoT, é capacidade de hibernação quando não está a ser utilizado, permitindo assim, uma poupança de energia, diminuindo o consumo de 20mA para 100µA [25].

Figura 11 – Placa ESP32 com Camera

Tal como o ESP8266, o ESP32 é uma boa solução para projetos que necessitam de um SoC com tamanho reduzido, com acesso à Internet, de baixo custo e que não comprometa a qualidade do sistema.

Na tabela 9 são apresentadas as principais especificações do ESP32.

Tabela 9 – Principais Características Técnicas do ESP32

ESP32
• TTensilica Xtensa 32-bit LX6 microprocessor with 2 cores • IEEE802.11 b/g/n • Bluetooth Low Energy • 520 KB RAM • 16 MB Flash • GPIO 22 • SPI-I2C-UART-I2S-CAN • Analog Input Pins 6

2.4.5. LoPy4 e Gpy

A LoPy4 [26] é uma placa de desenvolvimento compatível com *MicroPython* que suporta quatro tipos de rede: *LoRa*, *SigFox*, *Wi-fi* e *Bluetooth*. Devido a suportar estas quatro tecnologias de rede, é sem dúvida uma placa bastante resiliente do ponto de vista da rede, pois permite criar até quatro canais de comunicação diferenciados.

Conta com 4Mb de memória RAM e 8Mb de memória *flash*. Permitindo também *multi-threading*, ou seja, diversos fluxos paralelos de processamento, possibilitando a execução de diversas tarefas em simultâneo.

Figura 12 – Placa LoPy4

Por sua vez, a Gpy [27] oferece três tipos de rede distintos: *Wi-fi*, *Bluetooth* e Cat-M1/NB1. Em termos de memória e processador é igual à LoPy4 analisada anteriormente, a principal diferença é nas redes que suporta.

Figura 13 – Placa GPy

2.5. Modelagem e Impressão 3D

A impressão 3D já é bastante utilizada nos dias de hoje quer ao nível particular, quer ao nível industrial. Muitas empresas que utilizavam protótipos em metal, passaram a imprimir os seus protótipos em impressoras 3D. Desta forma, reduzem os custos e também o tempo de fabrico. Ao nível dos particulares, a comunidade de *makers* tem vindo a crescer cada vez mais. A partilha de conhecimento e o desenvolvimento de modelos 3D tem tido assim um crescimento exponencial nos últimos tempos.

É caso para dizer, "O Homem Sonha e a Obra Nasce", é o que acontece com a impressão 3D.

2.5.1. Materiais

Nos dias de hoje, existem uma grande variadade de materiais utilizados na impressão 3D. Desta forma, com apenas uma impressora é possivel criar objectos para os mais diversos fins. Deve-se escolher o melhor material possível consoante o tipo de peça que se pretende imprimir e de acordo com a sua funcionalidade. Vejamos em maior detalhe alguns desses materiais.

PLA (*Polylactic Acid*)

É um material biodegradável e amigo do ambiente pois é fabricado a partir do acuçar do milho e não é prejudicial para a saúde. É um material de fácil impressão e pode ser usado em impressoras abertas ou fechadas, com ou sem mesa aquecida. Existem diversas cores e é um material com bastante precisão dimensional (bom detalhe e resolução). Não é aconselhável o PLA [28] para o uso ou exposição a altas temperaturas ou químicos corrosivos.

Na tabela 10 é apresentado as principais vantagens do PLA, destacando-se no preço mais competitivo em relação aos restantes matérias e o facto de ser biodegradável.

Tabela 10 – Principais Vantagens e Desvantagens do PLA

Vantagens	Desvantagens
• Económico comparado com os restantes matérias; • Biodegradável (amigo do ambiente); • Bom acabamento; • Ideal para usar em protótipos; • Facilidade em imprimir.	• Baixa resistência a altas temperatuaras; • Baixa flexibilidade (pode partir sob stress); • Apenas permite arquitetura em estrela.

ABS (*Acrylonitrile Butadiene Styrene*)

É um material bastante popular no mundo da impressão 3D e não só, é também utilizado na indústria dos plásticos. Oferece resistência física e suporta altas temperaturas, no entanto, tem tendência a contrair-se e é de baixa precisão dimensional. Deverá ser impresso em ambiente fechado, de forma a manter uma temperatura constante na mesa e sem flutuações para não deformar a peça. Outra característica do ABS [29] é possuir um acabamento com acetona que dá mais brilho ao objeto e que permite utilizar uma lixa fina de forma a corrigir algumas imperfeições.

Na tabela 11 são apresentadas as principais vantagens do ABS, destacando-se sobretudo a resistência do material, que supera os restantes da concorrência.

Tabela 11 – Principais Vantagens e Desvantagens do ABS

Vantagens	Desvantagens
• Forte resistência a choques; • Ideal para peças submetidas a grandes esforços ou temperaturas elevadas.	• Enfraquece quando exposto a longos períodos a raios UV; • Acabamento menos detalhado; • Custo elevado.

PETG (*Polyethylene Terephthalate Glycol*)

Este material à base de Politereftalato de Etileno Glicol [30], junta as vantagens do ABS com o PLA, ou seja, maior força física e resistência a temperaturas elevadas com a precisão dimensional.

Na tabela 12 são apresentadas algumas vantagens e desvantagens do PETG, destaca-se a resistência superior ao PLA.

Tabela 12 – Principais Vantagens e Desvantagens do PETG

Vantagens	Desvantagens
• Resistente; • Rápido tempo de produção; • Mais económico que o ABS; • Bom acabamento final; • Grande variedade de cores; • Ideal para peças submetidas a grandes esforços ou temperaturas elevadas.	• Menos resistente que o ABS.

Na figura 14 é apresentado um comparativo entre os diversos tipos de materiais com base em algumas características principais.

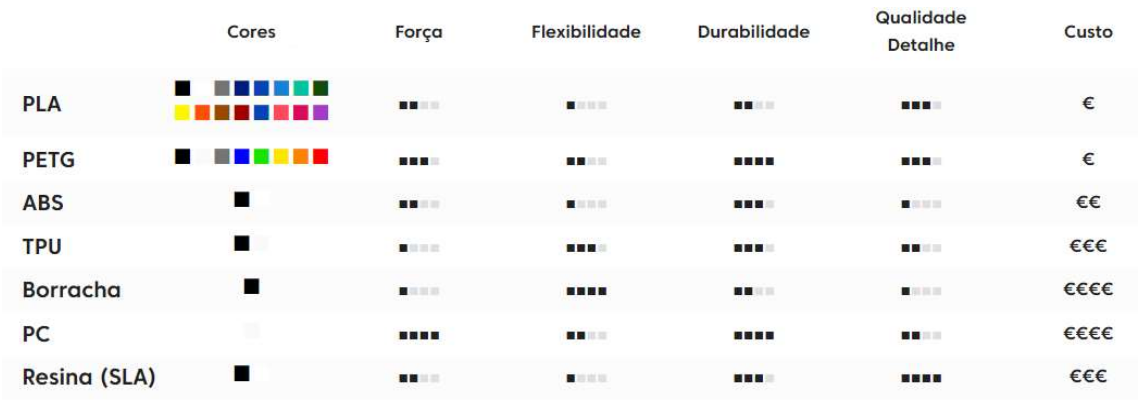


Figura 14 – Materiais 3D

2.5.2. Processo de Modelagem 3D

Relativamente à modelagem e desenho 3D, atualmente existem diversas opções, umas *free* ou *open-source* e outras soluções pagas.

O *Fusion 360* [31] e o *Blender* [32] são dois programas grátis que permitem elaborar uma modelagem 3D a partir do zero. São programas muito completos, em que o processo de aprendizagem e o conhecimento de todas as suas funcionalidades requer bastante tempo. No entanto, com algumas bases, é possível criar qualquer tipo de objeto.

Outro programa bem conhecido é o *Solidworks* [33], atualmente na versão 2020. Só está disponivel para arquitetura *Windows*, é um programa cheio de funcionalidades o que o torna também mais complexo relativamente a outros.

Depois de efetuada a modelagem 3D ou desenho técnico, é necessário passar por um fatiador, também designado por *slicer*. É neste programa que se vão ajustar todas as definições do *print*, nomeadamente: tamanho do filamento utilizado (tipicamente é 1.75mm), temperaturas da mesa e do *nozzle*, número de camadas, preenchimento entre camadas, colocação de suportes, velocidade de impressão, velocidade das ventotinhas, etc. Também neste campo dos *slicers* existem diversas opções. O *Ultimake Cura* [34] apresentado na figura 15 é dos mais utilizados.

Após a definição de todas as *settings,* é gerado um ficheiro com a extensão "Gcode", que é colocado na impressora 3D para imprimir. Este ficheiro posteriormente é interpretado pelo *firwmare* da impressora, que converte o código em movimentos dos motores, que faz com que o *nozzle* navegue nos eixos da máquina.

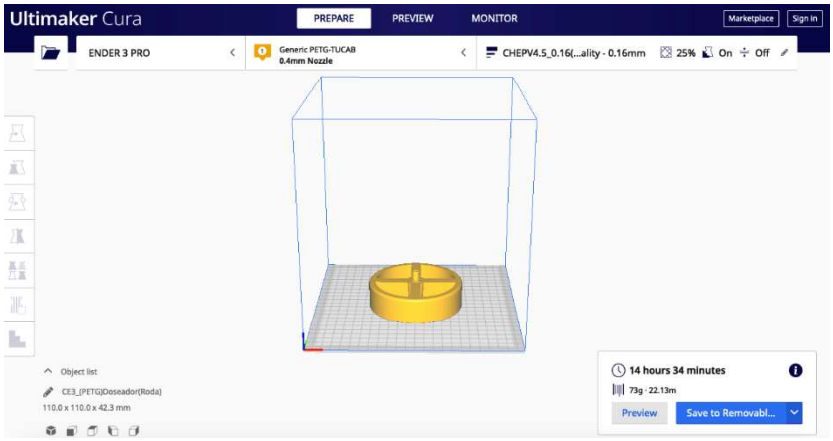


Figura 15 – Software Cura

2.5.3. Impressoras 3D

Relativamente à impressão 3D, nos dias de hoje já existem diversas opções no mercado. Antigamente uma impressora 3D custava centenas de euros, mas atualmente já é possível adquirir um *Kit* de impressora a rondar os 200 euros. Mas como em tudo, há para diversos preços, tamanhos e materiais. Alguns dos fabricantes mais conhecidos são: Creality [35], Prusa [36], BigTreeTech [37].

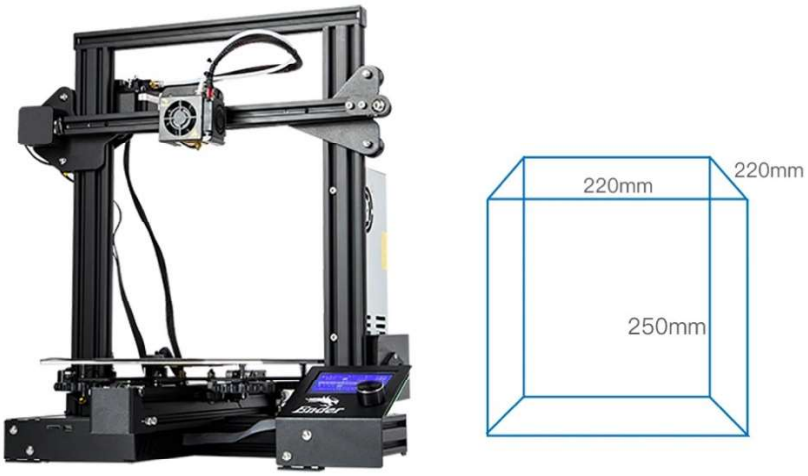


Figura 16 – Ender 3 PRO

Uma impressora 3D possui diversos componentes eletrónicos e mecânicos, dos quais se pode destacar:

- ***Extruder***

O *Extruder* é uma das peças mais importantes numa impressora 3D. É neste sector que o filamento passa e derrete até sair pelo *Nozzle*. Tipicamente um *Extruder* tem duas zonas, uma fria e outra quente. Isto acontece fruto da retração que é necessário fazer ao filamento, ou seja, nem sempre durante uma impressão é necessário estar continuamente a debitar filamento, por vezes é necessário parar, mas ao mesmo tempo é necessário manter o filamento perto do *Nozzle* porque a qualquer momento vai ser necessário voltar a libertar filamento. As ventoinhas colocadas no *Extruder* servem para manter estas duas zonas bem delimitadas e também manter a temperatura definida sempre constante. Na figura 17 podemos verificar estas duas zonas bem visíveis.

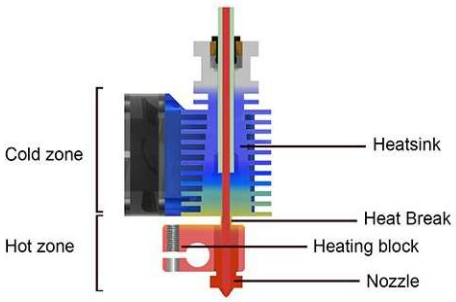


Figura 17 – Extruder [38]

- ***Nozzle***

Esta pequena peça de latão é por onde o filamento acaba por sair até mesa da impressora. Existem diversos tamanhos dos orifícios, o mais comum e utilizado é o 0.4 mm. Existem de 0.2 mm, 0.6 mm, 0.8 mm, 1 mm. Basicamente quando maior for o orifício mais filamento irá sair por este. Em termos de ganhos, ao usar tamanhos maiores, o tempo de impressão diminui, no entanto, perde-se sempre alguma resolução (leia-se pormenores ou detalhes) da peça. É necessário e importante arranjar um equilíbrio entre estes fatores.

Figura 18 – Nozzle

- **Motores**

Tipicamente uma impressora 3D tem pelo menos quatro motores. São utilizados três motores, uma para cada eixo X, Y, Z e o quarto para impulsionar o filamento até ao *Extruder*.

Figura 19 – Motores

- **Base**

A mesa é onde assenta o *print* que está a ser realizado pela impressora 3D. Normalmente, a base é aquecida durante as primeiras camadas de forma a obter uma melhor aderência entre elas. A temperatura da mesa pode atingir valores entre os 50 a 110 graus Celsius, dependendo do tipo de material que se está imprimir.

A primeira camada é a mais importante de todas, pois é sobre esta camada que todo o *print* irá ficar suportado. Por essa razão, a primeira camada deverá ter uma boa aderência à mesa para não se descolar. Existem técnicas que ajudam a que a primeira camada fique bem colada à mesa, através do uso de *sprays*, como por exemplo laca para o cabelo que contém propriedades que ajudam na aderência do objeto à base.

Figura 20 – Base de Impressora 3D

- **Filamento**

Conforme mencionado num ponto 2.5.1, existem diversos materiais que podem ser impressos numa impressora 3D. É necessário escolher o material que se melhor adequa à situação. Tipicamente, os rolos de filamento são de 1,75 mm de diâmetro e existem em diversas cores.

Independemente do material e por vezes na mesma gama e marca, poderá haver diferenças, é importante antes de qualquer impressão fazer um *print* de teste de modo a verificar se as definições de temperatura são as mais corretas uma vez que estas podem variar entre os 180 a 250 graus Celsius.

Tipicamente o filamento é vendido em rolos, conforme ilustrado na figura 21.

Figura 21 – Filamento 3D

Capítulo 3 – Arquitetura do Sistema

Como já foi referido no resumo desta dissertação, um dos objetivos deste projeto era desenvolver um sistema IoT bastante flexível, que permitisse a monitorização de animais de estimação em tempo real, em qualquer ambiente. Ao mesmo tempo, deveria ser possível adicionar de forma fácil novos sensores à solução, sem grande esforço para o utilizador e sem a necessidade de um novo desenvolvimento da aplicação *Mobile*. Para além disso, esta arquitetura deveria permitir que a mesma aplicação controle mais do que um dispositivo IoT.

Neste caso particular foi desenvolvido um dispensador de ração com um motor elétrico de forma a libertar ração de forma automática. Este dispensador integra:

- Sensor de temperatura e humidade para garantir que a ração permanece nas condições ideais para os animais;
- Sensor ultrassónico que permite obter o nível de ração no recipiente, ou seja, se está cheio, vazio ou próximo disso;
- Para além destes sensores, também é instalada uma camera que permite a qualquer momento visualizar os animais remotamente, através da aplicação disponível para Android e IOS.

Em termos de energia, para tornar o SMAER (Sistema para Monitorizar Animais de Estimação Remotamente) resiliente a falhas de energia foi colocada uma bateria (*Powerbank*) de forma a garantir o seu funcionamento permamente e facilitar a portabilidade.

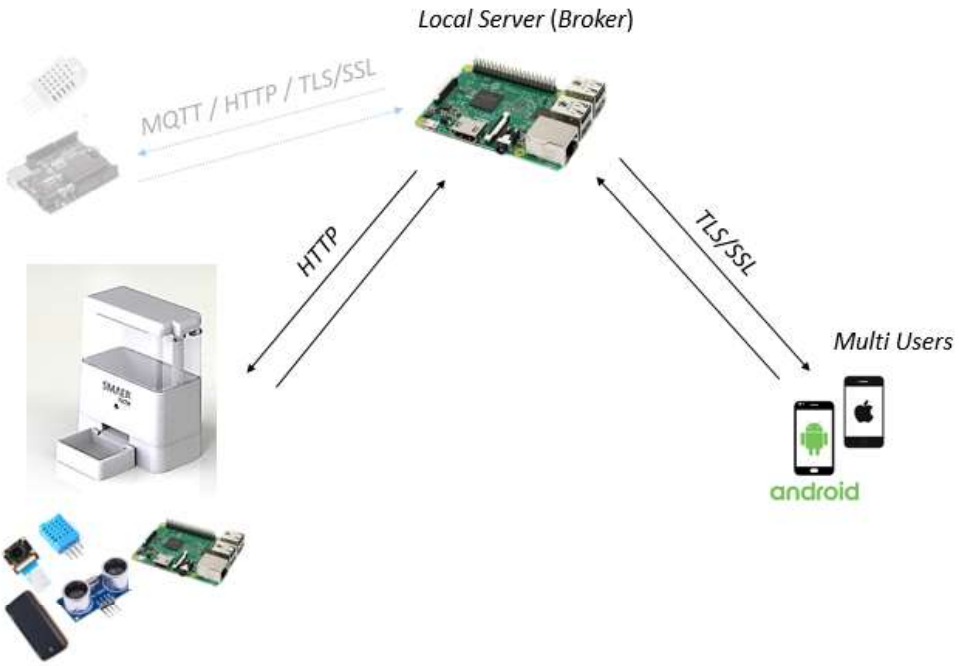


Figura 22 – Arquitetura Geral do Sistema

Nesta arquitetura é utilizado um *Broker* responsável por comunicar com os vários utilizadores da aplicação *Mobile* e também com os diversos dispostivos IoT que podemos instalar no nosso sistema. A comunicação entre os dispositivos IoT e o *Broker* como tipicamente encontram-se na mesma rede local é suportada no protocolo HTTP. No entanto, o *Local Server* também suporta os protocolos MQTT, TLS/SSL para comunicar com o *hardware.*

Como se pretende controlar os dispositivos IoT em qualquer lugar do mundo, é utilizada a Internet Pública para a comunicação entre os dispositivos *Mobile* e o *Broker*. Neste caso é utilizado um protocolo com segurança – o TLS/SSL.

3.1. Hardware

Relativamente à escolha do *Hardware* utilizado neste projeto, foram respeitados alguns critérios de seleção do material, nomeadamente o seu custo, optando sempre por componentes mais baratos mas que não comprometessem a capacidade do sistema.

3.1.1 Broker e SMAER

Para a função de servidor *Broker* a escolha recaiu sobre um *Raspberry Pi* 3 pela sua capacidade de processamento e as suas especificações técnicas serem bastantes superiores

aos restantes Arduinos analisados e mencionados na secção 2.4. Outra razão desta escolha deve-se ao preço relativamente acessível, apesar de mais caro que os Arduinos.

O facto de já ter incorporado um *chip Wi-fi* é sem dúvida nenhuma um facilitador no que respeita à configuração e operação com dispositivo.

O RPi (*Raspberry Pi*) ao utilizar um sistema operativo baseado em *Linux – Rasperry Pi OS*, torna bastante mais rápido e prático a instalação de qualquer *software*. Outro factor importante é que este equipamento já suporta cartões de memória, o que em termos de capacidade de armazenamento da informação para um servidor é bastante importante.

Relativamente ao SMAER (dispensador de Ração), pelo que já foi mencionado anteriormente sobre o RPi, se juntarmos a capacidade dos GPIOs que este dispositivo apresenta, à capacidade de integração com uma camera com melhor qualidade relativamente ao ESP32, o uso de diversas linguagens de programação como *Python*, C++, Nodejs, tornam este dispositivo muito completo e de tamanho reduzido.

3.1.2 Sensor Temperatura e Humidade (DHT 11)

Para controlar a temperatura e humidade da Ração no dispensador, foi instalado um sensor. Neste caso, foram analisadas duas opções: o DHT11 [39] e o DHT22 [40]. Ao comparar ambas as características técnicas apresentados na tabela 13, chegamos rapidamente à conclusão que o DHT22 é bastante superior: maior amplitude nas faixas de medicação de temperatura e humidade e menor percentagem de erro.

Figura 23 – DHT 11 vs DHT 22

No entanto, tendo em conta o tipo de aplicação considerado nesta dissertação, as faixas de medição de temperatura e humidade do DHT11 são suficientes e adequadas à utilização que se pretende, não justificando a opção pelo DHT22, cujo preço é quatro vezes superior relativamente ao DHT11.

Na tabela 13 é possível verificar um comparativo entre ambos os sensores.

Tabela 13 – DHT 11 vs DHT22

DHT 11	DHT 22
• Faixa de medição de humidade: 20 – 95% HR • Erro de medição de humidade: +/- 5% • Faixa de medição de temperatura: 0 – 50º C • Erro de medição da temperatura: +/- 2ºC • Tempo de resposta humidade: 2s • Tensão de operação: 3.3 – 5V	• Faixa de medição de humidade: 0 – 99.9% HR • Erro de medição de humidade: +/- 2% • Faixa de medição de temperatura: -40 a 80º C • Erro de medição da temperatura: +/- 0.5ºC • Tempo de resposta humidade: 5s • Tensão de operação: 3.3 – 5V

3.1.3 Sensor Ultrasónico (HC-SR04)

Para verificar a quantidade de Ração presente no dispensador, é instalado um Sensor Ultrasónico do tipo HC-SR04 [41] no topo do dispensador. Este sensor emite uma frequência ultrassónica na faixa dos 40kHz e aguarda pelo Eco.

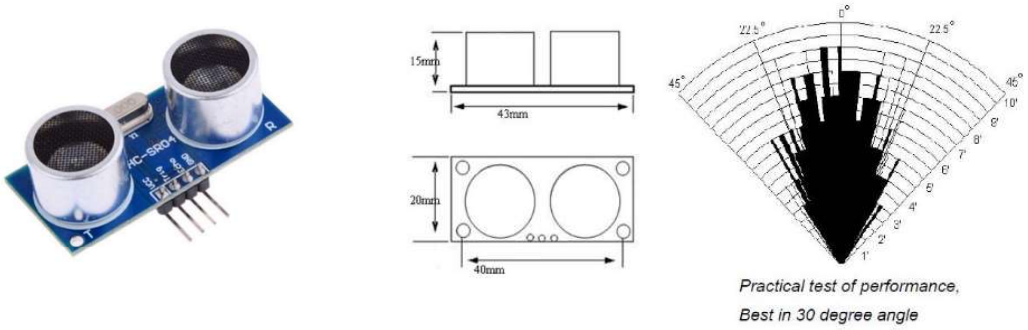


Figura 24 – Sensor Ultrasónico - HC-SR04

Para calcular a distância entre o sensor e o objetivo, registam-se dois tempos, o de envio da faixa de frequência ultrassónica (início do Pulso) e a receção dessa mesma frequência - Eco (fim do Pulso), obtendo a seguinte expressão:

$$Duração = Fim\ do\ Pulso - Início\ do\ Pulso$$

Posteriormente, ao tempo duração é necessário dividir por dois, pois o tempo calculado é o tempo deste do início até a receção do Eco, e o que se pretende é a distância até ao objeto. A velocidade do som é a velocidade de propagação de uma onda sonora num determinado ambiente (no ar a velocidade é 343m/s), ou seja:

$$Velocidade\ do\ som = \frac{Distância}{(Duração/\ 2)}$$

Simplificando esta equação, a distância é dada pela seguinte expressão:

$$Distância = \ 17150\ \times \left(\frac{Duração}{2}\right)$$

Na tabela 14 é possível analisar as principais características técnicas do Sensor Ultrasónico utilizado.

Tabela 14 – Especificações Técnicas do Sensor HC-SR04

Características técnicas do HC-SR04
• Faixa detecção: 3cm – 4m • Melhor no Ângulo de 30 graus • Consumo atual 15mA • Frequência UltraSónica 40kHz • Tensão de operação: 5V • Largura do Pulso: 10µs • Dimensões: 43x20x15 mm

3.1.4 Motor de Passo (28BYJ-48 ULN2003)

Para o movimento da roda que irá debitar a Ração para o prato, opção escolhida foi um Motor elétrico modelo 28BYJ-48, com *driver* ULN2003.

Figura 25 – Motor Passo (28BYJ-48)

É um motor simples, de 4 fases e pode operar entre os 5v – 12v. Uma vez que o *Raspberry Pi* opera nos 5v não necessita de nenhuma adaptação de tensões.

Na tabela 15 são apresentadas as principais especificações do Motor Elétrico 28BYJ-48.

Tabela 15 – Especificações Técnicas do Motor 28BYJ-48

Características técnicas do Motor 28BYJ-48
• Motor de Passo de 4 fases e 5v • Tensão de entrada de 5 – 12v • Ângulo de passo 5,625 x 1/64 • *Driver* ULN2003 • Dimensões (L x L x A): 32 x 32 x 12 mm

3.1.5 RPi Módulo de Camera

Relativamente à Camera para o *Raspberry Pi*, existem três opções no mercado: Módulo v1, Módulo v2 e *HQ* [42]. A diferença entre elas para além do preço é a resolução e o sensor incorporado em cada uma delas, conforme podemos verificar na tabela 16.

Figura 26 – RPi Módulo de Camera

Para este ambiente, a versão 1 é a mais económica e é suficientemente capaz de desempenhar uma boa função. Mesmo sendo o módulo com menor desempenho da RPi, é bastante superior à camera do ESP-32 que conta apenas com 2 Megapixéis de resolução e o *streaming* de vídeo é mais lento. Para além disso, a lente angular do RPi permite uma maior visão de cobertura.

Tabela 16 – Especificações Técnicas das Cameras Raspberry Pi

RPi Módulo Camera v1	**RPi Módulo Camera v2**	**RPi Módulo HQ**
• 5 Megapixéis • Resolução de Video: 1080p30, 720p60 e 640 × 480p60/90 • Resolução de Imagem até: 2592 × 1944 Pixéis • Sensor: OmniVision OV5647 • Abertura focal: 1/4" • Dimensões (L x L x A): 25 x 24 x 9 mm • Preço aproximado: 25€	• 8 Megapixéis • Resolução de Video: 1080p30, 720p60 e 640 × 480p60/90 • Resolução de Imagem até: 3280 × 2464 Pixéis • Sensor: Sony IMX219 • Abertura focal: 1/4" • Preço aproximado: 30€	• 12.3 Megapixéis • Resolução de Video: 1080p30, 720p60 e 640 × 480p60/90 • Resolução de Imagem até: 4056 × 3040 Pixéis • Sensor: Sony IMX219 • Preço aproximado: 50€

3.1.6 Powerbank

Para garantir o funcionamento permanente do SMAER, foi instalada uma *Powerbank* de 8000mAh de capacidade com um output até 2.4A. Desta forma, permite garantir que o RPi tem sempre energia mesmo no caso de falha de alimentação, oferecendo também uma maior portabilidade.

Esta *Powerbank* disponibiliza interfaces do tipo MicroUSB, USB-A e *Lightning* (cabo proprietário da marca *Apple*). Neste caso particular será utilizada a interface MicroUSB para conectar ao RPi.

Figura 27 – Powerbank iWalk

3.1.7 Esquema de Ligações

Como se pode verificar na figura 28, o *Raspberry Pi* conta com quarenta pinos GPIO conforme ilustrado na secção 2.4.2 e figura 28, o que permite ligar diversos sensores de entrada/saída.

Todos os sensores utilizados neste projeto, incluindo o motor e camera são ligados directamente ao RPi, conforme demostrado na figura 29.

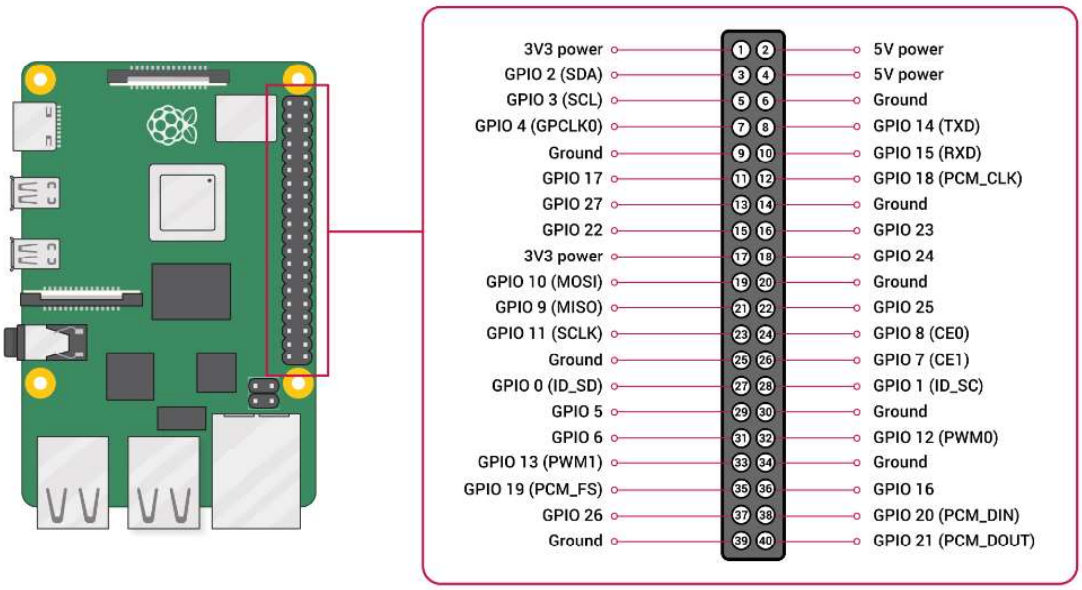


Figura 28 – Raspberry PI GPIOs

O esquema das ligações foi desenhado na aplicação *Fritzing* [43], conectando os sensores ao GPIOs do RPi, conforme ilustrado na figura 29.

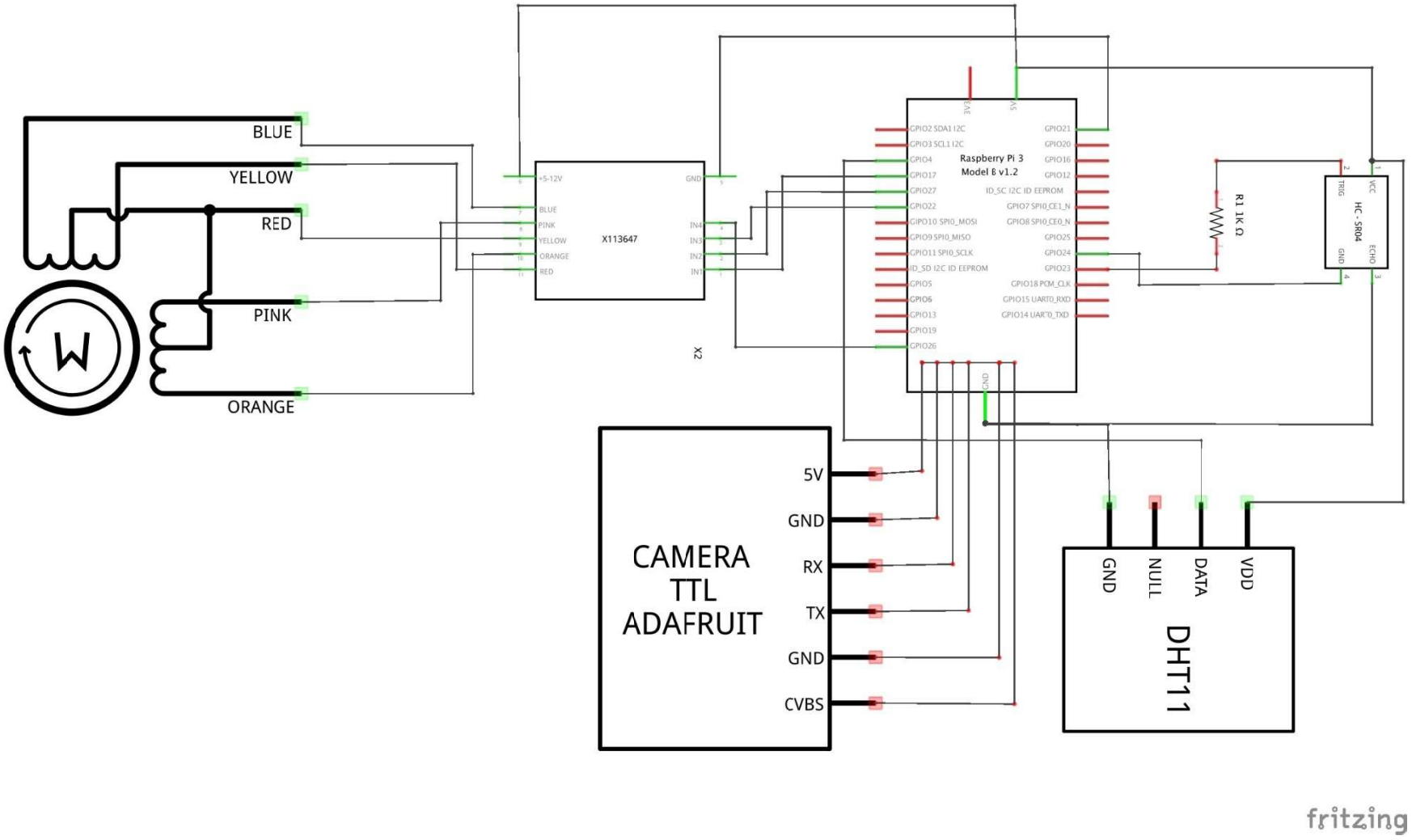


Figura 29 – Esquema de Ligações

3.2. Criação e Impressão 3D

Com base em algumas ideias e pesquisas sobre soluções existentes no mercado, o Dispensador de Ração apresentado neste projeto foi totalmente concebido de raiz no *software Solidworks*. O objetivo foi desenvolver um sistema diferenciador dos restantes, capaz de responder as necessidades do *hardware* utilizado no projeto.

Consiste num modelo de dez peças que depois de desenhado, impresso e assemblado, ficou com a aparência apresentada na figura 30.

É composto por um recipiente com tampa, onde é colocada a Ração que posteriormente através de um cone é entregue num sistema de rotação com divisórias.

Por sua vez, o doseador gira com o auxílio do motor elétrico que liberta a Ração para um prato exterior.

Todos os componentes eletrónicos utilizados (RPi, Motor, *PowerBank* e Cabos) encontram-se no compartimento inferior.

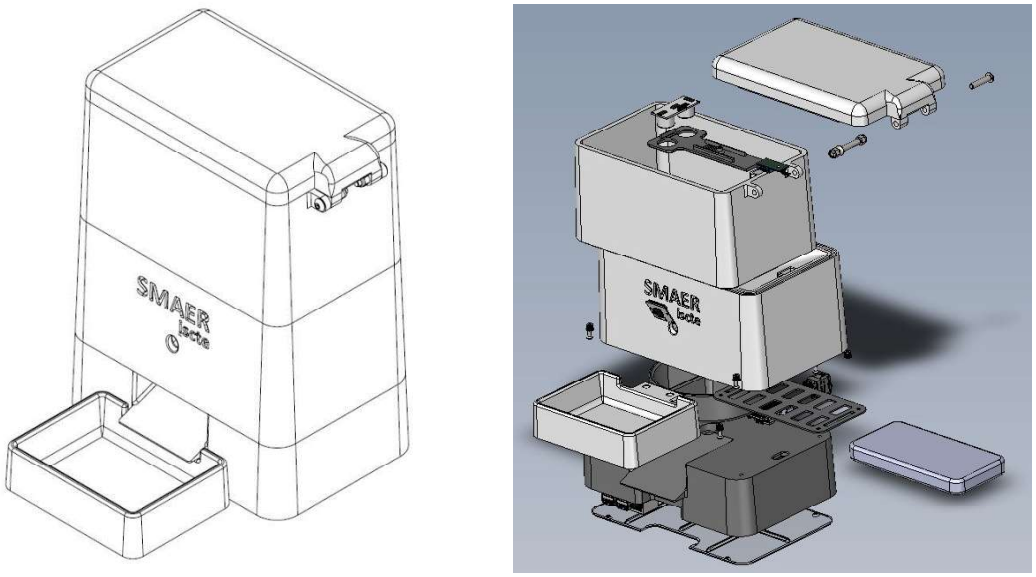


Figura 30 – SMAER

Na figura 31 é apresentado com mais detalhe as medidas técnicas do Dispensador de Ração e as respetivas localizações dos componentes utilizados e mencionados anteriormente:

1- Camera RPi
2- Sensor Ultrassónico e Sensor de Temperatura / Humidade
3- Doseador
4- Eletrónica (RPi, Motor, *PowerBank* e Cabos)

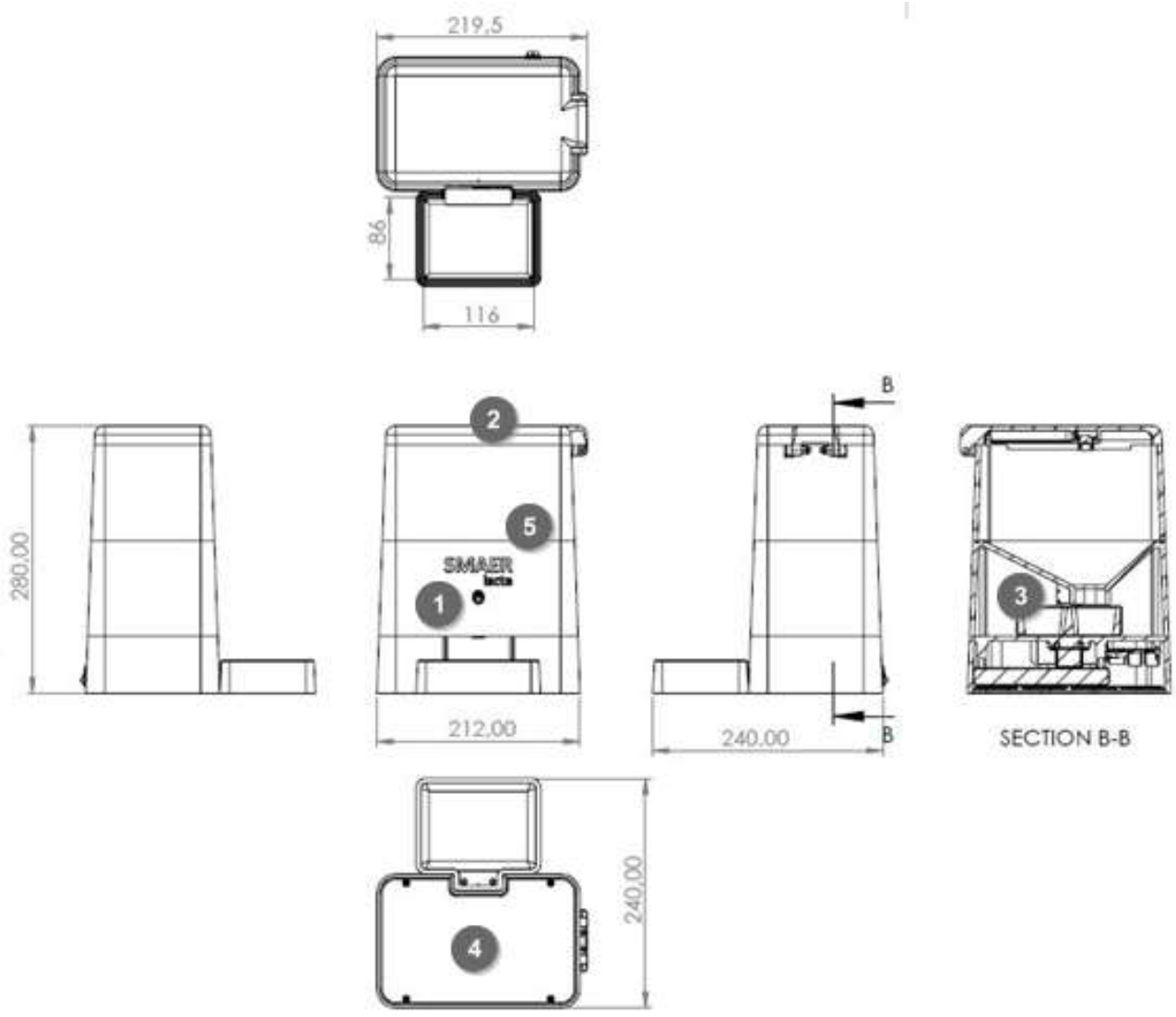


Figura 31 – Dimensões do SMAER

3.2.1 Impressão 3D

Relativamente à impressão das peças em 3D foram utilizadas duas impressoras, modelo Ender 3 Pro, conforme apresentado na figura 32.

No total, foram necessárias 208 horas para imprimir todas as peças usadas neste projeto, onde foram também utilizados dois tipos de materiais:

- **PETG –** para impressão da base (compartimento número 4 da figura 31) e do sistema de rotação. Devido ao esforço destas duas peças, foi escolhido o PETG devido à sua maior resistência comparado com o PLA.

- **PLA –** restante peças embora também necessitem de resistência, o seu esforço será menor pois apenas irá servir para guardar a Ração.

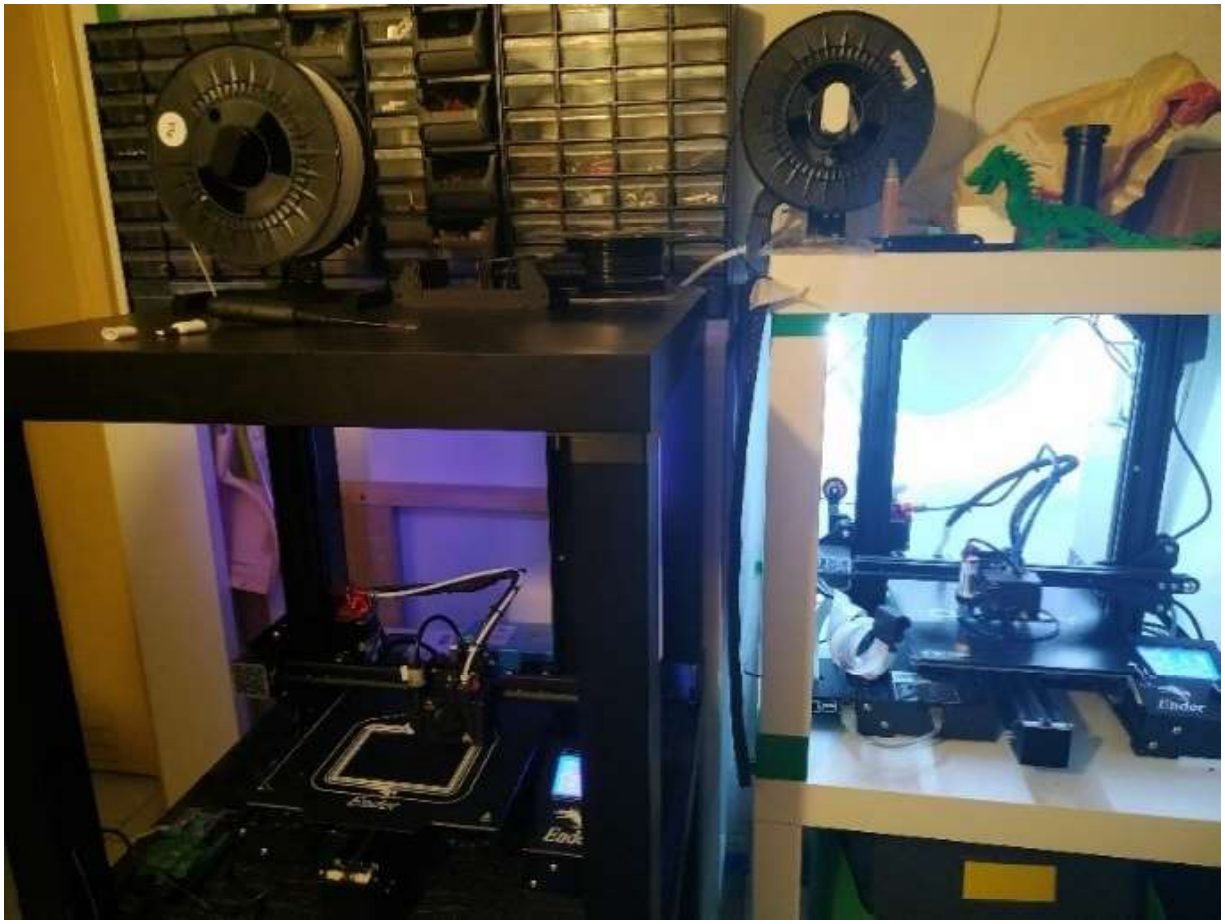


Figura 32 – Impressoras 3D Ender 3 PRO

Em termos definições de impressão, a principal diferença foi na temperatura utilizada, pois o material PETG requer uma temperatura mais elevada para imprimir que o PLA.

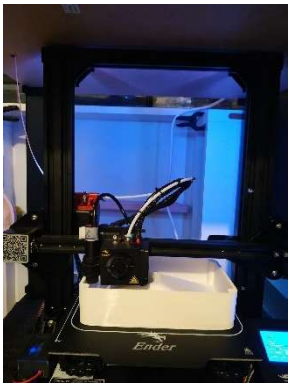
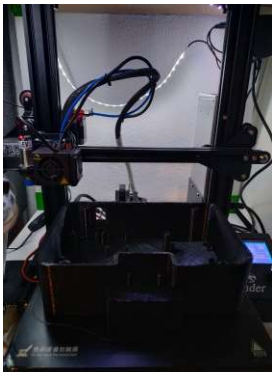
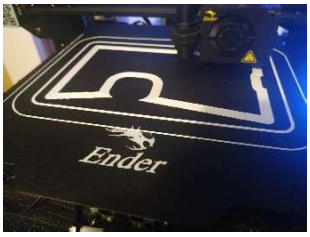
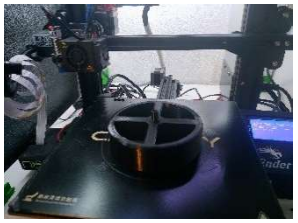
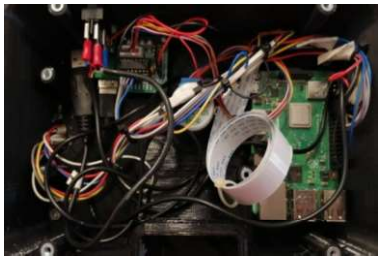
As restantes definições são iguais, o que permite também comparar a resistência dos materiais após a impressão.

Tabela 17 – Definições de Impressão 3D

Definições PLA	Definições PETG
• *Nozzle*: 0.4mm • Layer Height: 0.28mm • Espessura das Linhas: 0.4mm • Densidade: 25% • Temperatura do *Nozzle*: 230º C • Temperatura da Cama: 60 • Velocidade de Impressão: 40mm/s	• *Nozzle*: 0.4mm • Layer Height: 0.28mm • Espessura das Linhas: 0.4mm • Densidade: 25% • Temperatura do *Nozzle*: 240º C • Temperatura da Cama: 70 • Velocidade de Impressão: 40mm/s

Na tabela 18 são apresentadas algumas imagens do processo de impressão 3D, assemblagem e instalação dos componentes eletrónicos utilizados neste projeto.

Tabela 18 – Impressão 3D

 Base	 Impressão do Copo	 Impressão da Base
 Copo	 Impressão do Funil	 Roda com divisórias
 Sistema de Rotação	 Sensores de Temperatura/Humidade/ Ultrasónico	 Base e Funil
 Instatação de Motor e RPi3	 Funil	 Cablagem

 Todas as peças impressas	 SMAER Assemblado

3.3. Software

Um dos objetivos desta arquitetura é que fosse compatível com o maior número de *hardware* possível, quer ao nível das placas controladoras (RPi, Arduino, etc), quer sensores e também com os dispositivos móveis *Android* e *IOS*.

Após algumas pesquisas, verificou-se que existe uma plataforma designada de *Blynk* [44] desenvolvida e desenhada especialmente para dispositivos IoT. Trata-se de uma plataforma totalmente *open-source* e com uma grande comunidade de *developers*. É sem dúvida uma boa opção de utilização, não havendo assim a necessidade de desenvolver um *Broker* ou até mesmo aplicação *Mobile* de raiz.

Esta plataforma inclui três componentes principais conforme demostrado na figura 33:

- ***Blynk APP***: Aplicação disponível para *Android* e *IOS;*
- ***Blynk Server***: Responsável por interligar os dispositivos móveis ao *Hardware*;
- ***Blynk Libraries***: São as bibliotecas disponíveis para a maioria das placas controladoras, que permitem enviar e receber comandos.

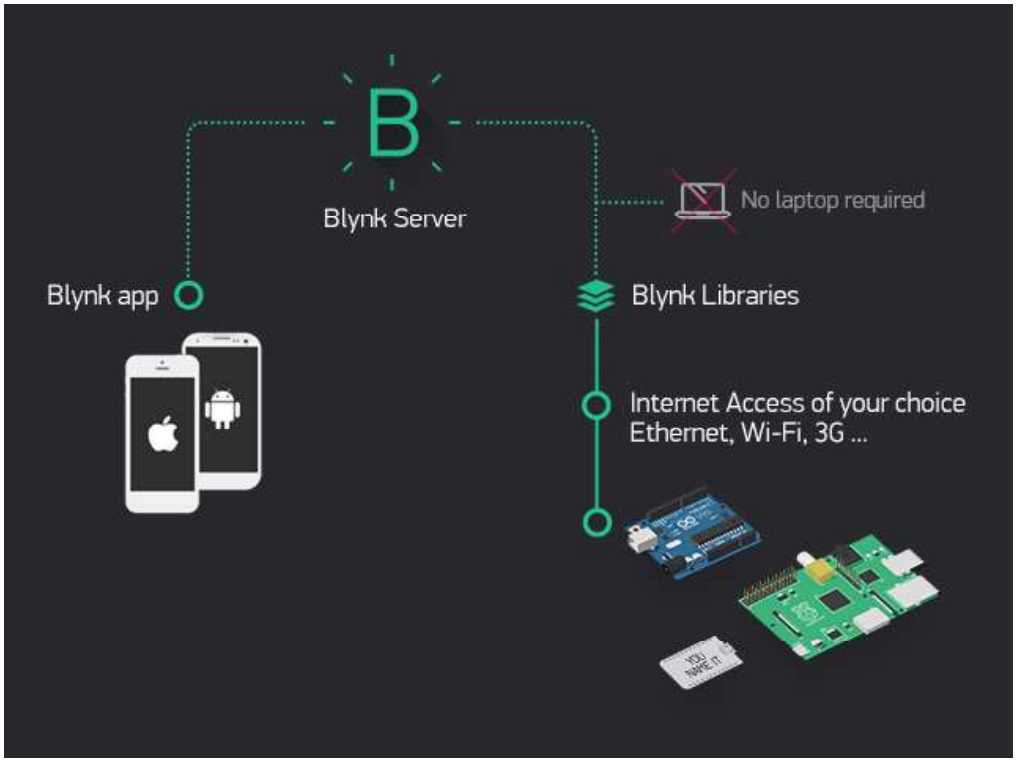


Figura 33 – Componentes da Plataforma Blynk [44]

3.3.1 Servidor

O *Blynk Server* [45] é um Servidor *Java* baseado em *Netty* [46] totalmente em código aberto. Este Servidor é responsável por encaminhar as mensagens entre a Aplicação *Mobile* e as placas controladoras como RPi, Arduino etc.

Antes do envio de comandos entre a aplicação e o Servidor é necessário conectar-se ao Servidor e proceder à autenticação.

No caso a aplicação *Mobile*, após efetuada autenticação com o Servidor com recurso ao *Token*, é criado um túnel encriptado para a troca de mensagens.

Na figura 34 é apresentada a troca de mensagens com o Servidor no momento da autenticação.

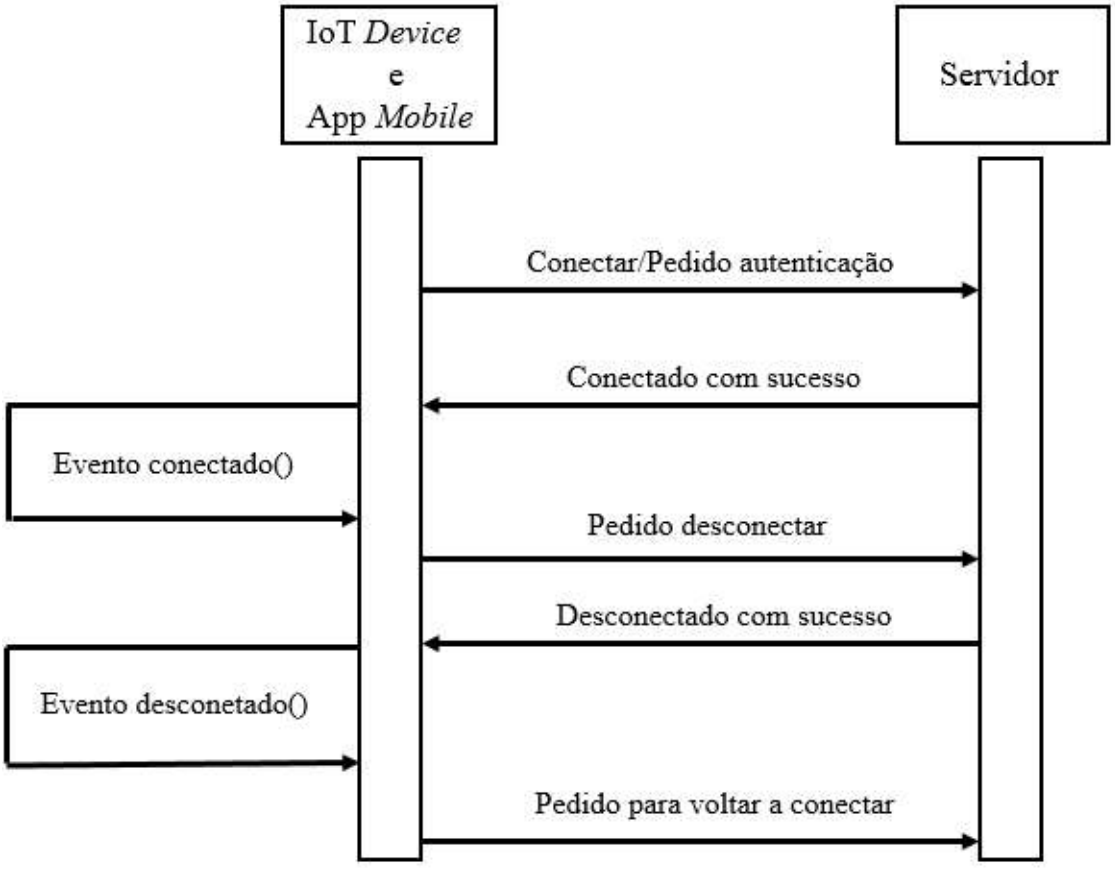


Figura 34 – Conexão ao Servidor

O dispositivo IoT ou aplicação *Mobile* tenta conectar-se ao Servidor através do endereço IP, Porta e respetivo *Token*.

O Servidor valida os dados e confirma ou não a autenticação. Em caso afirmativo, informa o "Cliente" e este já pode enviar comandos.

Posteriomente, todas as mensagens passam a ser enviadas para o Servidor. Este é responsável por comunicar com os diversos IoT *Devices* instalados e com as Aplicações *Mobile* conectadas a si.

O Servidor utiliza um mecanismo de GPIOs virtuais de forma a identificar univocamente cada *Widget* da aplicação *Mobile*.

Na figura 35 é possível demonstrar como funciona este processo. A aplicação *Mobile* solicita ao Servidor o estado do *hardware*, e este por sua vez solicita a informação ao IoT *Device*. Estes pedidos encontram-se em ciclo infinito, sendo possível ajustar a periocidade dos mesmos de forma a atualizar constantemente os dados na aplicação *Mobile*.

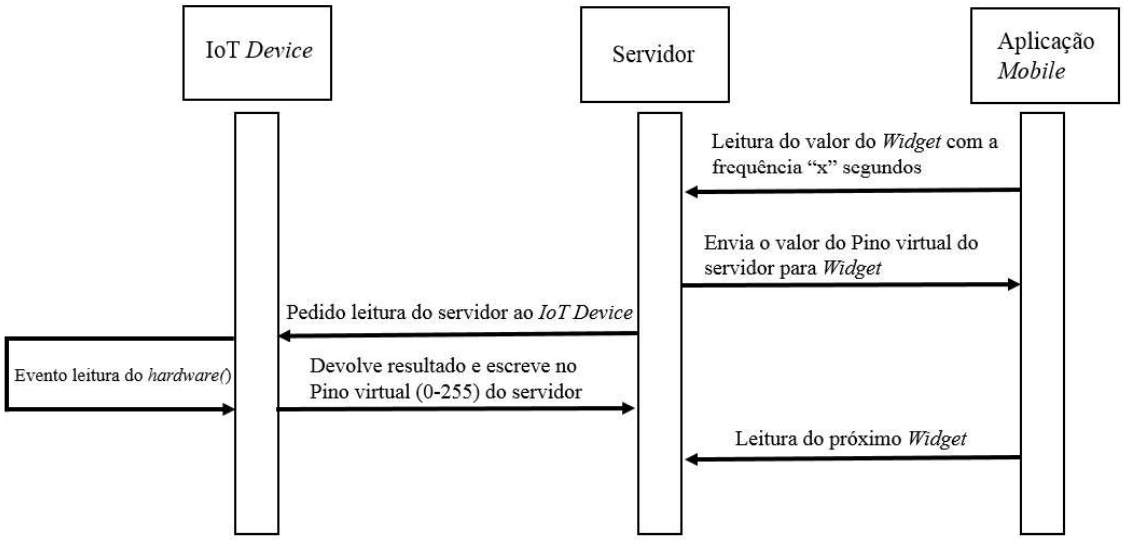


Figura 35 – Fluxo de Leitura do Hardware

Na figura 36 é possível analisar com maior detalhe como se processa a atualização de um GPIO num IoT *Device*.

O utilizador através da aplicação *Mobile* altera o valor do GPIO, o pedido é enviado para um GPIO virtual no Servidor, por sua vez, este envia o pedido para o IoT *Device* que irá executar o *script* previamente configurado.

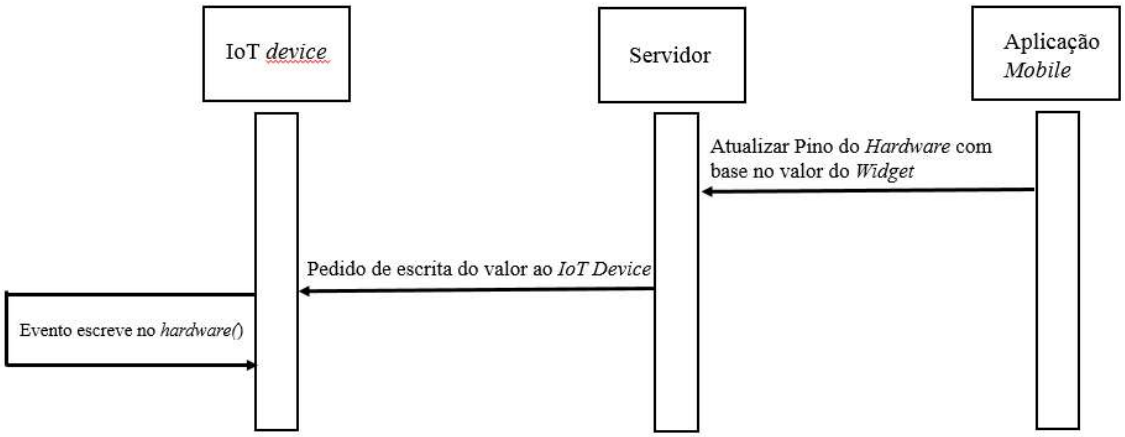


Figura 36 – Fluxo de Escrita no Hardware

O Servidor inclui uma página de administração que permite de forma rápida e fácil saber quais são as aplicações e os *Devices* que estão conectados ao *Broker* em cada momento.

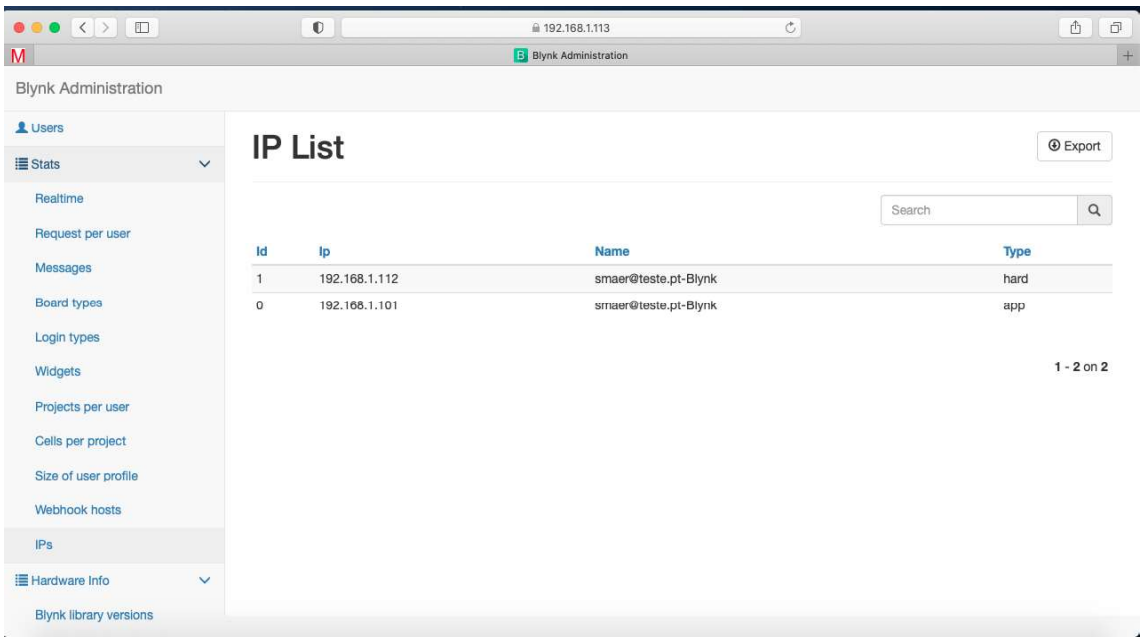


Figura 37 – Administração do Servidor

3.3.2 Scripts

Para que os sensores comuniquem com o RPi e este por sua vez comunique com o Servidor, foi necessário desenvolver dois *scripts* de raiz.

Ambos os *scripts* precisam de estar em permanente execução no SMAER para que tudo funcione, pois estes dois programas estão constantemente a obter os dados dos sensores e a enviá-los para o Servidor (*Broker*).

O *script* desenvolvido em *Python* (figura 38 e código disponível no Anexo C), para além de iniciar a conexão ao Servidor para o envio dos dados, possui também as capacidades de controlo do motor elétrico e a leitura da distância proveniente do Sensor Ultrassónico.

A escolha da linguagem de programação *Python* em detrimento de outras linguagens existentes, teve essencialmente a ver com a sua simplicidade de código e por ser uma linguagem nativa do RPi. Para além deste factor, permite extensibilidade dos seus recursos, ou seja, integrar componentes .NET e Java e ainda importar bibliotecas de C++.

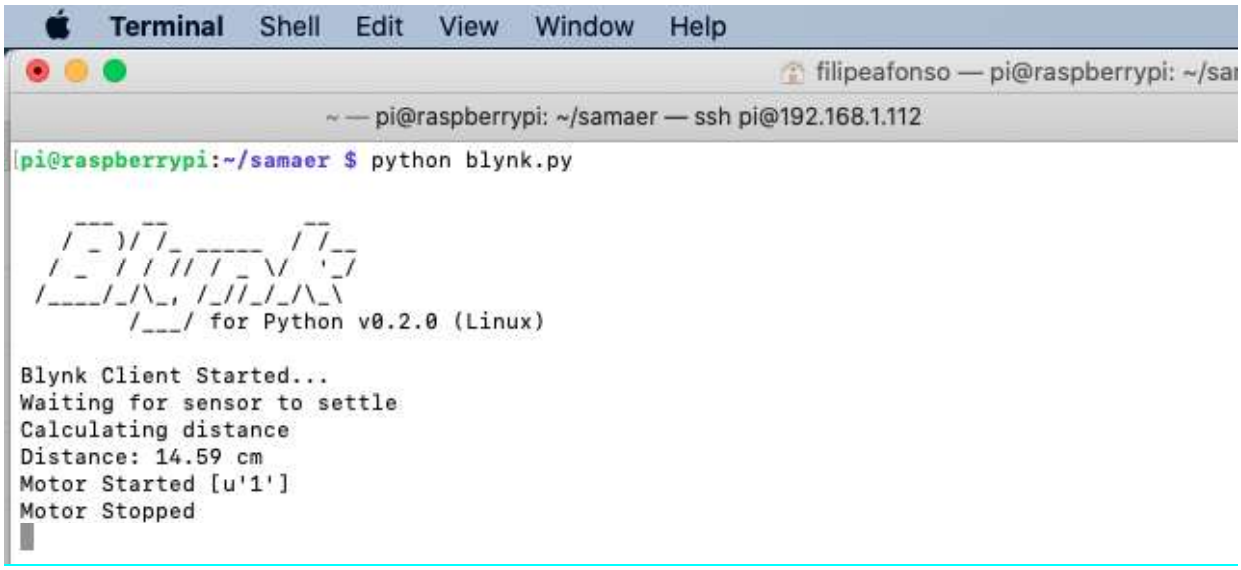


Figura 38 – Script em Python

O segundo *script* foi desenvolvido na linguagem Node.js [47]. É responsável por recolher e enviar as informações do sensor de temperatura e humidade (figura 39 e código disponível no Anexo C).

Na verdade era possível apenas utilizar um único *script* para controlar os sensores e enviar os dados para o Servidor. No entanto, a escolha de dois *scripts* foi meramente opcional e deveu-se sobretudo por desafio para testar o comportamento do sistema para duas linguagens diferentes e também para comprovar que o Servidor permite diversos clientes e ligações em simultâneo.

Para ambos os casos foi necessário proceder à instalação de algumas bibliotecas para garantir todas as funcionalidades necessárias ao bom funcionamento do sistema.

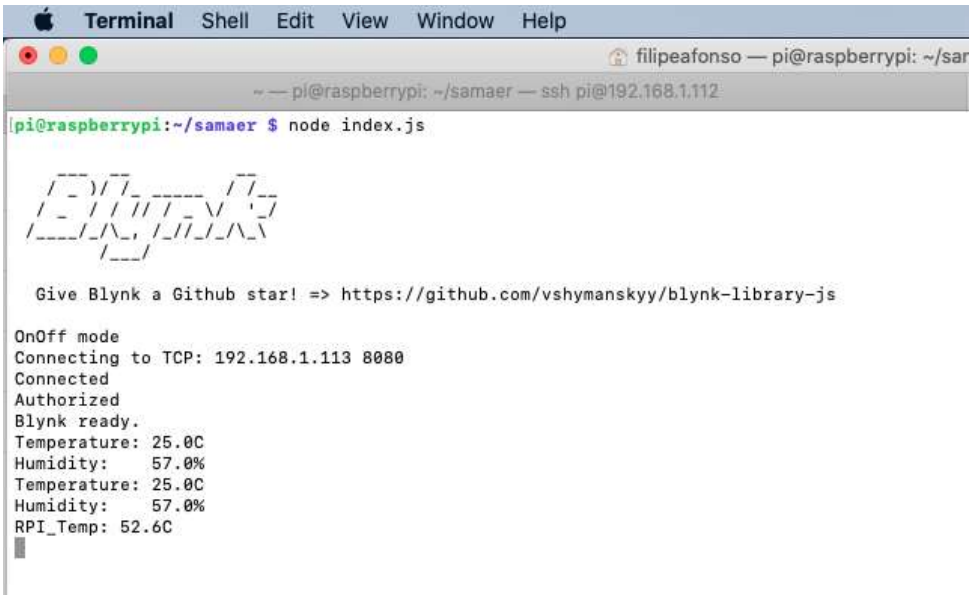


Figura 39 – Script em Nodejs

3.3.3 Aplicação *Mobile*

A aplicação *Mobile* é sem dúvida um ponto forte de todo o sistema. A capacidade de controlar o SMAER através de uma *APP,* permite geri-lo através da Internet em qualquer parte do mundo. Para além disso, a aplicação é bastante flexível o que permite de forma fácil e intuitiva adicionar novas funcionalidades (botões, *widgets*, *display*) sem que o utilizador necessite de programar código na aplicação (Anexo B). Adicionalmente, outro ponto forte é a capacidade de controlar diversos dispositivos IoT com a mesma aplicação *Mobile* e por vários utilizadores em simultâneo.

Conforme é ilustrado na figura 40, a aplicação *Mobile* apresenta as principais funcionalidades:

1. Visualização da Temperatura – com base no sensor DHT11 instalado no topo do SMAER, é possível visualizar o valor da temperatura dentro da caixa.

2. Botão "Dar" – ao pressionar este botão é ativado o motor elétrico, que por sua vez faz girar o mecanismo de rotação para dispensar Ração para o prato.

3. Com base na informação enviada pelo Sensor Ultrassónico instalado no topo, é calculada a distância entre o cimo e o fundo do recipiente onde é armazenada a Ração. Assim sendo, quanto maior for a distância entre os dois pontos, menor será a quantidade de Ração presente no recipiente. Já quando acontece o contrário, quanto menor for a distância entre os pontos, mais Ração contém o recipiente. O valor é apresentado em forma de percentagem.

4. Visualização da Humidade – permite visualizar a humidade presente dentro do recipiente onde é colocada a Ração.

5. Botão "Agendar Refeição" – possibilita definir uma hora para um dispensar automático da Ração.

6. Camera – proporciona a visualização em tempo real das imagens capturadas pelo SMAER.

7. Histórico – permite visualizar o histórico dos valores de temperatura e humidade registados pelo sensores, existindo a possibilidade de enviar os dados por *email* em formato CSV.

8. Permite visualizar a temperatura do RPi que controla todos os sensores.

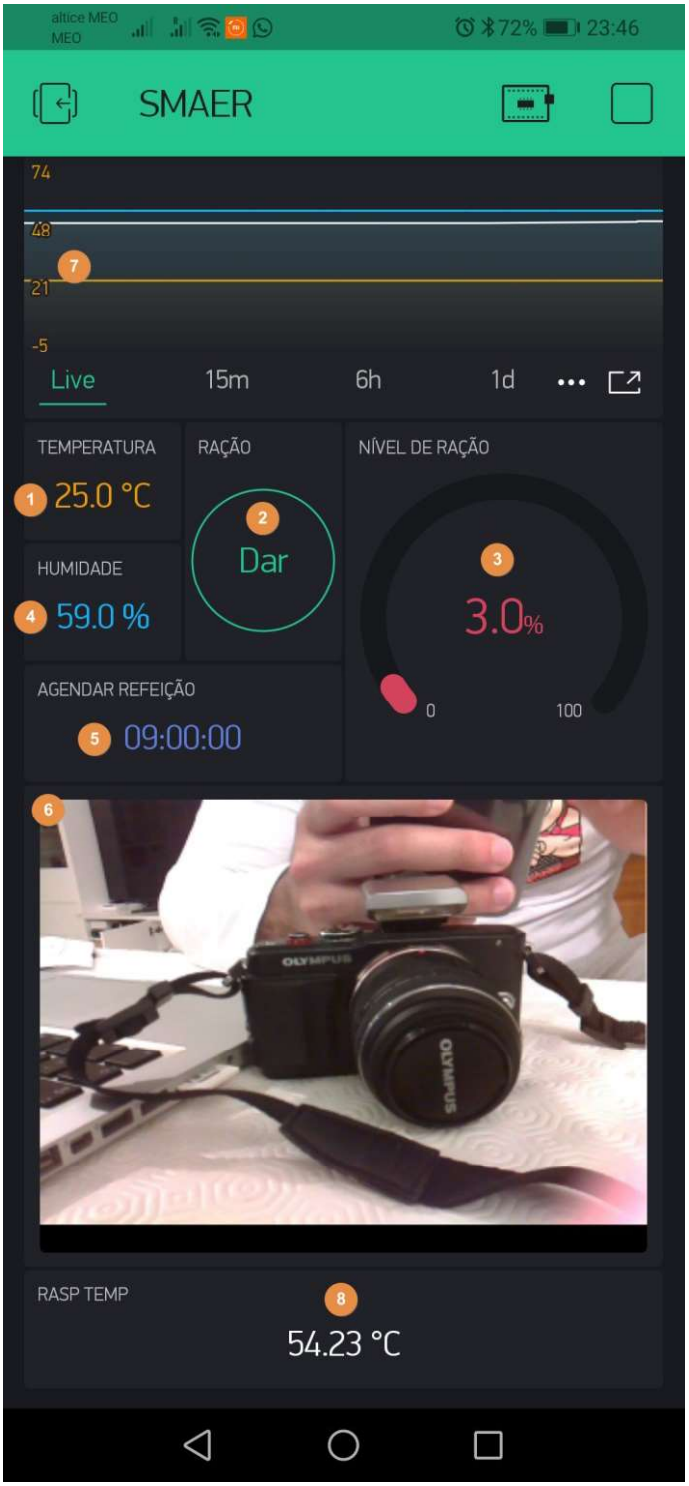


Figura 40 – Aplicação SMAER

3.3.4 Segurança

Relativamente à segurança das comunicações entre os dispositivos IoT e aplicação *Mobile*, como já foi referido anteriormente, é utilizada uma ligação criptografada através do protocolo TLS/SSL.

Para além disso, no momento de criação de um novo projeto IoT, na aplicação *Mobile* é gerado automaticamente um *Token* conforme ilustrado na figura 41. Este identificador é único e necessário para conectar o *hardware* ao *smartphone*.

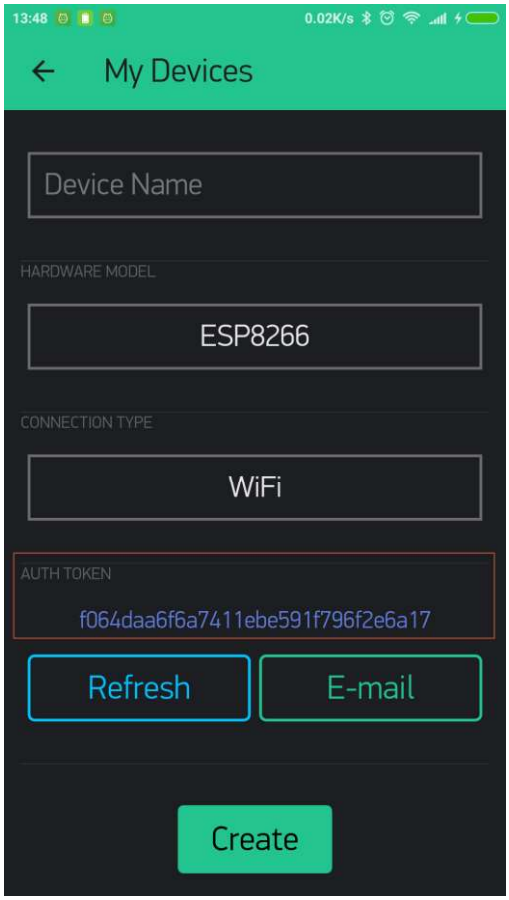


Figura 41 – Token

No que diz respeito ao Servidor, estão disponíveis duas portas:

- **8080 –** Protocolo TCP para ligação dos dispositivos IoT e o Servidor (sem encriptação).
- **9443 –** Protoloco TLS/SSL para aplicação Mobile e o Servidor.

Capítulo 4 – Análise e Discussão dos Resultados

Para comprovar a funcionalidade prática do sistema desenvolvido, foi implementado um cenário de testes com a utilização do *software* e *hardware* mencionado neste projeto. Com a utilização de um *Router* 3G foi criada uma rede interna para interligar os três componentes, conforme ilustrado na figura 42.

O protótipo SMAER está funcional e pode ser visualizado neste vídeo de apresentação: https://youtu.be/73TMFNqaFa8

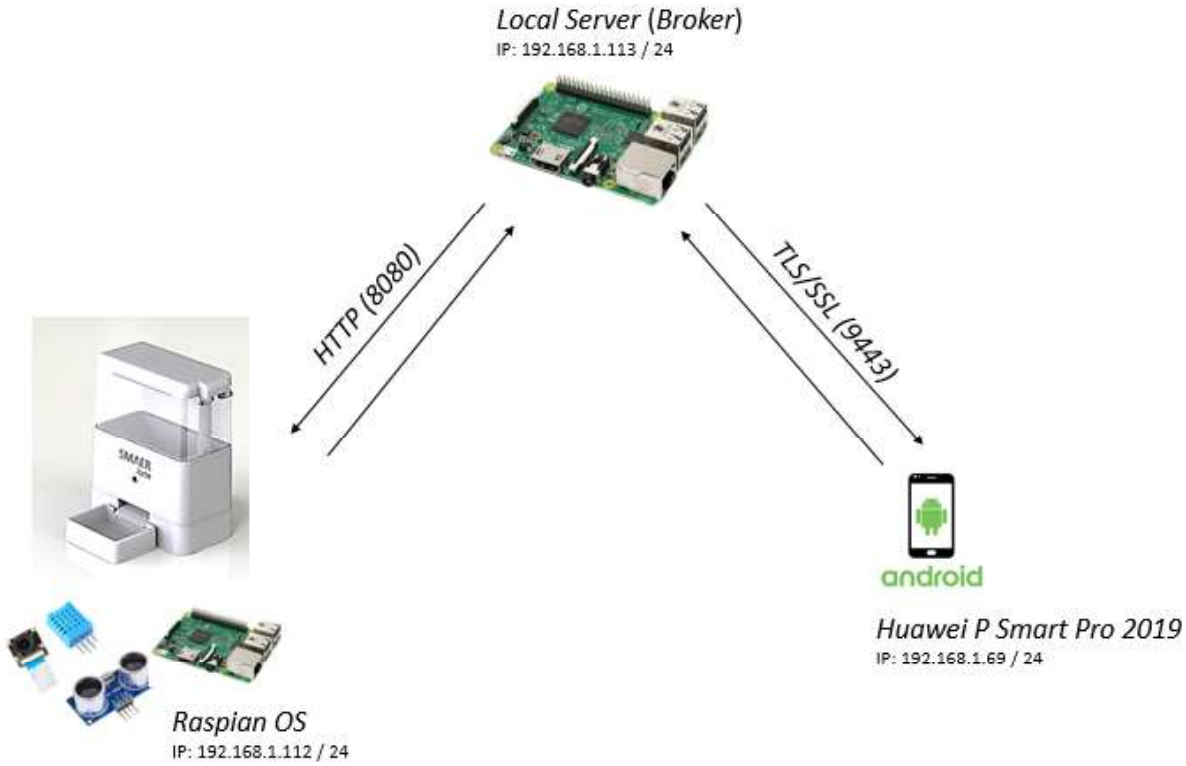


Figura 42 – Cenário de Testes

Previamente, conforme ilustrado na figura 43, todos os componentes foram testados antes da sua instalação no dispensador de Ração, de modo a garantir a sua operacionalidade.

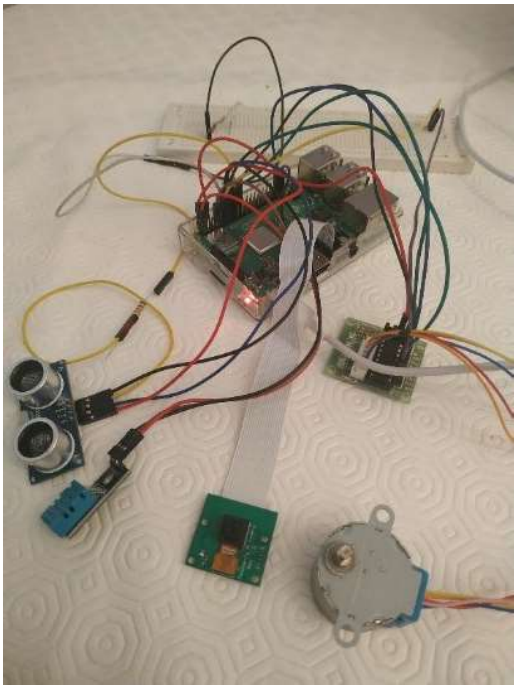
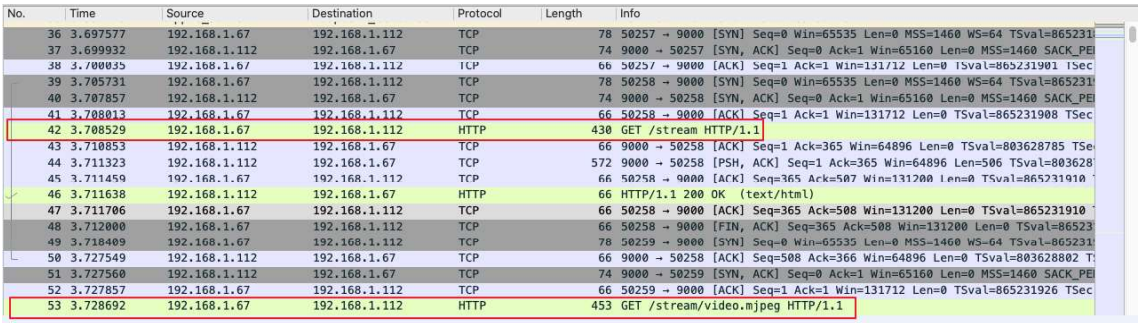


Figura 43 – Testes ao Hardware

Com ajuda do *software Wireshark* [48] que permite analisar e capturar o tráfego na rede de pacotes, foi possível testar e verificar duas situações:

A primeira diz respeito à camera utilizada no RPi. Verificou-se que ao aceder diretamente ao endereço HTTP do *stream* de vídeo era possível obter as imagens da camera, o que indicava problemas de segurança, pois qualquer pessoa com o endereço *web* poderia aceder directamente ao vídeo sem qualquer controlo ou validação do utilizador.

Na imagem 44 podemos verificar os pacotes do *stream* de vídeo através do protocolo Http sem qualquer proteção.



No.	Time	Source	Destination	Protocol	Length	Info
36	3.697577	192.168.1.67	192.168.1.112	TCP	78	50257 → 9000 [SYN] Seq=0 Win=65535 Len=0 MSS=1460 WS=64 TSval=865231
37	3.699932	192.168.1.112	192.168.1.67	TCP	74	9000 → 50257 [SYN, ACK] Seq=0 Ack=1 Win=65160 Len=0 MSS=1460 SACK_PE
38	3.700035	192.168.1.67	192.168.1.112	TCP	66	50257 → 9000 [ACK] Seq=1 Ack=1 Win=131712 Len=0 TSval=865231901 TSec
39	3.705731	192.168.1.67	192.168.1.112	TCP	78	50258 → 9000 [SYN] Seq=0 Win=65535 Len=0 MSS=1460 WS=64 TSval=865231
40	3.707857	192.168.1.112	192.168.1.67	TCP	74	9000 → 50258 [SYN, ACK] Seq=0 Ack=1 Win=65160 Len=0 MSS=1460 SACK_PE
41	3.708013	192.168.1.67	192.168.1.112	TCP	66	50258 → 9000 [ACK] Seq=1 Ack=1 Win=131712 Len=0 TSval=865231908 TSec
42	3.708529	192.168.1.67	192.168.1.112	HTTP	430	GET /stream HTTP/1.1
43	3.710853	192.168.1.112	192.168.1.67	TCP	66	9000 → 50258 [ACK] Seq=1 Ack=365 Win=64896 Len=0 TSval=803628785 TSe
44	3.711323	192.168.1.112	192.168.1.67	TCP	572	9000 → 50258 [PSH, ACK] Seq=1 Ack=365 Win=64896 Len=506 TSval=803628
45	3.711459	192.168.1.67	192.168.1.112	TCP	66	50258 → 9000 [ACK] Seq=365 Ack=507 Win=131200 Len=0 TSval=865231910
46	3.711638	192.168.1.112	192.168.1.67	HTTP	66	HTTP/1.1 200 OK (text/html)
47	3.711706	192.168.1.67	192.168.1.112	TCP	66	50258 → 9000 [ACK] Seq=365 Ack=508 Win=131200 Len=0 TSval=865231910
48	3.712000	192.168.1.67	192.168.1.112	TCP	66	50258 → 9000 [FIN, ACK] Seq=365 Ack=508 Win=131200 Len=0 TSval=86523
49	3.718409	192.168.1.67	192.168.1.112	TCP	78	50259 → 9000 [SYN] Seq=0 Win=65535 Len=0 MSS=1460 WS=64 TSval=865231
50	3.727549	192.168.1.112	192.168.1.67	TCP	66	9000 → 50258 [ACK] Seq=508 Ack=366 Win=64896 Len=0 TSval=803628802 T
51	3.727560	192.168.1.112	192.168.1.67	TCP	74	9000 → 50259 [SYN, ACK] Seq=0 Ack=1 Win=65160 Len=0 MSS=1460 SACK_PE
52	3.727857	192.168.1.67	192.168.1.112	TCP	66	50259 → 9000 [ACK] Seq=1 Ack=1 Win=131712 Len=0 TSval=865231926 TSec
53	3.728692	192.168.1.67	192.168.1.112	HTTP	453	GET /stream/video.mjpeg HTTP/1.1

Figura 44 – Wireshark Http

Posteriomente, foram capturados pacotes entre o *Smartphone* e o Servidor e verificou-se efetivamente que os dados são enviados através do protocolo TLS entre ambos, não sendo possível consultar qualquer informação do tipo de dados que são enviados entre o utilizador e o Servidor. Desta forma, é garantida a privacidade dos dados e dos utilizadores.

A *stream* de vídeo uma vez que é processada dentro da aplicação *Mobile*, acaba por ser igualmente encriptada.

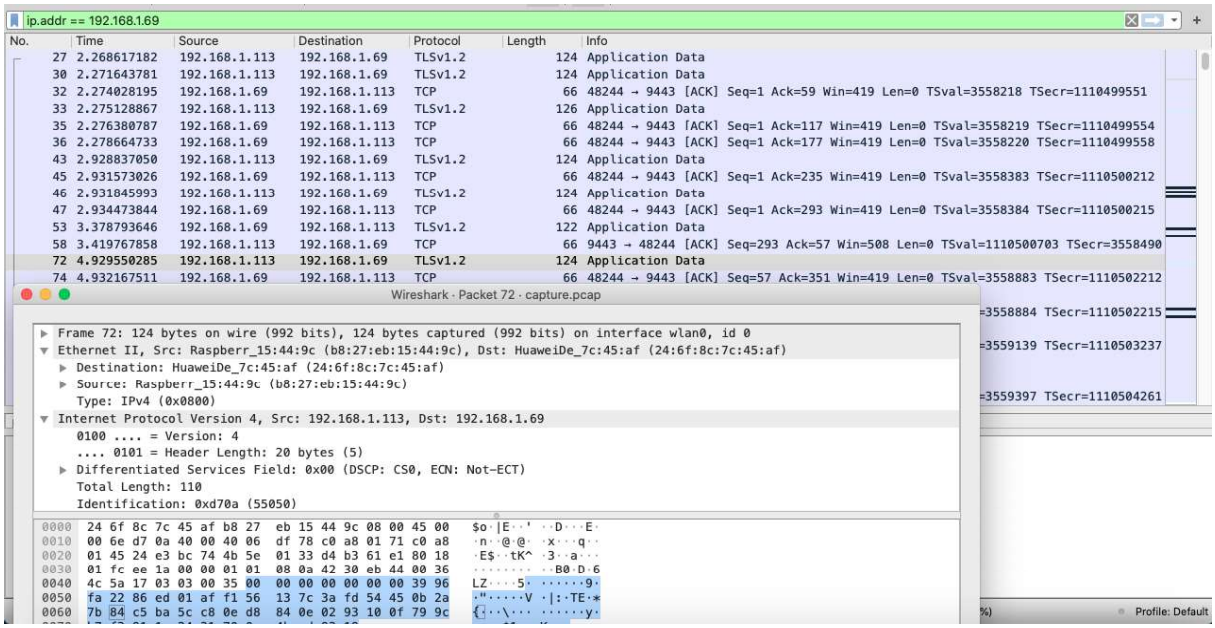


Figura 45 – Wireshark TLS/SSL

4.1. Consumo Energético

Um dos fatores a ter em conta e bastante importante num sistema IoT é o baixo consumo energético dos dispositivos utilizados, pois tipicamente são sistemas de pequena dimensão, o que não permite a utilização de baterias de grande volume. Além disso operam por vezes de forma remota em que a poupança de energia ajuda a prolongar o equipamento ativo durante mais tempo.

Com ajuda de um multímetro foram realizados alguns ensaios para obter o consumo de energia do sistema SMAER. O consumo foi medido com todos os sensores e camera ligados, rede *Wi-fi* ativa e em comunicação com servidor, motor elétrico em funcionamento. Desta forma, conseguimos obter o pico máximo do consumo de energia conforme indicado na figura 46.

A bateria utilizada neste projeto apresenta um *output* máximo de 2.4A (secção 3.1.6). Uma vez que o consumo de corrente no pior caso apresenta um valor de 0.840A, conclui-se que a bateria tem capacidade suficiente para alimentar todo o sistema sem problemas.

A tensão também foi medida nas mesmas condições (figura 47), obtendo-se um valor de 4,68V × 0,840A = 3,93 Watts.

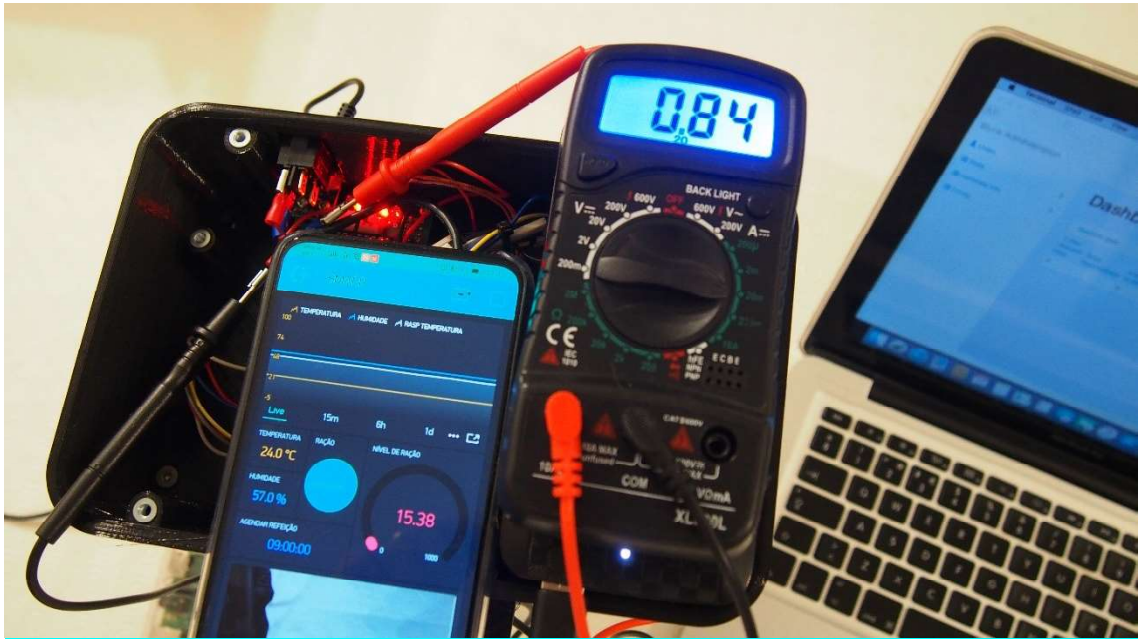


Figura 46 – Consumo energético do SMAER

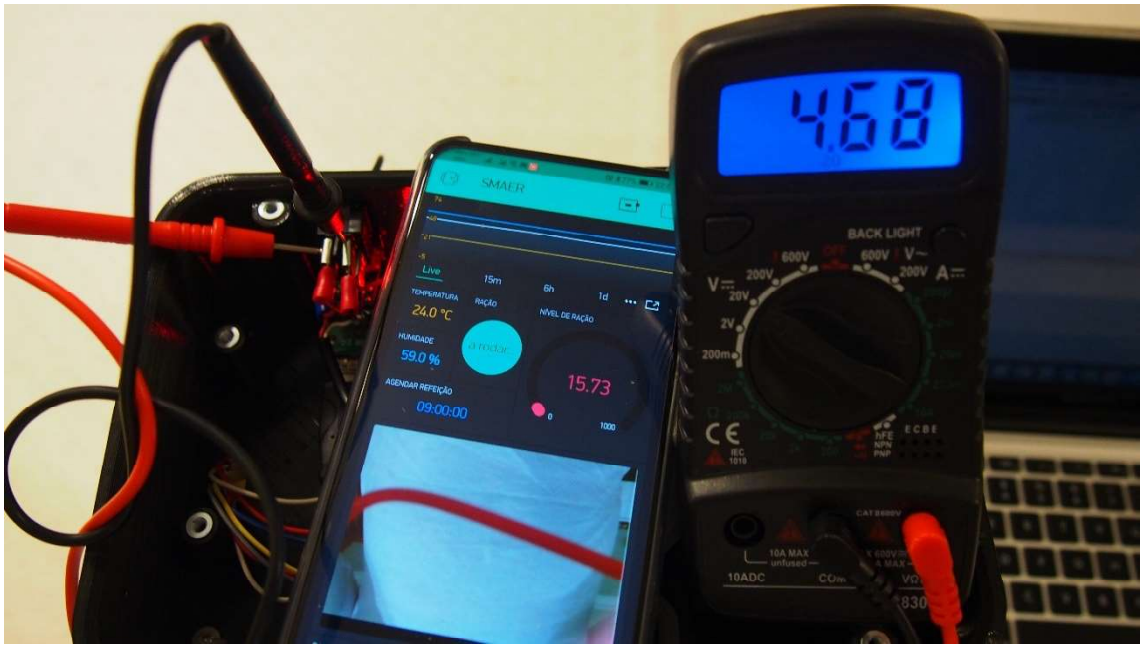


Figura 47 – Tensão de Corrente

O cálculo do tempo estimado de uma bateria é dado pela seguinte fórmula:

$$Tempo\ estimado\ bateria = \frac{Capacidade\ da\ Bateria}{Consumo\ do\ Dispositivo}$$

Introduzindo na equação os valores mencionados anteriormente (bateria - secção 3.1.6), chegamos ao seguinte resultado:

$$Tempo\ estimado\ bateria = \frac{8000\ mAh}{840\ mAh} \cong 9{,}5\ horas$$

Admitindo que a *Powerbank* tenha perdido alguma da sua capacidade inicial desde que veio de fábrica, e pelo fato de que o consumo medido com o multímetro é no pico da utilização, ou seja, não será constante este valor, assim sendo em caso de falha de AC, podemos concluir que o sistema irá continuar a funcionar durante pelo menos 9 horas. Numa situação normal, será tempo suficiente para o fornecedor de energia consiga resolver uma eventual avaria.

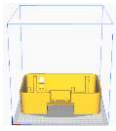
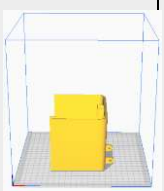
4.2. Custos

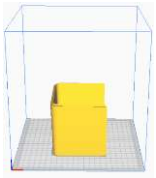
Relativamente aos custos do protótipo SMAER, podemos dividi-los em duas partes: a Impressão 3D e o *Hardware*.

No que diz respeito à impressão 3D, neste caso particular, apenas é apresentado o custo dos filamentos utilizados. O custo de eletricidade gasto pelas impressoras 3D não esta a ser contabilizado porque não foi possível medir, mas será um valor reativamente baixo, quase irrelevante. Foram utilizados dois materiais diferentes neste projeto mas os preços do PLA e PETG foram exatamente iguais.

Na tabela 19 é apresentado com maior detalhe cada peça impressa e o seu custo associado.

Tabela 19 – Custos da Impressão 3D

Peça	Tipo de Material	Tempo de Impressão	Quantidade (gramas)	Filamento (metros)	Custo (12€/ 1Kg)
	PLA	65h29m	550g	165,71m	6,6€
	PETG	43h29m	383g	115,43m	4,6€
	PLA	21h58m	208g	62,57m	2,5€
	PLA	17h30m	108g	32,59m	1,30€
	PLA	17h27m	145g	43,78m	1,74€

	PLA	16h11m	135g	40,66m	1,62€
	PETG	14h34m	73g	22,13m	0,88€
	PLA	8h18m	74g	22,32m	0,89€
	PLA	3h01m	20g	5,99m	0,24€
Total		*207h55m (aprox. 9 dias)*	*1696g*	*511,18m*	***20,36 €***

Relativamente à segunda parte, o *Hardware* utilizado neste projeto teve como referência a loja *Mauser* [49].

Na tabela 20 é apresentado em detalhe o custo de cada componente à data da elaboração desta dissertação.

Tabela 20 – Custos do Hardware

Equipamento	Referência Mauser	Preço c/ Iva
Raspberry Pi 3	096-5700	39,93€
Raspberry Pi Cam	096-7650	28,20€
Motor de passo 5VDC (ULN2003)	096-8831	3,99€
Cabos Jumper	096-5810	3,15€
Sensor Temperatura e Humidade (DTH-11)	096-7807	2,75€
Sensor Ultrassónico (HC-SR04)	096-6220	2,10€
Total	-	**80,12€**

Neste cenário, relativamente ao Servidor (*Broker*) foi também utilizado um RPi 3, no entanto este componente é facultativo uma vez que é possível utilizar a *Blynk Cloud* ou utilizar um qualquer outro computador para fazer de servidor.

Capítulo 5 – Conclusões e Recomendações

5.1. Principais Conclusões

Nesta dissertação projetou-se e implementou-se um sistema flexível que permite monitorizar animais de estimação remotamente e em tempo real, utilizando uma abordagem orientada para a IoT e impressão 3D.

Esta arquitetura permite ao utilizador expandir e controlar múltiplos dispositivos IoT através da mesma aplicação *Mobile*. Possibilita igualmente adicionar de forma fácil e rápida novas funcionalidades (sensores, botões, *displays*) ao dispositivo IoT sem ter que programar código na aplicação *Mobile*, sendo apenas necessário adicionar os *widgets* à aplicação, configurar o GPIO e programar o código do lado do IoT *Device*. Permite também que diversos utilizadores controlem os vários dispositivos IoT instalados num ambiente de domótica.

Este sistema utiliza *hardware* de baixo custo mas não compromete o seu desempenho, antes pelo contrário. O desenho e a criação do protótipo 3D foram desenvolvidos totalmente de raiz, ficando com um sistema funcional e único.

Em termos de *software* foi utilizado uma plataforma *open-source* desenvolvida sobretudo para dispositivos IoT, o que obrigou a compreender todo o seu funcionamento para poder ser adaptada e utilizada no sistema implementado. Foram também criados dois *scripts* para integração dos sensores utilizados (Sensor de Temperatura e Humidade, Sensor Ultrassónico) e também do Motor.

Em termos de custos, no final, este protótipo rondou os 100 (cem) euros. Este valor pode ser ainda mais baixo caso se abdique do RPi e da camera. No caso de se abdicar da camera é possível ainda utilizar um ESP o que possibilita que o valor desça consideravelmente.

Em termos de consumo energético, o SMAER tem um consumo médio de 3,93 Watts/Hora. Foi instalada uma Bateria (*Powerbank*) com autonomia estimada de 9,5 horas, de forma a garantir a resiliência de todo o sistema em caso de falha de energia do fornecedor.

Todos os objetivos propostos foram atingidos com sucesso.

5.2. Limitações do estudo e principais dificuldades encontradas

Ao longo da dissertação foram encontradas algumas dificuldades, nomeadamente na elaboração do desenho técnico do SMAER. Apesar do meu conhecimento com a ferramenta *SolidWorks* foi necessário algum tempo de aprendizagem para elaborar o protótipo.

Outro aspeto em que também tive dificuldades foi na impressão 3D, desde a forma de imprimir as peças, as medidas das furações, corrigir e acertar as *settings* para uma boa impressão, foram necessárias bastantes horas e algumas peças impressas de teste até acertar.

5.3. Propostas de investigação futura

Apesar do protótipo apresentado estar funcional e de acordo com os objetivos inicialmente propostos, existem sempre oportunidades de melhoria, das quais destaco:

Possibilidade de adicionar novos sensores para novas funcionalidades. Por exemplo, adicionar um sensor de pressão no prato com objetivo de receber uma notificação quando este está vazio e ao mesmo tempo despoletar o processo de libertação da ração automaticamente.

Outra funcionalidade interessante de implementar no sistema seria um mecanismo para detetar situações de escravamento da ração, ou seja, verificar que o motor está em esforço e neste caso deveria rodar no sentido contrário de forma a desencravar a ração presa no SMAER.

Elaborar um protótipo semelhante mas para a água.

Adicionar outros protocolos de comunicação, como Bluetooth LE, LoRaWAN ao sistema.

Integração deste protótipo no sistema Google Home [50] ou Alexa [51] num contexto de Domótica.

Incorporar mecanimos de Inteligência Artificial para deteção e alerta de comportamentos anormais dos animais, tendo em conta o padrão do consumo de ração ao longo do dia.

Anexo A. Configuração RaspberryPi

Para ambos os RPi (SMAER e Servidor) foi instalado o *Raspberry Pi OS*.

Para instalar, é necessário navegar até ao site: https://www.raspberrypi.org/downloads/ e descarregar a última versão do *Raspberry Pi Imager*.

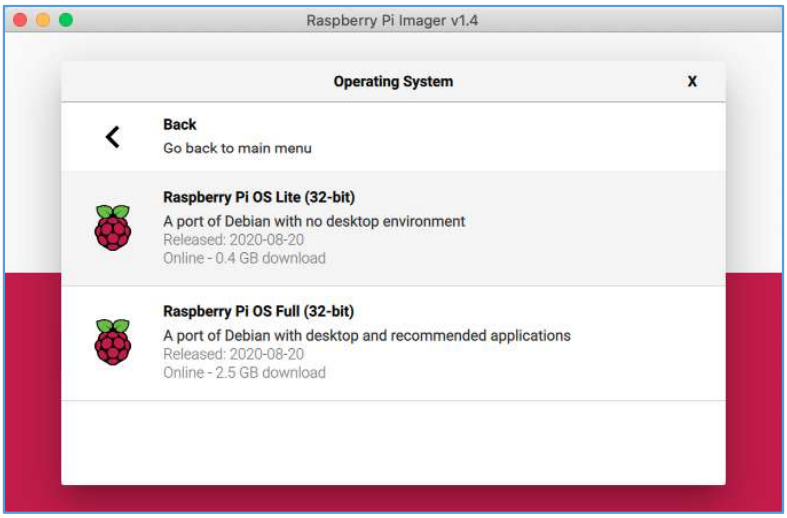


Figura A. 1 – Raspberry Pi OS Escolha de Versão

Posteriormente é necessário escolher a versão (*Lite ou Full*) e selecionar o cartão de memória onde se pretende fazer a instalação do *Raspberry Pi OS*.

Figura A. 2 – Raspberry Pi OS

No final da instalação surge a mensagem que foi criado com sucesso, é necessário colocar o cartão de memória no RPi.

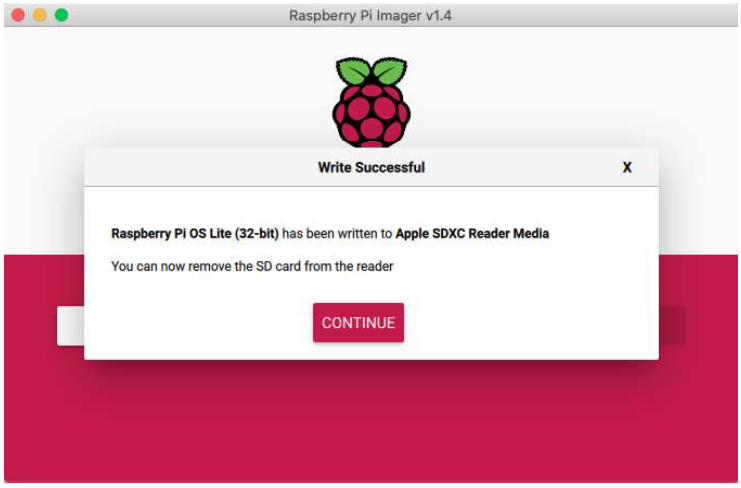


Figura A. 3 – Instalação Terminada

Para aceder ao RPi é necessário utilizar uma ligação SSH (*Secure Shell*). No *MacOSX* é utilizado o *Terminal* e respetivo comando conforme a imagem Figura A.4.

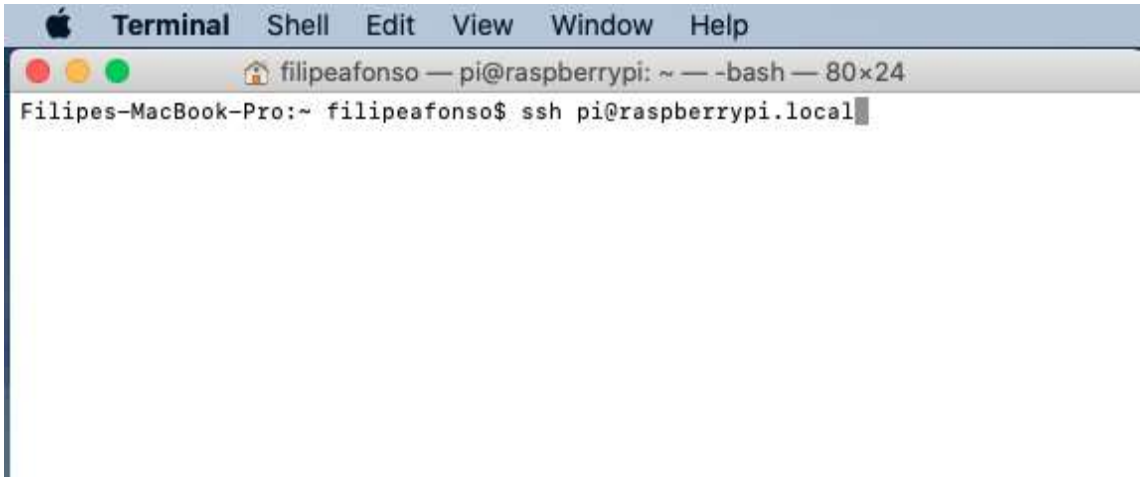


Figura A. 4 – Ligação SSH ao Servidor

Posteriormente é necessário introduzir a palava-chave para se conectar ao RPi.

Figura A. 5 – Introdução da Palavra-Chave

Para aceder ao menu de configuração do RPi é necessário introduzir o seguinte comando:

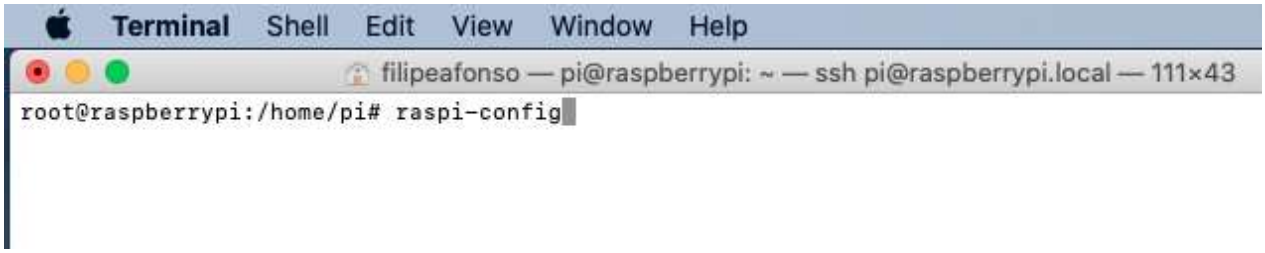


Figura A. 6 – Palavra-chave

Neste menu é possível configurar algumas definições base do RPi, como por exemplo o nome do equipamento, rede Wifi, *overclocking,* idioma do teclado*, GPIOS etc.*

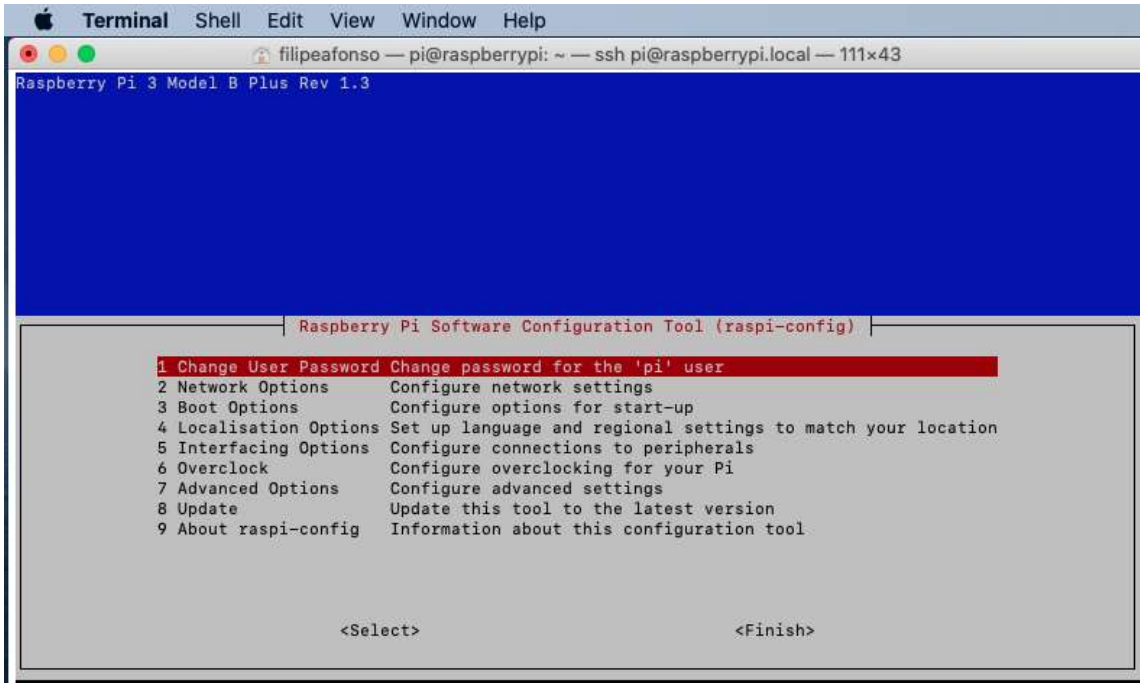


Figura A. 7 – Raspberry Pi Software Configuration Tool

Configuração da Camera

Para o funcionamento da camera, foi instalado o *software* UV4L no SMAER através dos seguintes comandos *Linux*:

Adicionar a chave do repositório do UV4L:

```
curl http://www.linux-projects.org/listing/uv4l_repo/lpkey.asc | sudo apt-key add -
```

Adicionar o repositório ao *Raspberry Pi OS*:

```
echo "deb http://www.linux-projects.org/listing/uv4l_repo/raspbian/stretch stretch main" | sudo tee -a /etc/apt/sources.list
```

Atualizar o *Raspberry Pi OS*:

```
sudo apt-get update
```

Instalar o UV4L:

```
sudo apt-get install -y uv4l uv4l-raspicam uv4l-raspicam-extras uv4l-webrtc uv4l-raspidisp uv4l-raspidisp-extras
```

Verificar e alterar as configurações necessárias:

```
Sudo nano /etc/uv4l/uv4l-raspicam.conf
```

Aceder ao painel de administração do UV4L:

```
Http://192.168.1.112:9000/
```

Figura A. 8 – Painel UV4L

Anexo B. Configuração da Plataforma *Blynk*

Configuração da Aplicação *Mobile*

A aplicação *Blynk* está disponível para *download* na *PlayStore (Android)* e *AppStore (IOS)*. Após instalação é necessário criar uma conta com base num endereço de *email* e *password*.

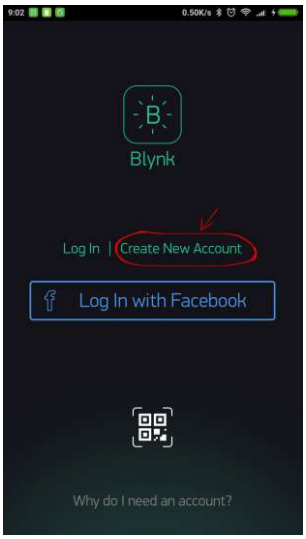


Figura B. 1 – Criar Conta

Posteriormente, criar um projeto:

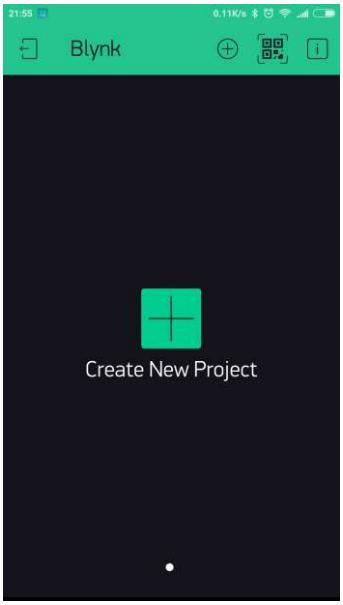


Figura B. 2 – Criar Projeto

Selecionar a placa utilizada no IoT *Device,* neste caso foi o RPi3:

Figura B. 3 – Selecionar a Placa

Gerar o *Token* para configurar no IoT *Device*:

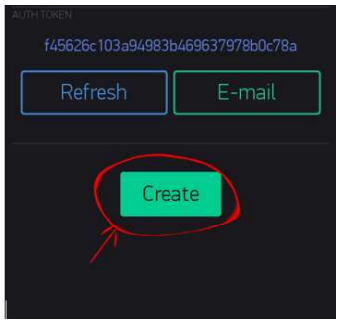


Figura B. 4 – Gerar o Token

Posteriormente, adicionar os *Widgets* ao projeto e programar o código no *script* do SMAER. Desta forma, as funcionalidades do sistema passam a estar disponíveis na Aplicação *Mobile*.

Figura B. 5 – Adicionar Widgets

Configurar os GPIOs virtuais e o tipo de funcionalidade para cada um:

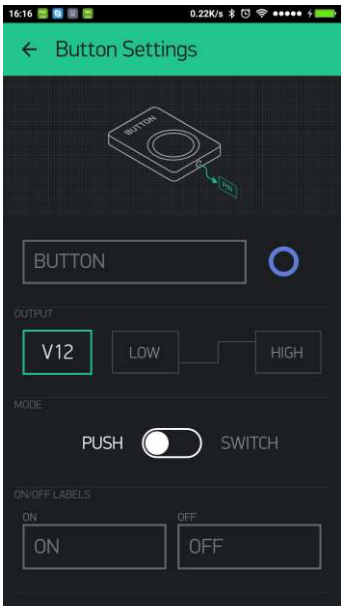


Figura B. 6 – Configurar o GPIO

No final, Aplicação *Mobile* ficará com o seguinte aspeto:

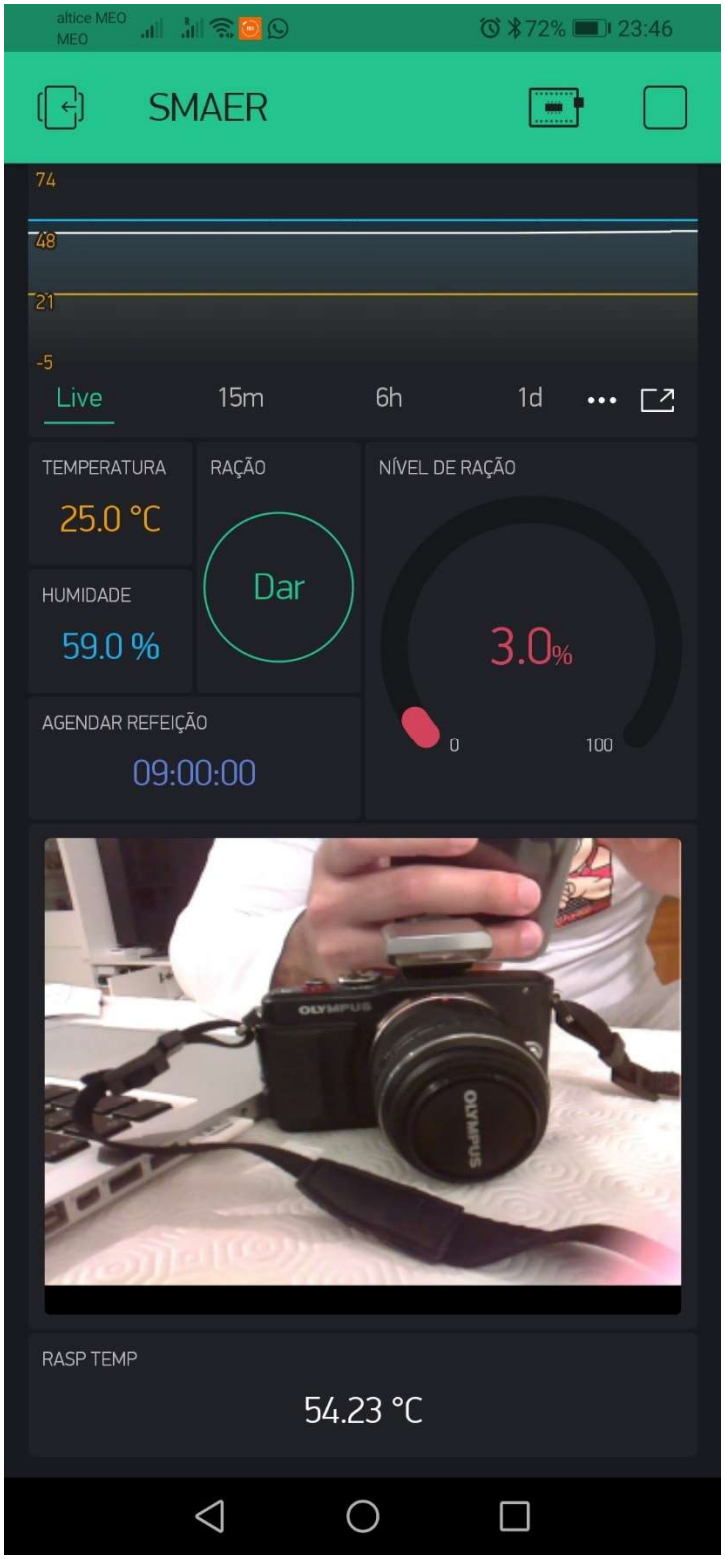


Figura B. 7 – Widgets

Configuração do SMAER

Após a instalação do *Raspberry Pi OS*, é necessário instalar o *Python* e *NodeJS,* através dos seguintes comandos:

```
sudo apt-get install python3
```

```
sudo apt-get install python3-pip
```

```
pip install blynk-library-python
```

```
sudo apt-get update && sudo apt-get upgrade
```

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

Posteriormente, instalar a bliblioteca do *Blynk* para *Python*:

```
sudo npm install blynk-library –g
```

```
sudo npm install onoff -g
```

Configuração do Servidor

Após a configuração do *Raspberry Pi OS*, é necessário instalar o Java 8 no servidor através dos seguintes comandos *Linux*:

```
sudo apt install openjdk-8-jdk openjdk-8-jre
```

Verificar a versão do Java:

```
java –version
```

Instalar o servidor *Blynk*:

```
wget "https://github.com/blynkkk/blynk-
server/releases/download/v0.41.13/server-0.41.13-java8.jar"
```

Arrancar com o servidor *Blynk*:

```
java -jar server-0.41.13-java8.jar -dataFolder /home/pi/Blynk
```

Adicionar o serviço ao arranque do servidor para que sempre o RPi inicie automaticamente o servidor *Blynk*. Escrever no ficheiro "/etc/rc.local" a seguinte linha:

```
java -jar /home/pi/server-0.41.13-java8.jar -dataFolder /home/pi/Blynk &
```

Anexo C. Código Utilizado

Foram desenvolvidos dois *scripts* de raiz, um escrito em linguagem *Python* e outro em *Node.js* para a implementação das funcionalidades do SMAER. Ambos os *scripts* encontram-se disponíveis e acessíveis através dos seguintes *links*:

- Python *Script* - https://bit.ly/3kJUYWE
- NodeJS *Script* - https://bit.ly/2HTA74r

Anexo D. Artigo Científico

Smart System for Monitoring Pets Remotely

Filipe Afonso
ISCTE - IUL
Lisbon, Portugal
fmnao11@iscte-iul.pt

Nuno Souto
ISCTE - IUL
Instituto de Telecomunicações
Lisbon, Portugal
nuno.souto@iscte-iul.pt

Pedro Sebastião
ISCTE - IUL
Instituto de Telecomunicações
Lisbon, Portugal
pedro.sebastiao@iscte-iul.pt

***Abstract*— This paper proposes a new flexible and low cost system that allows monitoring pets remotely in real time and works like a pets food dispenser, using an approach oriented to IoT and 3D Printing. The main advantage of this system, which distinguishes it from other solutions, is its scalability in terms of the number of sensors used and the devices controlled by the Mobile Application. In addition, all the components used are efficient and low cost. The system uses parts designed and printed in 3D, making it totally different and unique from other available solutions.**

Keywords—Internet of Things, 3D Printing, Sensors, ESP32, Arduino, Raspberry Pi.

I. Introduction

Nowadays, there is an increasing need to connect any device to the Internet, not only to send data to remote servers in order to process and store the information, but also to enable remote control of the devices from anywhere in the world, through the Internet. Every day and every time new equipment is being connected to the WEB and it is estimated that by 2025, the number of IoT (Internet of Things) devices will reach the value of 77.44 billion [1].

Intelligent or smart houses, where it is possible to use a smartphone to remotely control the lights of a house, the air conditioning or even other less intelligent equipment connected to a standard electrical outlet, are examples of this new paradigm that we call the Internet of Things (IoT).

IoT was designed with the aim of gaining time, improving our quality of life and is present in many experiments of our quotidian, such as transport, health and industrial automation, among others.

On the other hand, Artificial Intelligence (AI), corresponds to a branch of computer science dedicated to the study and research on ways to replicate human intelligence in machines, making them capable of performing tasks that normally require human skills. The most commonly used algorithms in AI systems include learning, logical reasoning, planning, solving problems and making decisions.

Virtual assistants and Chatbots are some examples of services that already use AI nowadays. This kind of technology uses the data of millions of people and generates every day better answers to our needs.

If we combine these two aspects, IoT for obtaining the data and trends, and AI to analyze and treat the data without the need of the human being, we obtain a "machine" that molds itself according to the information it receives over time, following an evolutionary process of learning.

This is why the more devices connected to the Internet network we use, the more intelligent they become.

II. Related Work and Motivation

The fact that we live in a society where more and more people own their pets, leads us to have some concerns when we go on vacations or longer weekends. There is always the concern to know how our animals are, what they do at that moment, and especially that these family members do not lack food.

Although there are already some automatic feed dispensers on the market, they are quite limited in terms of functionality's. As a rule these existing devices are only mechanical and without any kind of intelligence or built-in sensors. They end up working in an isolated way because they do not have any connectivity with the networks.

If we add to these less intelligent devices the concept of IoT, we have control over these devices anywhere in the world [2]. Besides, it will be possible to implement routines, send alerts and transfer data in real time. This allows us to enhance new features and services [3] [4] like food and water consumption.

Motivated by this, in this paper we propose something new and distinctive, totally open-source with low cost materials and accessible to all, because usually these solutions are quite expensive [5].

III. Proposed System Architecture

The purpose of this paper is to develop a flexible and low cost system that allow monitoring pets remotely. The specific implementation described in this paper consists of a feed dispenser with an electric motor in order to release feed automatically. This dispenser incorporates:

- A temperature and humidity sensor to ensure that the feed remains in ideal conditions for the animals.
- An ultrasonic sensor that allows to estimate the feed level inside the container.
- A camera that allows the remote observation of the animals, at any time through an application developed for Android and IOS.

In terms of power, to make the system resilient to power failures, a battery (Powerbank) is adopted in order to ensure a permanent operation and facilitate portability. The system architecture is shows on figure 1 below.

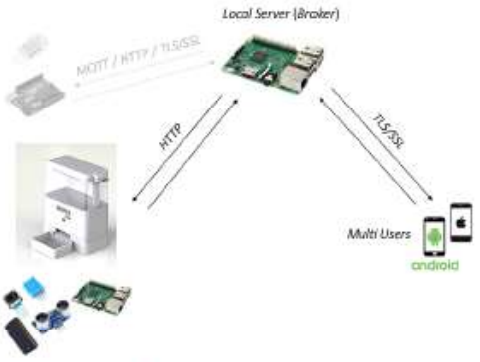


Fig. 1. System Architecture

A. Hardware

Regarding the choice of the hardware used in the implemented prototype, some selection criteria of the material was followed, namely its cost, always opting for cheaper components but not compromising the system capacity. The components used were the following:

- Broker – a Raspberry Pi version 3 with Raspberry Pi OS;
- DHT11 - a temperature and humidity sensor that allows temperature readings between 0°C and +50°C and humidity readings between 20% to 95% [2] (environment temperature and humidity);
- HC-SR04 – a Ultrasonic sensor with 40kHz of frequency [3] to provide distance measurements allowing the estimation of the amount of food inside the container;
- 28BYJ-48 with driver ULN2003 - an electric motor with four phases which can operate between 5 – 12 volts with a step angle of 5,625° with a 64:1 gear reduction;
- Raspberry Pi Cam Module version 1 - a camera with 5 Megapixels, supporting video with resolution 1080p30,720p60,640×480p60/9 and OmniVision OV564 sensor [8];
- iWalk Powerbank [9] – a battery with 8000mAh and 2.4A output.

B. The Prototype Design and 3D Printing

The whole feeding prototype was designed in Solidworks [9] and consists of a ten pieces model which after assembled, looks as shown in figure 6. It consists of a container with a lid, where the feed is placed which is later delivered through a cone in a rotation system with dividers. In turn, the feeder rotates with the help of the electric motor that releases the feed to an outside.

All electronic components used (Raspberry Pi, Motor, Powerbank and Cables) are in the lower compartment. Figure 2 shows the dimensions and the location of the electronic parts.

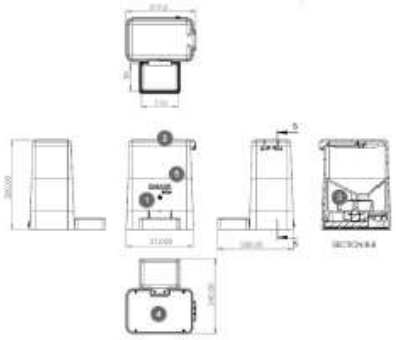


Fig. 2. Dimensions of the Prototype of Food Feeding System

For the 3D printing of the different prototype parts, two materials were used: PLA (Polylactic Acid) and PETG (Polyethylene Terephthalate Glycol). The base (compartment number 4 of figure 6) and the rotation system were printed with PETG because the of the higher mechanical stress that these two pieces are subjected to (PETG has a higher strength compared with PLA). For printing all the 3D parts used in the described implementation, 208 hours were necessary with Ender 3 PRO printer.

The figure 3 shows all the parts assembled.

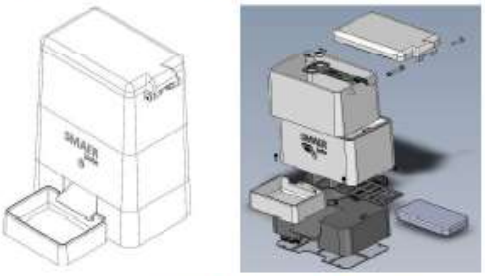


Fig. 3. The Prototype of Food Feeding System

C. Communication

The proposed architecture (figure 1) considers the use of a Broker, responsible for the communication with the multiple users of the Mobile Application and also with the various IoT devices that can be installed on the system. The communication between the IoT devices and the Broker, as they are often in the same local network, is supported through Hypertext Transfer Protocol (HTTP). However, the local server also supports MQTT (Message Queuing Telemetry Transport) and TLS/SSL (Transport Layer Security / Secure Sockets Layer) protocols to communicate with the hardware.

The IoT devices or Mobile Application try to connect to the Broker through the IP address, Port and the Token.

On figure 4 we can see the diagram of the communication, all the IoT Devices and the Users talk with the Broker. First needs to connecting using the IP address, Port and Token. After that can use the events to read or write the values with the sensors on devices.

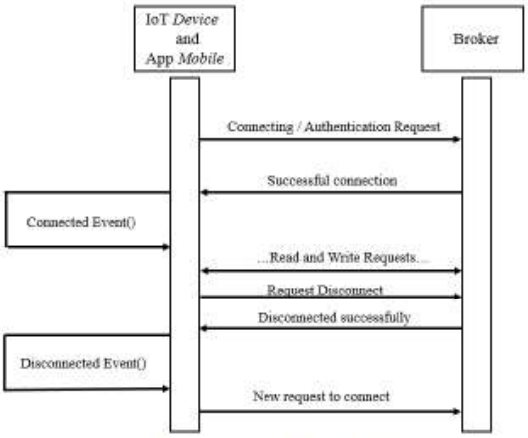


Fig. 4. Communications with the Broker

The schematic diagram of the connections is shown in Figure 5.

Presentation video: https://youtu.be/73TMFNqaFa8

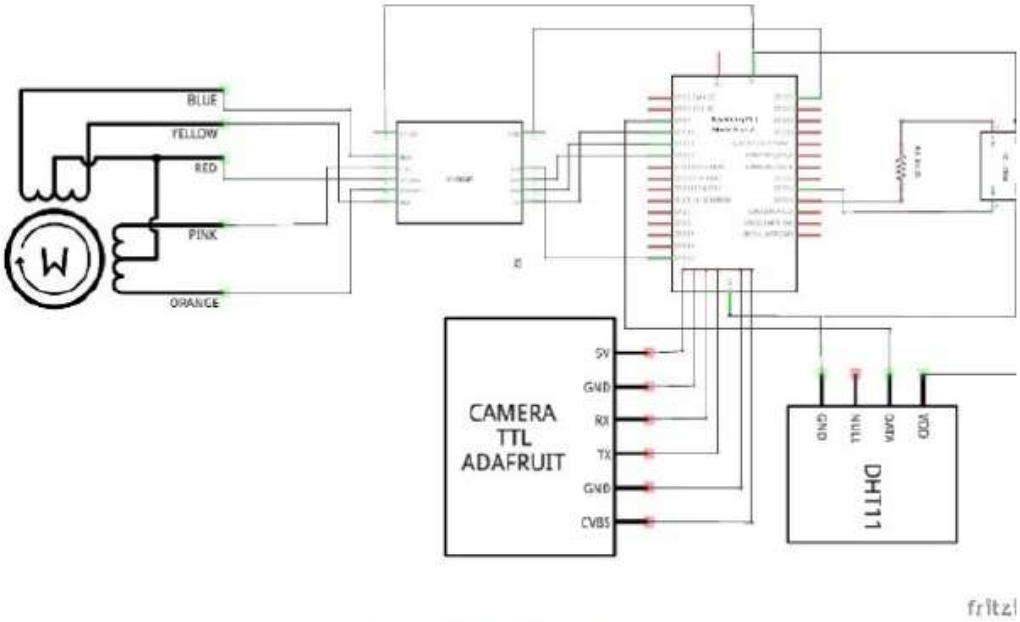


Fig. 5. The Schematic Diagram

Before assembly all the parts (hardware and software), tests were carried out to ensure that everything was working properly like shown in figure 6.

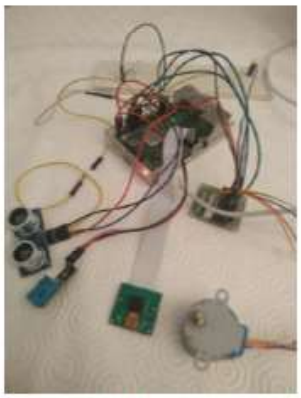


Fig. 6. Testing all the components

D. Software

The implemented Broker is based on the open source Blynk Server [9]. This platform offers various functionalities, namely, it enables the remote control or monitoring of projects over the Internet. Adding the unique token into the code of the device, we can identify the device and communicate with them.

The Blynk Application is very flexible. We can add widgets, buttons and displays to control the sensors on IoT Devices. For this project was created application demonstrated in the figure 7. On this app we can see the temperature and humidity, the percentage of food available on feeding system, the camera images and the button to dispenser food.

Fig. 7. Application Mobile

IV. Results and Discussions

To prove the practical functionality of the developed system, a test scenario was implemented using the software and hardware mentioned in this project.

The prototype is functional and can be viewed in this presentation video: https://youtu.be/73TMFNqaFa8.

A. Energy Consumption

One of the critical factors to be taken into account when implementing an IoT system is the energy consumption of the devices used, which typically should be low as they have often small dimensions which restrict the use of large batteries. In addition, since they have to operate remotely, reduced energy consumption helps to prolong the equipment active for longer.

Consumption was measured with all sensors and camera connected, Wi-Fi network active and in communication with the server and also with the electric motor running. In this

Presentation video: https://youtu.be/73TMFNqaFa8

way, we were able to obtain the maximum peak power consumption as shown in figure 8.

Fig. 8. The Consumption of the System

The battery used in this project has a maximum output of 2.4A. The voltage was also measured under the same conditions, resulting a value of 4.68V×0.840A = 3.93 Watts. The calculation of the estimated autonomy of the prototype can then be obtained using the following formula:

$$\text{Estimate Time} = \frac{\text{Battery Capacity}}{\text{Consumption}} \quad (1)$$

.

Assuming that Powerbank has all the initial capacity since it came from the factory, and the fact that the consumption measured with the multimeter was performed at peak use, i.e. this value will not be constant, it can be concluded that in case of AC failure the system will continue to operate for at least 9 hours. In a normal situation, this will be enough time for the power supply to recover from a fault.

V. Conclusions and Future Work

In this paper we described the implementation of flexible and low cost system that allows monitoring pets remotely in real time. The proposed solution works like a pets food dispenser and, uses an approach oriented to IoT and 3D Printing.

The main advantage of the system, which distinguishes it from other solutions, is it scalability in terms of the number of sensors used and the devices controlled by the Mobile Application. All the components used are efficient and low cost. The system uses parts designed and printed in 3D, making.

The adopted architecture allows simple integration of additional sensors and functionalities, as well as the support of other communication protocols, such as Bluetooth LE, LoRaWAN and NB-IoT.

Acknowledgment

This work was partially supported by the Institute of Lisbon and by the FCT/MCTES through national funds and when applicable co-funded by EU funds under the project UIDB//50008/2020.

Presentation video: https://youtu.be/73TMFNqaFa8

References

[1] "Statista Research Department, "Internet of Things - number of connected devices worldwide 2015-2025"," [Online]. Available: https://www.statista.com/statistics/471264/iot-number-of-connected-devices-worldwide/. [Accessed 11 Setember 2020].

[2] Y. Chen and M. Elshakankiri, "Implementation of an IoT based Pet Care System," *Fifth International Conference on Fog and Mobile Edge Computing (FMEC), Paris, France,* pp. 256-262, 2020.

[3] A. Luayon, G. Tolentino, V. Almazan, P. Pascual and M. Samonte, "PetCare: a smart pet care IoT mobile application" , Proceedings of the 10th International Conference on E-Education, E-Business, E-Management and E-Learning," p. 427–431, January 2019.

[4] A. A. Hammam, M. M. Soliman and A. E. Hassanein, "DeepPet: A Pet Animal Tracking System in Internet of Things using Deep Neural Networks," pp. 38-43, 2018.

[5] M. Raza, N. Aslam, H. Le-Minh, S. Hussain, Y. Cao and N. M. Khan, "A Critical Analysis of Research Potential, Challenges, and Future Directives in Industrial Wireless Sensor Networks," vol. 20, pp. 39-45, 2018.

[6] Datasheet, "DHT11_Aosong," [Online]. Available: https://www.electronicoscaldas.com/datasheet/DHT11_Aosong.pdf. [Accessed 01 October 2020].

[7] "Ultrasonic Ranging Module HC - SR04 DataSheet," [Online]. Available: https://cdn.sparkfun.com/datasheets/Sensors/Proximity/HCSR04.pdf. [Accessed 18 September 2020].

[8] "RPi Camera Modules," [Online]. Available: https://www.raspberrypi.org/documentation/hardware/camera/. [Accessed 07 Outuber 2020].

[9] "Review iWalk 8000mah slim powerbank," [Online]. Available: https://www.chargerharbor.com/review-iwalk-t08-8000mah-slim-power-bank/. [Accessed 27 12 2020].

[10] "Solidworks," [Online]. Available: https://www.solidworks.com/. [Accessed 22 October 2020].

[11] B. Server. [Online]. Available: https://github.com/blynkkk/blynk-server#blynk-server. [Accessed 01 December 2020].

[12] "Alexa," [Online]. Available: https://www.amazon.com/smart-home-devices/b?ie=UTF8&node=9818047011. [Accessed 01 December 2020].

[13] "Google Home," [Online]. Available: https://play.google.com/store/apps/details?id=com.google.android.apps.chromecast.app&hl=pt. [Accessed 01 December 2020].

Referências Bibliográficas

[1] "Statista Research Department, "Internet of Things - number of connected devices worldwide 2015-2025"," [Online]. Available: https://www.statista.com/statistics/471264/iot-number-of-connected-devices-worldwide/.

[2] Y. Chen e M. Elshakankiri, "Implementation of an IoT based Pet Care System," *Fifth International Conference on Fog and Mobile Edge Computing (FMEC), Paris, France,,* pp. 256-262, 2020.

[3] A. Luayon, G. Tolentino, V. Almazan, P. Pascual e M. Samonte, "PetCare: a smart pet care IoT mobile application" , Proceedings of the 10th International Conference on E-Education, E-Business, E-Management and E-Learning," p. 427–431, Janeiro 2019.

[4] A. A. Hammam, M. M. Soliman e A. E. Hassanein, "DeepPet: A Pet Animal Tracking System in Internet of Things using Deep Neural Networks," *2018 13th International Conference on Computer Engineering and Systems (ICCES), Cairo, Egypt,* pp. 38-43, 2018.

[5] M. Raza, N. Aslam, H. Le-Minh, S. Hussain, Y. Cao e N. M. Khan, "A Critical Analysis of Research Potential, Challenges, and Future Directives in Industrial Wireless Sensor Networks," *in IEEE Communications Surveys & Tutorials,* vol. 20, pp. 39-95, 2018.

[6] F. Karray, M. W. Jmal, A. Garcia-Ortiz, M. Abid e A. M. Obeid, "A comprehensive survey on wireless sensor node hardware platforms," *Computer Networks,* vol. 144, pp. 89-110, 2018.

[7] C. Bell, Beginning Sensor Networks with Arduino and Raspberry Pi, Ed. Apress, 2013.

[8] E. Ferro e F. Potortì, "Bluetooth and Wi-Fi wireless protocols: A survey and a comparison," em *IEEE Wireless Communications*, Vol. 12 ed., 2005, p. 12–26.

[9] A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari e M. Ayyash, "Internet of Things: A Survey on Enabling Technologies, Protocols, and Applications," em *IEEE Communications Surveys and Tutorials*, 2015.

[10] J. Marais e R. Malekian, "LoRa and LoRaWAN Testbeds: a Review," em *IEEE Africon 2017 Proceedings*, Vol. 24 ed., 2017, p. 1496–1501.

[11] "Narrowband – Internet of Things (NB-IoT)," [Online]. Available: https://www.gsma.com/iot/narrow-band-internet-of-things-nb-iot/. [Acedido em 21 Setembro 2020].

[12] "What are the Differences Between LTE CAT M1 and NB-IoT Connectivity?," [Online]. Available: https://www.soracom.io/blog/what-are-the-differences-between-lte-cat-m1-and-nb-iot-connectivity/. [Acedido em 21 Setembro 2020].

[13] W. Bharati, W. Kirti, N. Ramith e N. Amala, "IEEE Conference Publication," em *Comparison with HTTP and MQTT In Internet of Things (IoT)*, International Conference on Inventive, 2018, p. 249–253.

[14] "How MQTT Works -Beginners Guide," [Online]. Available: https://www.gsma.com/iot/narrow-band-internet-of-things-nb-iot/. [Acedido em 22 Setembro 2020].

[15] “TLS version 1.3 RFC,” [Online]. Available: https://tools.ietf.org/html/rfc8446. [Acedido em 10 Outubro 2020].

[16] “Internet Engineering Task Force,” [Online]. Available: Disponível em: https://www.ietf.org/. [Acedido em 10 Outubro 2020].

[17] “TLS HandShake,” [Online]. Available: https://tools.ietf.org/html/rfc8446#page-24. [Acedido em 11 Outubro 2020].

[18] “IEEE 7th Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON),” em *Intelligent Appliances Controller Using Raspberry Pi Through*, 2016.

[19] “Arduino,” [Online]. Available: https://www.arduino.cc/. [Acedido em 17 Setembro 2020].

[20] “The Raspberry Pi Foundation – Raspberry,” [Online]. Available: https://www.raspberrypi.org/. [Acedido em 10 Setembro 2020].

[21] “Python,” [Online]. Available: https://www.python.org/. [Acedido em 19 Setembro 2020].

[22] “Raspberry Pi 4,” [Online]. Available: https://www.raspberrypi.org/products/raspberry-pi-4-model-b/specifications/. [Acedido em 12 Setembro 2020].

[23] “Raspberry Zero W,” [Online]. Available: https://www.raspberrypi.org/products/raspberry-pi-zero-w/.

[24] “Node MCU,” [Online]. Available: https://www.nodemcu.com/index_en.html. [Acedido em 15 Outubro 2020].

[25] “ESP-32 DataSheet,” [Online]. Available: https://www.espressif.com/sites/default/files/documentation/.

[26] “LoPy4 Board,” [Online]. Available: https://pycom.io/product/lopy4/.

[27] “GPy Board,” [Online]. Available: https://pycom.io/product/gpy/. [Acedido em 13 Setembro 2020].

[28] “PLA,” [Online]. Available: https://www.impresoras3d.com/pt/filamento-pla-consejos-caracteristicas-y-mucho-mas/. [Acedido em 10 Setembro 2020].

[29] “ABS,” [Online]. Available: https://www.impresoras3d.com/pt/un-repaso-a-los-trucos-mas-usados-para-imprimir-en-abs/. [Acedido em 02 Setembro 2020].

[30] “PETG,” [Online]. Available: https://3dlab.com.br/como-imprimir-com-o-filamento-petg/. [Acedido em 12 Setembro 2020].

[31] “Blender,” [Online]. Available: https://www.blender.org/. [Acedido em 29 Setembro 2020].

[32] “Blender,” [Online]. Available: https://www.blender.org/. [Acedido em 21 Setembro 2020].

[33] [Online]. Available: https://www.solidworks.com/. [Acedido em 18 Setembro 2020].

[34] “Ultimaker Cura,” [Online]. Available: https://ultimaker.com/software/ultimaker-cura. [Acedido em 02 Setembro 2020].

[35] “Creality,” [Online]. Available: https://www.creality3dshop.eu/.

[36] “Prusa,” [Online]. Available: https://www.prusa3d.com/. [Acedido em 11 Setembro 2020].

[37] “BigTreeTech,” [Online]. Available: http://www.bigtree-tech.com/. [Acedido em 08 Setembro 2020].

[38] “https://filament2print.com/gb/blog/94_bowden-direct-extrusion.html,” [Online]. Available: https://filament2print.com/gb/blog/94_bowden-direct-extrusion.html. [Acedido em 02 Outubro 2020].

[39] “DHT11_Aosong,” [Online]. Available: https://www.electronicoscaldas.com/datasheet/DHT11_Aosong.pdf. [Acedido em 01 Outubro 2020].

[40] “DHT22 Sensor,” [Online]. Available: https://www.sparkfun.com/datasheets/Sensors/Temperature/DHT22.pdf. [Acedido em 06 Outubro 2020].

[41] “Ultrasonic Ranging Module HC - SR04 DataSheet,” [Online]. Available: https://cdn.sparkfun.com/datasheets/Sensors/Proximity/HCSR04.pdf. [Acedido em 18 Setembro 2020].

[42] “RPi Camera Modules,” [Online]. Available: https://www.raspberrypi.org/documentation/hardware/camera/. [Acedido em 07 Outubro 2020].

[43] “Fritzing software,” [Online]. Available: https://fritzing.org/. [Acedido em 08 Outubro 2020].

[44] “Blynk,” [Online]. Available: https://docs.blynk.cc/. [Acedido em 18 10 2020].

[45] “Blynk Server,” [Online]. Available: https://github.com/blynkkk/blynk-server#blynk-server. [Acedido em 19 10 2020].

[46] “Netty,” [Online]. Available: https://github.com/netty/netty. [Acedido em 19 10 2020].

[47] “https://nodejs.org/en/,” [Online]. Available: https://nodejs.org/en/. [Acedido em 10 09 2020].

[48] “Wireshark,” [Online]. Available: https://www.wireshark.org/. [Acedido em 02 10 2020].

[49] “Mauser,” [Online]. Available: www.mauser.pt. [Acedido em 03 10 2020].

[50] “Google,” [Online]. Available: https://play.google.com/store/apps/details?id=com.google.android.apps.chromecast.app&hl=pt. [Acedido em 25 10 2020].

[51] “Amazon,” [Online]. Available: https://www.amazon.com/smart-home-devices/b?ie=UTF8&node=9818047011. [Acedido em 25 Outubro 2020].