



Universidade do Minho
Escola de Engenharia

MESTRADO INTEGRADO EM ENGENHARIA DE TELECOMUNICAÇÕES E
INFORMÁTICA

MICRODISPOSITIVOS DE RF PARA COMUNICAÇÃO SEM-FIOS

SISTEMA DE MONITORIZAÇÃO DE ESTUFAS DOMÉSTICAS
RELATÓRIO INTERMÉDIO

Grupo 9:

Augusto César da Silva Mota - A76563

Hugo Daniel da Costa Cunha Machado - A80662

João Fernando Alonso Barbosa - A68453

José Pedro Afonso Rocha - A70020

Miguel Ângelo Teixeira Pereira - A82875

Tiago Martins Teles - A76266

Guimarães, 12 de fevereiro de 2021

Índice

| | | |
|----------|--|-----------|
| 1 | Introdução | 5 |
| 2 | Fundamentos Teóricos | 6 |
| 2.1 | Ideia geral | 6 |
| 3 | Desenvolvimento | 7 |
| 3.1 | Arquitetura do Sistema | 7 |
| 3.2 | Arquitetura do Sistema Sensor Estufa | 9 |
| 3.2.1 | Hardware | 10 |
| 3.2.2 | Software | 15 |
| 3.3 | Cloud IoT | 20 |
| 3.3.1 | BackEnd | 20 |
| 3.4 | Aplicação <i>web</i> | 26 |
| 3.4.1 | <i>User Interface</i> (UI) | 26 |
| 3.4.2 | Frontend | 27 |
| 4 | Testes, Resultados e Discussão | 32 |
| 4.1 | Sessão inicial de configuração da rede doméstica | 32 |
| 4.2 | Comunicação entre estufa e a plataforma | 33 |
| 4.3 | Conteúdo da base de dados | 36 |
| 5 | Conclusão | 38 |
| 6 | Referências | 40 |

Índice de Imagens

| | | |
|----|---|----|
| 1 | Arquitetura do sistema. | 8 |
| 2 | Arquitetura do sistema sensor. | 9 |
| 3 | NodeMCU-32 [3]. | 10 |
| 4 | Estrutura interna sensor humidade [4]. | 11 |
| 5 | Coefficiente de Temperatura Negativo [6]. | 11 |
| 6 | Pinout do Moisture Sensor. | 12 |
| 7 | Pinout do photo LDR resistor. | 13 |
| 8 | Mini Water Pump. | 13 |
| 9 | Esquema do step motor | 14 |
| 10 | Fluxograma da verificação dos limites do perfil | 18 |
| 11 | Base de dados. | 23 |
| 12 | Exemplo de um <i>endpoint</i> de uma rota. | 25 |
| 13 | Exemplo de uma função de resposta. | 25 |
| 14 | Página de boas-vindas. | 27 |
| 15 | Barra de opções da página de boas-vindas. | 27 |
| 16 | Página de registo. | 28 |
| 17 | Página de <i>login</i> | 28 |
| 18 | Página da <i>Dashboard</i> | 29 |
| 19 | Página da <i>Dashboard</i> | 29 |
| 20 | Página de registo de nova estufa. | 29 |
| 21 | Página de controlo de dispositivos e amostras. | 30 |
| 22 | Página de perfis. | 30 |
| 23 | Página de registo de perfis. | 31 |
| 24 | Página de configuração da rede. | 32 |
| 25 | Tópicos (MQTT) do dispositivo. | 33 |
| 26 | POST do registo do dispositivo. | 33 |
| 27 | POST das amostras. | 33 |
| 28 | POST das amostras. | 33 |
| 29 | Definições do perfil. | 34 |
| 30 | Receção de perfil e atuação. | 34 |
| 31 | Ações da estufa. | 35 |

| | | |
|----|---|----|
| 32 | Abertura e fecho da estufa. | 35 |
| 33 | Tabela <i>User</i> | 36 |
| 34 | Tabela <i>Device</i> | 36 |
| 35 | Tabela <i>Perfil</i> | 36 |
| 36 | Tabela <i>rel_user_device</i> | 36 |
| 37 | Tabela <i>History</i> | 37 |

Índice de Tabelas

1 Recursos da API. 22

1. Introdução

Na unidade curricular de Microdispositivos de RF para Comunicações Sem Fios, foi-nos proposto realizar um projeto que consiste na criação de um sistema de comunicação sem fios para a monitorização de parâmetros de um sistema com base na utilização de sensores e dispositivos RF.

Nos dias que vivemos, tem-se dado cada vez mais valor e importância aos produtos biológicos. Como tal, existe uma maior procura e também uma subida dos preços desses produtos. São indicados como produtos que trazem vantagens para a saúde do consumidor por terem uma grande quantidade de antioxidantes e, como são produzidos sem recurso a produtos químicos, são menos prejudiciais para o planeta.

O nosso grupo decidiu, em unanimidade, criar uma pequena estufa de uso doméstico que, tal como nos foi pedido, fosse monitorizada e controlada com um sistema de comunicação sem fios.

Como a agricultura é uma área que ainda tem pouco investimento tecnológico ao nível da monitorização e registo de variáveis, este projeto pode, mais tarde, ser adaptado para ambientes maiores e com características diferentes daquelas que vão ser implementadas.

Neste relatório será descrito todo o sistema implementado, bem como as razões para a escolha dos componentes e tecnologias que o constituem.

2. Fundamentos Teóricos

2.1. Ideia geral

Para este projeto decidimos fazer uma estufa automatizada para uso doméstico. E para tal, foi necessário estudar de forma muito breve a arte do cultivo [1]. De forma a conseguir monitorizar e controlar os aspetos mais importantes durante o cultivo.

Os principais aspectos que decidi-mos monitorizar são a temperatura e humidade do ar, humidade do solo e luminosidade. E os aspetos a controlar são a abertura da estufa e a rega.

De forma a automatizar todo este processo de cultivo, será possível definir os limites para cada tipo de cultivo. Para que a estufa se mantenha dentro deles sozinha, sem interação humana. Por exemplo se a temperatura máxima for 50 °C, caso a temperatura interior ultrapasse esse valor, a estufa será aberta automaticamente de forma a arrefecer a temperatura interior. O mesmo acontece com a rega, caso a humidade do solo esteja abaixo do valor mínimo, a estufa rega automaticamente.

Todos estes aspectos serão controláveis através de uma plataforma *web*, o que tornará a interação entre o consumidor final e a estufa simples e mínima. Esta plataforma terá um painel de controlo, onde será possível impor os limites para cada tipo de cultivo e monitorizar os aspetos de cultivo.

3. Desenvolvimento

3.1. Arquitetura do Sistema

A arquitetura do sistema pode ser separada em três áreas: dispositivo, *middleware* e o cliente.

O dispositivo é composto por quatro sensores e 2 atuadores conectados a um NodeMCU e efectua a sua comunicação por WiFi. Pressupõe-se que o utilizador do dispositivo terá uma ligação à internet e o dispositivo tem uma opção de configuração para facilitar a ligação WiFi.

No *middleware* estão os componentes que fazem a ponte entre os dispositivos e os clientes. É utilizado um *broker* MQTT, para executar ações imediatas nos dispositivos e para informar os clientes regularmente acerca do estado atual, e uma API para receber todas as informações passíveis de serem guardadas na base de dados relacional e posterior consulta pelos clientes.

Por último, os clientes são a interface com os utilizadores. Estes permitem não só consultar o estado atual e executar ações imediatas nos dispositivos como também ver o histórico de todos os sensores ao longo do tempo.

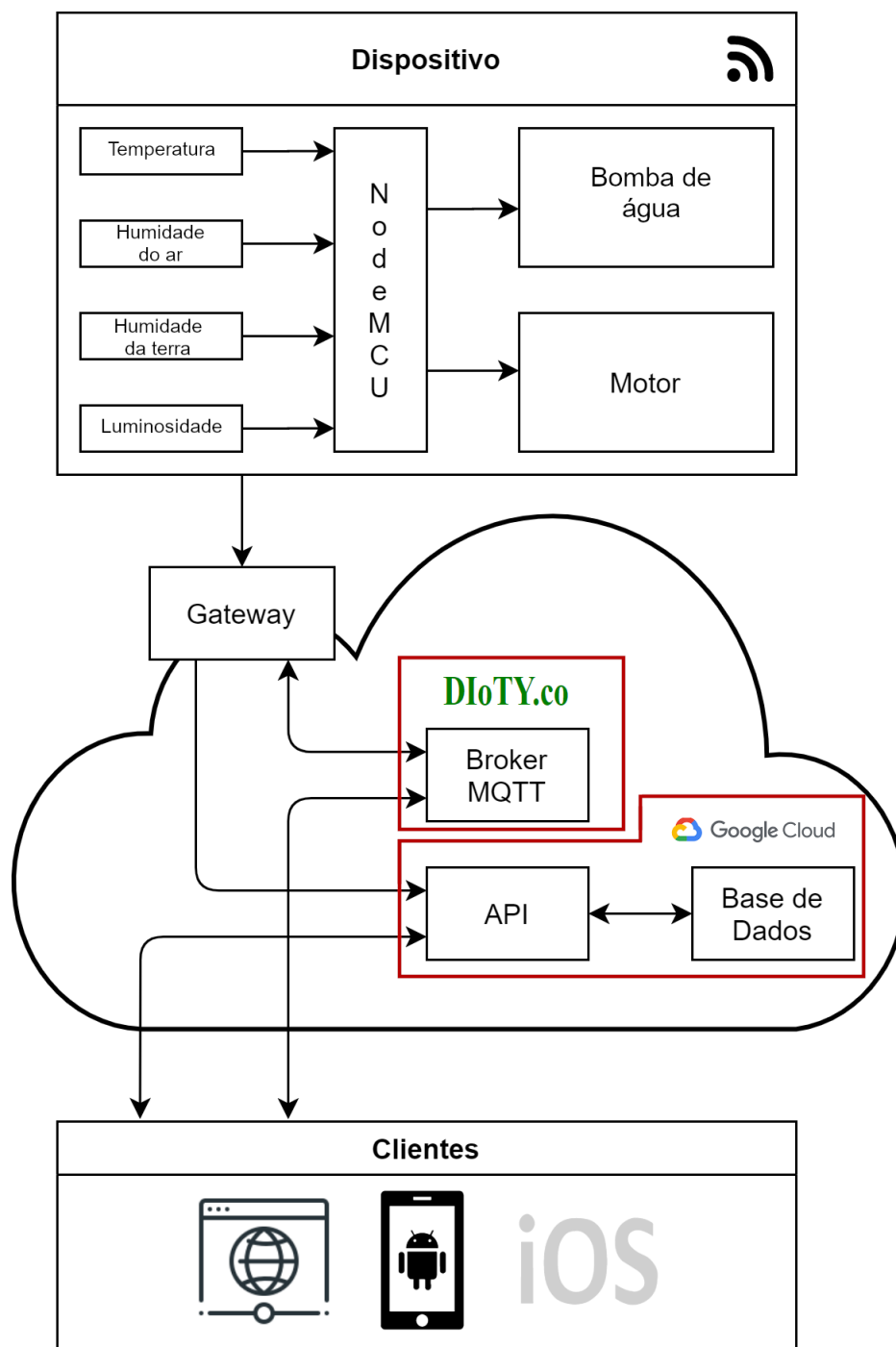


Figura 1: Arquitetura do sistema.

3.2. Arquitetura do Sistema Sensor Estufa

O sistema sensor estufa (Figura 2) vai ser constituído por um conjunto de módulos dos sensores de luminosidade, temperatura e humidade do ar e humidade da terra, um conjunto de módulos atuadores para rega e abertura de estufa e por fim um respetivo micro-controlador para a gestão destes módulos.

Estes componentes serão responsáveis pela recolha de dados, controlo do cultivo em questão e serão posteriormente enviados para a *cloud* IoT definida na arquitetura, onde serão processados e disponibilizados para os utilizadores.

Este sistema sensor receberá da *cloud* um conjunto de parâmetros a cumprir para funcionar de forma autónoma.

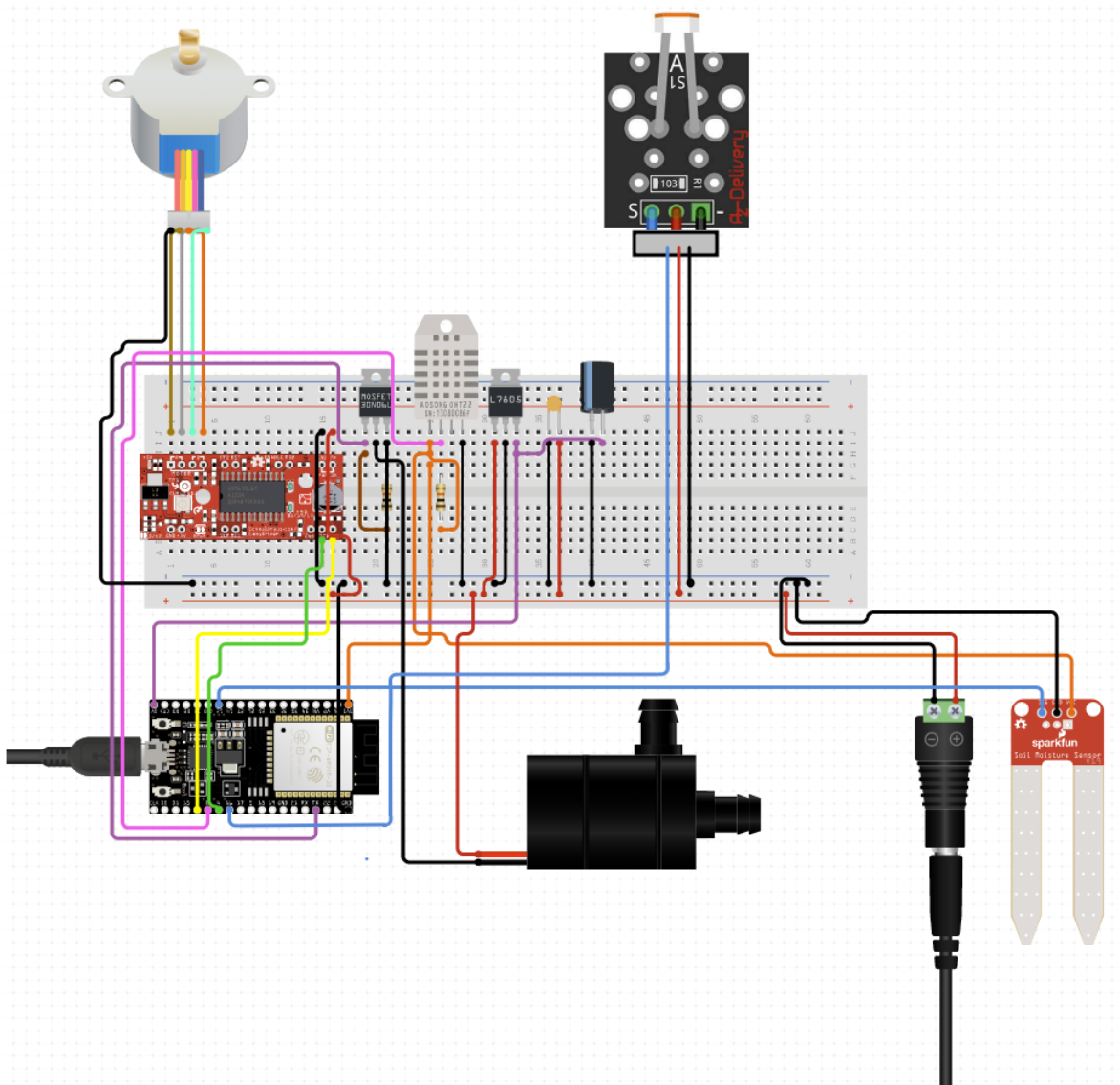


Figura 2: Arquitetura do sistema sensor.

3.2.1. Hardware

3.2.1.1 Micro-Controlador

O micro-controlador utilizado neste projeto foi o NodeMCU-32 [3]. Esta é uma placa de desenvolvimento baseada no módulo ESP32 Wi-Fi, um equipamento electrónico desenvolvido com o intuito de conectar projectos à Internet com grande facilidade e baixo custo.

Este é muito eficiente, constituído por um módulo ESP-WROOM-32 que contém o ESP32, um cristal de 40 MHz, antena embutida e uma porta micro USB. O NodeMCU-32 também contém *Bluetooth* v4.2 embutido e um micro-controlador dual-core 32-bit, o que torna este dispositivo mais prático e aumenta as possibilidades de uso.

Esta placa de desenvolvimento é extremamente adequada para aplicações de IoT como é pretendido no nosso projecto.

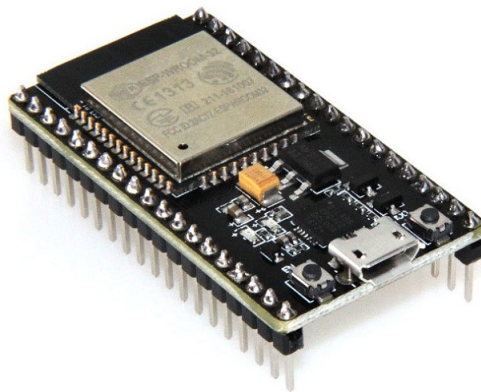


Figura 3: NodeMCU-32 [3].

3.2.1.2 Sensor de Temperatura e Humidade do ar (DHT 22) [5]

Este módulo com baixo consumo de corrente é constituído por um sensor de temperatura e humidade, e também um conversor analógico/digital para realizar comunicação com o micro-controlador. Uma das grandes vantagens da utilização deste módulo é oferecer uma leitura de valores com precisão de valores decimais.

O componente de sensor humidade, é constituído por dois eléctrodos com substrato de retenção de humidade. No substrato, os íons são liberados à medida que o vapor de água é absorvido por ele, aumentando a condutividade entre os eletrodos. As alterações na resistência entre os dois eletrodos vai ser proporcional à humidade relativa, sendo que a humidade relativa mais alta diminui a resistência entre os eletrodos, e a humidade relativa mais baixa aumenta a resistência entre os eletrodos.

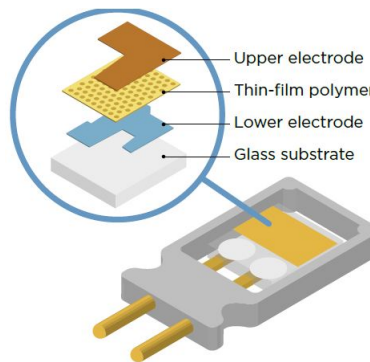


Figura 4: Estrutura interna sensor humidade [4].

A componente de sensor temperatura é constituído por um termistor, sendo um resistor térmico que muda a sua resistência com a temperatura. Este termistor é feito de forma a que a sua resistência mude drasticamente com a temperatura.

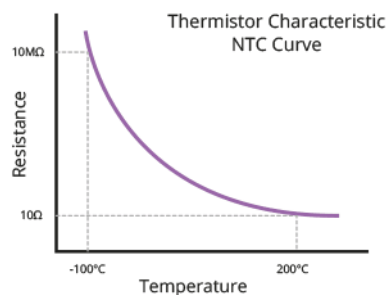


Figura 5: Coeficiente de Temperatura Negativo [6].

3.2.1.3 Sensor de Humidade do solo (YL-69) [8]

Este sensor é constituído por uma sonda em forma de garfo com dois condutores que se vão encontrar expostos no solo em qualquer tipo de lugar. Funciona com a sonda em forma de garfo com dois condutores expostos atuando como um resistor variável onde a resistência varia de acordo com o teor de água no solo, sendo esta resistência inversamente proporcional à humidade do solo:

- Quanta mais água se encontrar no solo, melhor condutividade, resultando um valor menor na resistência
- Quanta menos água se encontrar no solo, pior condutividade, resultando um valor maior na resistência.

Este sensor gera uma tensão de saída de acordo com o valor da resistência, e se for medido é possível obter o nível de humidade.

O sensor, é composto por um módulo eletrónico que vai realizar uma conexão da sonda com o arduíno, módulo que vai produzir uma tensão de saída de acordo com o valor da resistência da ponta prova e de seguida é disponibilizado num pino de saída analógica. Este sinal analógico de seguida é enviado para um comparador de alta precisão para digitalizar e disponibilizar num pino de saída digital.

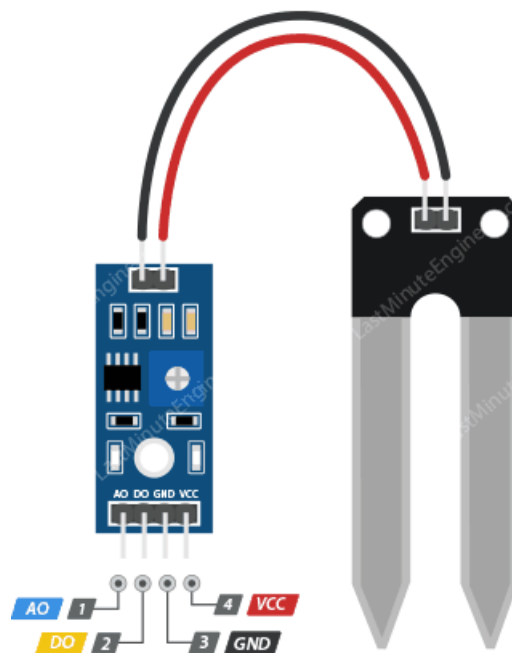


Figura 6: Pinout do Moisture Sensor.

3.2.1.4 Sensor de Luz (KY-018 Photo LDR Resistor) [9]

Este tipo de sensor, é um dispositivo sensível á luz que é utilizado para indicar a presença ou ausência de luz para medir a sua intensidade. Quando se encontra na presença de um ambiente escuro, o valor da sua resistência é muito alto, mas quando se encontra numa zona iluminada, o valor da resistência diminui drasticamente. Este sensor é constituído por um resistor depende de luz que tem uma sensibilidade que varia com o comprimento de onda da luz que é aplicada.

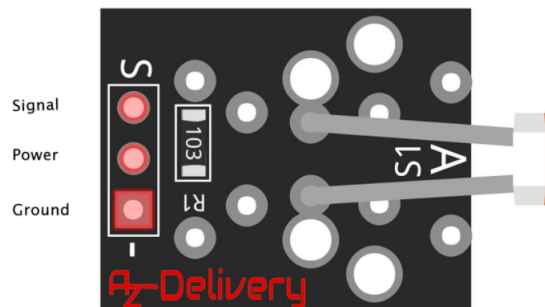


Figura 7: Pinout do photo LDR resistor.

3.2.1.5 Atuador bomba de água (DC 3V/5V)

Esta bomba de água [11] é basicamente um motor DC que é alimentado com 3V/5V. Quando se encontra ligado, a bomba suga a água pela lateral do invólucro de plástico e liberta para fora da porta do tubo. Mudar a polaridade deste atuador não vai transformar num dispositivo de sucção, este actuador apenas tem a função de bombear a água e apenas só. Este vai ter a função de regar os nossos vegetais que se encontram na estufa.

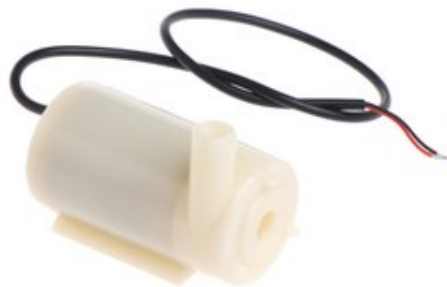


Figura 8: Mini Water Pump.

3.2.1.6 Atuador *Stepper Motor*

Os *stepper motor* [13] já se encontram disponíveis no mercado há muito tempo, sendo utilizados em bastantes aplicações. Este tem um pequeno tamanho, mas é bastante eficiente em termos de energia.

Este atuador é constituído por um pequeno motor, potenciómetro e um circuito de controlo. À medida que este motor gira a resistência do potenciómetro muda de modo a que o circuito de controlo possa regular precisamente quando existe movimento. Quando este motor se encontrar na posição desejada, a alimentação que é fornecida ao motor vai ser interrompida, se ainda não se encontrar na posição desejada, o motor continua a girar na direcção desejada. Em baixo encontra-se um esquema que demonstra o funcionamento deste.

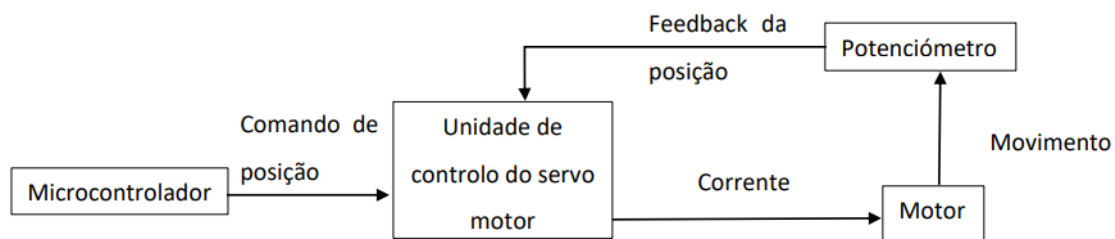


Figura 9: Esquema do step motor

3.2.2. Software

No lado do micro-dispositivo, em termos de software este terá um funcionamento bastante simples, e ao mesmo tempo extremamente funcional e pensado em prol do consumidor final.

A estrutura do código está dividida em várias etapas:

- Configuração da rede domestica;
- Conexão com o servidor NTP (Network Time Protocol) e TOPT (*Time-based One-Time Password*);
- Conexão com o *broker* MQTT;
- Inicialização dos vários módulos:
 - Módulo do sensor de Luminosidade;
 - Módulo do sensor de humidade do solo;
 - Módulo do sensor de temperatura e humidade do ar;
 - Módulo do atuador do *Stepper Motor*;
 - Módulo do atuador da bomba de água;
- Registo do dispositivo;
- Registo das amostras recolhidas;
- Verificação dos limites estabelecidos pelo perfil;

3.2.2.1 Configuração da rede e funcionamento

Este, na primeira vez que é iniciado, inicializa uma rede *hotspot* com um *webserver*, para que o utilizador possa configurar a sua rede domestica, evitando que o nome da rede e palavra passe estejam *hardcoded*.

Estas configurações da rede são gravadas na memória *flash* do dispositivo para que não seja necessário configurar a rede sempre que o dispositivo é desligado. Caso seja necessário alterar as configurações, esta implementado um detetor de duplo *reset*, para que o dispositivo volte o modo de configuração de rede.

3.2.2.2 Conexão com o servidor NTP (Network Time Protocol) e TOPT (*Time-based One-Time Password*)

Esta etapa e configuração tem o propósito de fornecer um pouco mais de segurança a toda a comunicação feita entre o dispositivo e a *cloud*.

A conexão com o servidor NTP permite que o tempo do dispositivo esteja sincronizado com um servidor exterior, complementando assim o módulo TOTP que gera palavras passes baseadas no tempo introduzido. O TOTP requer ao início que seja introduzido uma *hash* de onde este deriva as suas palavras passes.

Desta forma é possível que tanto a *cloud* como o dispositivo gerem a mesma palavras passe, uma vez que a *hash* introduzida é derivada da identificação única do dispositivo.

3.2.2.3 Conexão com o *broker* MQTT e funcionamento

Com a conexão à internet ativa, o dispositivo estabelece uma ligação ao *broker* predefinido e subscrive a um tópico exclusivo a ele. Nesta configuração é também definida uma função chamada de *callback*, que é a função para onde o programa entra quando recebe dados no tópico subscrito.

O propósito desta subscrição é comunicar de forma direta com o dispositivo.

Os dados recebidos aqui estão em formato csv, e as possíveis mensagens são as seguintes:

- "[totp];0;[0/1]" - Esta mensagem está destinada ao controlo do motor para abertura ou fecho da estufa. O segundo campo é referente ao motor e o terceiro ao estado pretendido (*Off/ON*);
- "[totp];1;1" - Esta mensagem está destinada ao controlo da bomba da água. A designação dos campos igual á mensagem anterior;
- "[totp];2;[tempMax];[tempMin];[humArMax];[humArMin];[humSoloMax];[humSoloMin]" - Esta mensagem está destinada a definir o novo perfil da estufa, definindo os limites para atuar;

3.2.2.4 Módulos dos sensores usados e funcionamento

Estes módulos são respetivos aos sensores utilizados, luminosidade, temperatura e humidade do ar e humidade do solo. E todos eles possuem um funcionamento muito simples e semelhante.

Na inicialização destes, são configurados os respetivos pinos. E nas leituras todos as amostras exceto a temperatura, são normalizadas para percentagem.

3.2.2.5 Módulos dos atuadores usados e funcionamento

Estes módulos são respetivos aos atuadores utilizados, bomba de água e o *stepper motor*. E como os sensores também possuem um funcionamento simples e semelhante.

O módulo do *stepper motor* quando é inicializado, recebe o vários parâmetros de configuração. Este pode atuar de várias maneiras:

- Abrir a estufa, caso a esta esteja fechada;
- Fechar a estufa, caso a esta esteja aberta;
- Definir os máximos e mínimos (são inicializados numa primeira vez com valores predefinidos);
- Verificar os limites de temperatura e humidade do ar definidos, e abrir caso os máximos sejam ultrapassados ou fechar no caso dos mínimos.

No módulo da bomba de água, repete-se o funcionamento descrito anteriormente com pequenas diferenças:

- Regar, enquanto o máximo não é ultrapassado;
- Definir os máximos e mínimos (são inicializados numa primeira vez com valores predefinidos);
- Verificar os limites de humidade do solo definido, e regar caso o valor esteja abaixo do mínimo.

3.2.2.6 Registo do dispositivo e amostras

O registo do dispositivo, é efetuado sempre na Inicialização do dispositivo e apenas uma vez. Este é enviado para a *cloud* através de um método *POST* do protocolo HTTP, este pedido contém a identificação única do dispositivo e uma porção do *hash* gerado a partir da identificação única.

O registo das amostras é efetuado a cada 2 minutos e como o pedido anterior também é enviado por HTTP para a *cloud*. Este pedido contém a identificação única do dispositivo, o *timestamp*, a temperatura e humidade do ar, humidade do solo, a luminosidade e o estado da estufa (aberta/fechado).

3.2.2.7 Verificação dos limites estabelecidos pelo perfil

A verificação dos limites é efetuada a cada minuto, e consiste na chamada das funções, de verificação, descritas anteriormente nos módulos dos atuadores. Este processo pode ser representado pelo seguinte fluxograma 10.

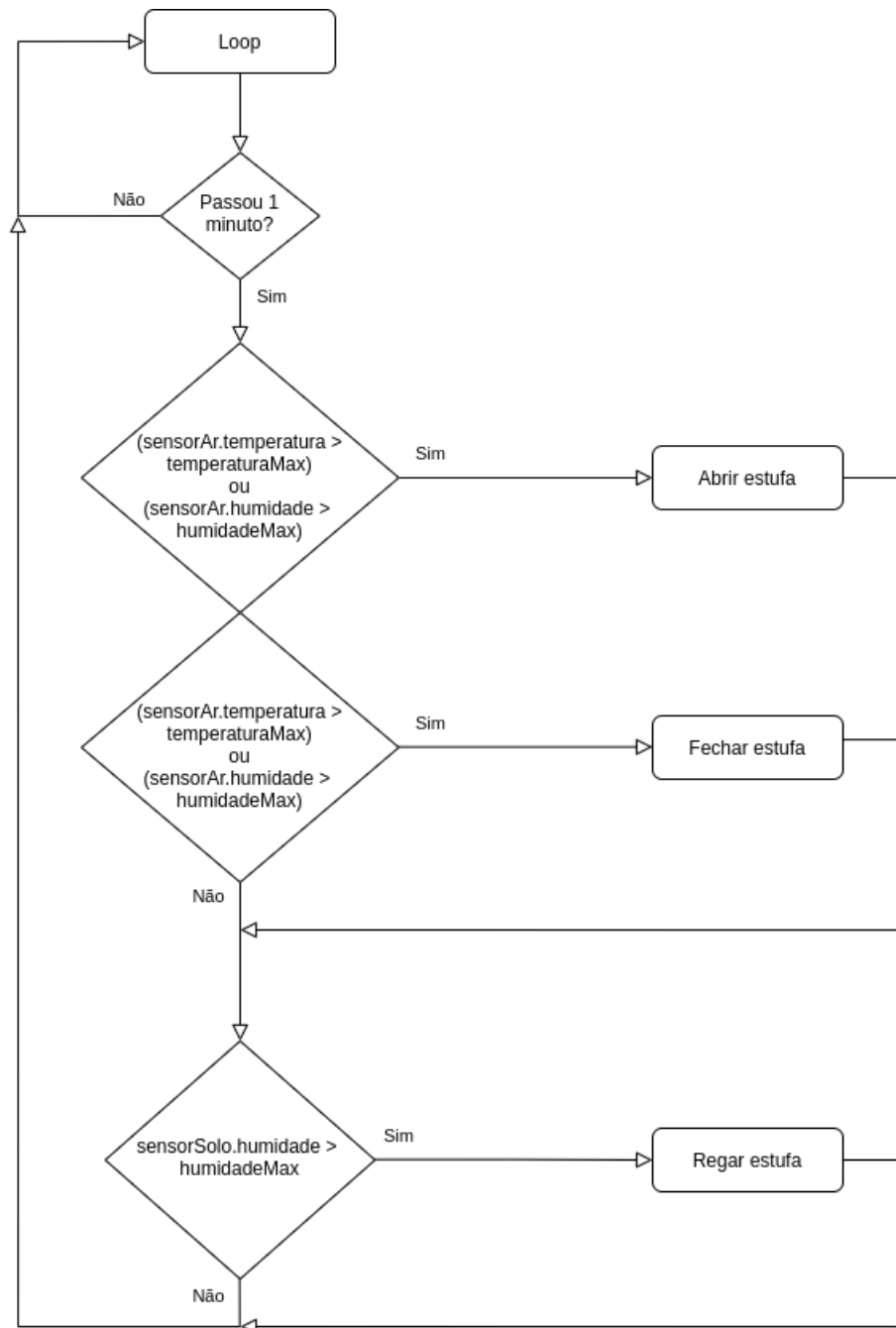


Figura 10: Fluxograma da verificação dos limites do perfil

3.2.2.8 Bibliotecas utilizadas

Para toda a realização do código do dispositivo, foram utilizadas as seguintes bibliotecas:

- *DNSServer.h*;
- *WiFi.h*;
- *WebServer.h*;
- *ESP_WiFiManager.h*;
- *ESP_DoubleResetDetector.h*;
- *TOTP.h*;
- *NTPClient.h*;
- *WiFiUdp.h*;
- *HTTPClient.h*;
- *DHT.h*;
- *Stepper.h*;
- *PubSubClient.h*;
- *mbedtls/md.h*;

3.3. *Cloud* IoT

3.3.1. BackEnd

3.3.1.1 Broker MQTT

A sigla MQTT traduz-se em *Message Queuing Telemetry Transport* e refere-se a um protocolo emergente nesta área dos IoT baseado na pilha TCP/IP. Este protocolo permite uma comunicação assíncrona entre as partes intervenientes, geralmente de máquina para máquina, de uma forma simples, leve e eficaz utilizando um *broker* que serve de intermediário.

Este é um protocolo com baixos requisitos de largura de banda e de *hardware*, poderá também ser considerado pouco fiável, por razões de alta latência. Mas visto que na nossa aplicação não possui esse tipo de exigências, este protocolo é a escolha ideal. Além disso, as mais recentes versões prevêem a possibilidade de utilizar diferentes níveis de QoS, garantindo uma qualidade superior na entrega das mensagens.

O *broker* funciona com um sistema de publicações e subscrições e é responsável por geri-las, permitindo que uma publicação seja entregue a vários subscritores. Tem como funções: receber todas as mensagens, filtrar as mensagens recebidas e publicar as mensagens para todos os clientes subscritos. Resumindo, funciona como um *Hub* central por onde passam todas as mensagens.

O MQTT utiliza uma *string* UTF-8 que o *broker* utiliza para filtrar mensagens. A essa *string* chamamos de tópico. Os tópicos são subdivididos por diferentes níveis separados por uma barra. No nosso trabalho, os tópicos têm a seguinte forma: `/[Identificação do dispositivo]/[in/out].[16]`

Este protocolo traz vantagens para o nosso projeto porque irá ser por este meio que iremos obter os estados dos sensores em *near-real-time*, bem como a execução de algumas operações. O *broker* que estamos a utilizar é o DIOITY [<http://mqtt.dioty.co/>]. [17]

3.3.1.2 API

Para a comunicação entre duas aplicações de software, onde existe uma requisição de serviços entre essas aplicações, é necessário código que consiga realizar corretamente esta troca de dados.

O grupo decidiu que iria usar uma *Rest* API, um tipo de API que utiliza pedidos HTTP para a gestão de dados na comunicação de aplicações do tipo *web*.

As metodologias HTTP utilizadas por este tipo de API são as seguintes:

- Método **GET** - obter um recurso;
- Método **PUT** - mudar ou atualizar um recurso;
- Método **POST** - criar um recurso;
- Método **DELETE** - eliminar um recurso.

Salienta-se que estas metodologias são definidas pelo protocolo HTTP (*Hypertext Transfer Protocol*).

Este tipo de API é muito usado em situações *web* devido às rotinas que são usadas serem *stateless*, ou seja, sem estado associado onde os pedidos entre servidor *web* e serviços associados são baseados na informação gerada na interação entre os dois. Isto torna-se determinante pois permite que este sistema seja escalável e de fácil reutilização de serviços em caso de falha de algum deles [15].

O grupo, desenvolveu uma API própria e bem definida, composta por todas as rotas essenciais para a manipulação dos dados recebidos pelo *gateway* do sistema e para envio para e, na tabela 1, podemos visualizar esses mesmos recursos:

Tabela 1: Recursos da API.

| URI | Método HTTP | Descrição | Parâmetros |
|------------------------------|-------------|------------------------------------|---|
| /register | POST | Regista utilizador | -username -mail -pass |
| /utilizadores | PUT | Atualiza dados de utilizador | -username -mail -pass |
| /utilizadores | DELETE | Remove utilizador | -username -mail -pass |
| /login | POST | Efetua o login | -username -pass |
| /logout | GET | Efetua logout | |
| /utilizadores/ | GET | Obtém a lista de utilizadores | |
| /utilizadores/:id_user | GET | Obtém utilizador | -id_user |
| /devices | POST | Regista um dispositivos | -serial_number -registcode |
| /devices | GET | Obtém todos os dispositivos | |
| /devices/:serial_number | DELETE | Remove dispositivo específico | -serial_number |
| /history | POST | Adiciona uma entrada no histórico | -serial_number -timest -temp -hum_air -hum_earth -luminosity -states |
| /history | GET | Obtém todo o histórico | |
| /history/:serial_number | GET | Todo histórico de um dispositivo | -serial_number |
| /history/:serial_number | DELETE | Remove histórico de um dispositivo | -serial_number |
| /profiles | POST | Adiciona uma entrada de perfil | -designacao -temp_min -temp_max -hum_air_min -hum_air_max -hum_earth_min -hum_earth_max |
| /profiles | GET | Obtém todos os perfis | |
| /profiles/:id_perfil | GET | Obtém um perfil específico | -id_perfil |
| /profiles/:id_perfil | DELETE | Remove um perfil específico | -id_perfil |
| /reluser | POST | Adiciona uma estufa | -id_user -serial_number -designacao |
| /reluser | GET | Obtém todas as estufas | |
| /reluser/:id_rel_user_device | GET | Obtém uma estufa específica | -id_rel_user_device |
| /reluser/:id_rel_user_device | DELETE | Remove uma estufa específica | -id_rel_user_device |

A nossa API mudou substancialmente em relação à idealizada pelo grupo e que foi apresentada no relatório intermédio pois as nossas necessidades, à medida que o se ia avançando no desenvolvimento da comunicação entre servidor e dispositivos, fizeram com que tivéssemos de reformular a API, nomeadamente, nas rotas que realizam os pedidos HTTP.

3.3.1.3 Base de Dados

Para complementar o sistema, a base de dados irá servir para guardar a relação entre os utilizadores e os dispositivos, bem como um histórico de medições que cada dispositivo efectua regularmente.

Esta irá também guardar uma colecção de perfis predefinidos para controlar diversos tipos de culturas que posteriormente serão enviados para os dispositivos. Estes perfis poderão ser usados tal como definido ou com os parâmetros alterados antes de serem enviados para os dispositivos.

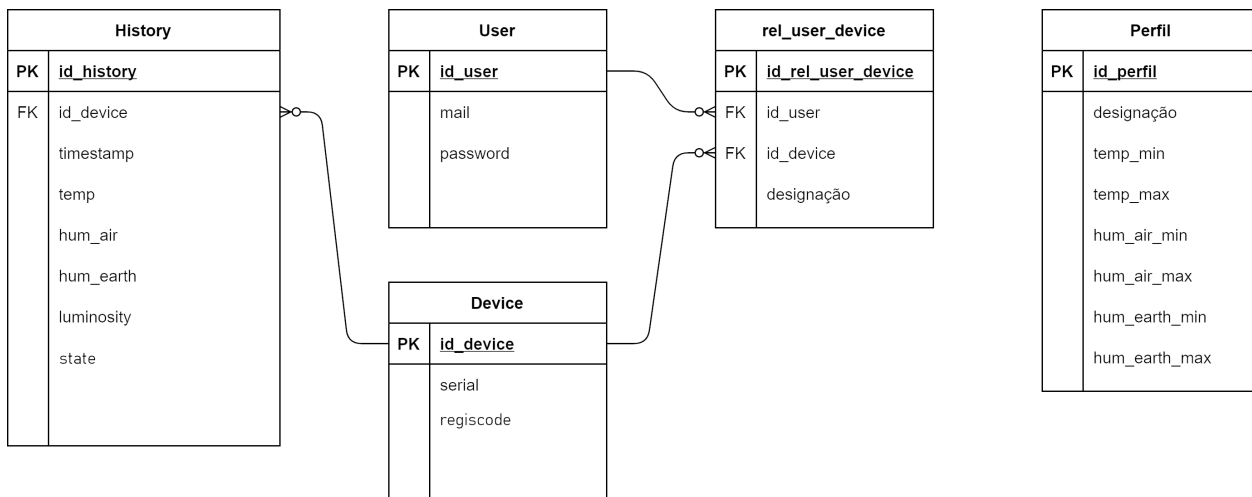


Figura 11: Base de dados.

A base de dados foi implementada em *MySQL*, tendo sido criada uma instância de ligação na máquina local para que seja possível de ser acedida pelo servidor *nodeJS*, implementado que corre na mesma máquina, de modo a executar-se as operações necessárias sobre a base de dados para que todos os dados sejam guardados com conformidade e de acordo com o pretendido.

A vantagem de usar *MySQL* é muito grande, pois exige que a base de dados seja relacional, o que faz com que haja uma boa integridade dos dados e facilidade na manipulação destes através de operações (*query*) já implementadas na construção da BD.

A seguir é possível visualizar duas imagens que representam excertos de código SQL realizado pelo grupo para definição da nossa BD:

3.3.1.4 Servidor *nodeJS*

O servidor implementado pelo grupo é uma parte crítica de todo o funcionamento deste sistema que foi desenvolvido, pois é ele que trata de receber todos os dados vindos dos vários dispositivos que se tenham registado no serviço desenvolvido, de colocar os dados na base de dados e de enviar os dados para o *frontend* desenvolvido de modo a que um utilizador deste serviço consiga visualizar os dados necessários e controlar as suas estufas registadas de uma maneira simples, eficiente e organizada.

Para isso, foi usada a ferramenta *nodeJS* que nos permite executar código para *backend* e *frontend*, em linguagem *JavaScript*, numa única instância. Esta instância permite que vários serviços sejam interligados pelo um nó central capaz de receber e enviar informação fornecida por esses serviços e vice-versa, sendo que, no nosso caso, servirá para:

- Receber dados dos dispositivos que controlam as estufas registadas pelos utilizadores deste sistema;
- Armazenamento e controlo de dados na BD conectada ao servidor;
- Receber ou enviar respostas aos pedidos de utilizadores e aplicação *web* do sistema;
- Infraestrutura de comunicação para o envio e recepção de dados imediatos entre aplicação *web* e os dispositivos das estufas através do agente usado *BrokerMQTT*;

O nosso servidor é composto pelos seguintes ficheiros:

- *server.js*;
- *app.js*;
- *servBD.js*.

O ficheiro *server.js* contém todo o código necessário para instanciar o servidor e todos os serviços agregados a ele. Ao ser executado, cria um servidor local com duas portas abertas para ligação à aplicação *web* e para os pedidos HTTP recebidos através das rotas criadas pelo grupo para a nossa API e já mencionadas neste relatório.

O ficheiro *app.js* contém todas as rotas definidas para as operações implementadas na nossa API, ou seja, define os *endpoints* para os pedidos HTTP dos serviços que pretendam enviar e guardar informação têm que usar.

A seguir, podemos visualizar na imagem 12 um excerto de código que representa uma rota para um pedido HTTP presente no ficheiro:

```
app.get('/history/:serial_number', verifyToken, function (req, res) {
  servBD.getHistoryById(req, function (error, results) {
    if (error) {
      res.status(error.code).send(error.message);
    }
    else {
      return res.status(status.OK).send({ error: false, data: results, message: 'Device history' });
    }
  });
});
```

Figura 12: Exemplo de um *endpoint* de uma rota.

Podemos ver que esta rota permite que um pedido HTTP GET consiga obter uma resposta com a informação acerca do estado (histórico) de um dispositivo em específico. A resposta será a informação obtida pela *query* definida na nossa API **`servBD.getHistoryById(req, callback)`**.

Por último, o ficheiro *servBD.js* é essencialmente a nossa API, onde estão todas as funções necessárias para as respostas aos pedidos HTTP definidos pelas nossas rotas. Ao receber-se um pedido no servidor, este tem a ele associado uma *query* para operação sobre a nossa base de dados, que pode ser um INSERT, SELECT, UPDATE ou mesmo um DELETE. Depois de efetuar o pedido, o servidor devolve uma resposta que pode ser uma confirmação ou um conjuntos de dados específicos, dependendo do pedido realizado.

Na imagem 13, podemos visualizar uma das funções presentes na nossa API que efetua uma resposta a um pedido HTTP que, neste caso, é uma resposta a um pedido de inserção de um dispositivo na base de dados:

```
insertDevice: function (req, callback) {
  if (!req.body.serial_number || !req.body.registcode) {
    let err = { code: status.BAD_REQUEST, message: "Please provide a device" };
    return callback(err, null);
  } else {
    //console.log(req.body.serial);
    let query = "call insert_device(?,?)";
    let table = [req.body.serial_number, req.body.registcode];
    query = mysql.format(query, table);
    pool.query(query, function (error, results) {
      if (error) {
        err = { code: status.INTERNAL_SERVER_ERROR, message: error };
        console.log(JSON.stringify(err));
        return callback(err, null);
      }
      else {
        console.log(JSON.stringify(results));
        return callback(null, results);
      }
    });
  }
},
```

Figura 13: Exemplo de uma função de resposta.

3.4. Aplicação *web*

3.4.1. *User Interface* (UI)

A interface do utilizador da nossa aplicação *web* inicia com uma página de boas-vindas "*Home*" onde os utilizadores serão convidados a efetuar um registo ou, caso já possuam uma conta, efetuar o *login*.

Após o *login*, é apresentada ao utilizador uma *dashboard* principal, onde são mostradas informações relativas às várias estufas do utilizador em sessão. Por estufa, são apresentados os seus dados de identificação, bem como informações sobre a temperatura, níveis de humidade, luminosidade e o estado atual da estufa (aberta/fechada).

Também é possível adicionar novas estufas, predefinir novos perfis e utilizá-los na configuração das estufas. Cada estufa apresentada possui também ferramentas para que o cliente facilmente possa tomar ações sobre ela, como por exemplo, abrir e fechar a mesma.

Paralelamente aos dados em tempo real, é possível consultar o histórico de cada estufa, isto é, consultar as diversas amostras enviadas ao longo do tempo que mostram a evolução do estado da estufa e dos seus parâmetros ao longo do tempo. O design será responsivo e poderá ser usado em *web*.

3.4.2. Frontend

Nesta secção, vamos apresentar a nossa interface, recorrendo a imagens. Na figura 14, podemos ver a nossa página inicial de boas vindas.



Figura 14: Página de boas-vindas.

Nesta página inicial, encontramos também uma barra de opções, apresentada na figura 15

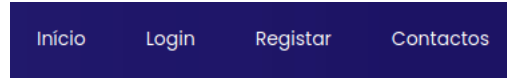
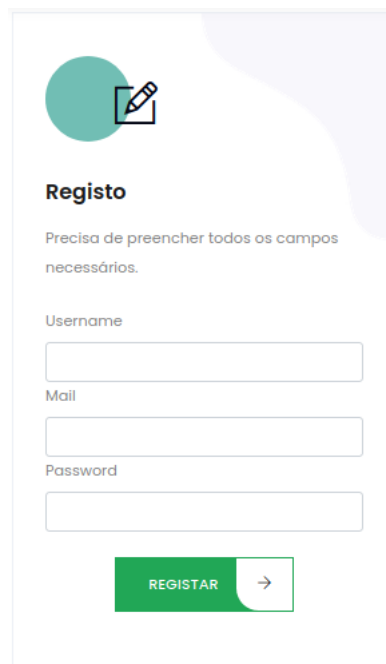


Figura 15: Barra de opções da página de boas-vindas.

Na opção "Registar", somos redireccionados para a página de registo, que podemos ver na figura 16. No registo, é pedido ao utilizador para criar um *username*, indicar um *e-mail* e uma *password*.

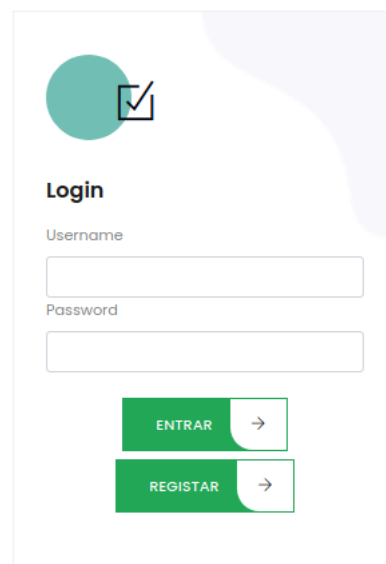
Quando este processo é finalizado com sucesso, é executado um *Post* no servidor para guardar estes dados na base de dados. É criada também uma *Hash* da *password* escolhida.



The registration form, titled "Registo", features a teal circle icon with a pencil. It includes a message: "Precisa de preencher todos os campos necessários." Below this are three input fields labeled "Username", "Mail", and "Password". At the bottom is a green button labeled "REGISTAR" with a right-pointing arrow.

Figura 16: Página de registo.

Após o registo, o utilizador deverá efetuar o *login*, fornecendo o seu *username* e a *password*. Para verificar se um utilizador está registado, no processo do *login* é efetuado um *Post* e criada uma *Hash* da password introduzida. Ao comparar a *Hash* da password introduzida com a que tinha sido criada no registo, podemos verificar se a password está correta e se existe o registo. Na figura 17, vemos a nossa página de *login*.



The login form, titled "Login", features a teal circle icon with a checkmark. It includes two input fields labeled "Username" and "Password". Below these are two green buttons: "ENTRAR" with a right-pointing arrow, and "REGISTAR" with a right-pointing arrow.

Figura 17: Página de *login*.

Quando o *login* é efetuado com sucesso, a página inicial é a *Dashboard* onde são apresentadas as estufas do utilizador e o seu estado atual. Na barra de opções também são apresentadas novas opções que serão apresentadas em baixo - figuras 18 e 19.

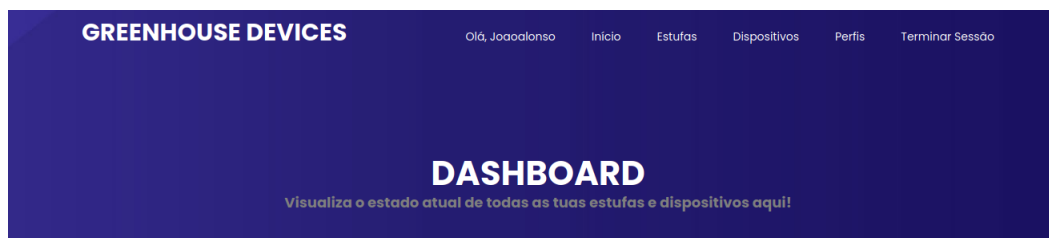


Figura 18: Página da *Dashboard*.

Nesta página, também podemos gerir as ações que queremos efetuar na nossa estufa. Isto é, existem 3 botões de opção - abrir, fechar e regar - que, quando selecionados, fazem uma publicação pelo *Broker* MQTT para abrir e fechar a estufa ou para ativar a rega.



Figura 19: Página da *Dashboard*.

Quando o utilizador pretende adicionar uma nova estufa, deve registá-la clicando na opção "Estufas". Nesta opção somos redireccionados para uma página de registo de estufa que podemos ver na figura 20.

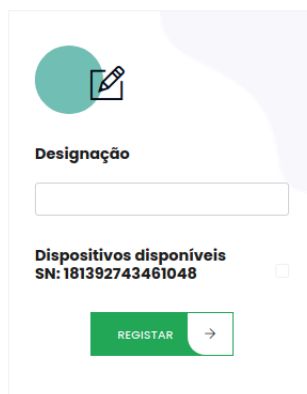


Figura 20: Página de registo de nova estufa.

Depois de adicionar uma nova estufa, esta irá aparecer na página da *Dashboard*.

Na opção "Dispositivos", podemos observar os dispositivos que temos ligados e observar diferentes amostras com as diferentes variáveis. Temos, como exemplo, a figura 21.

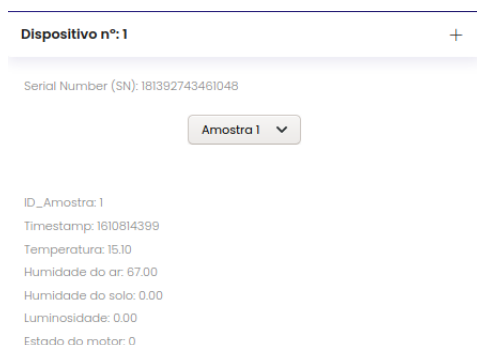


Figura 21: Página de controlo de dispositivos e amostras.

Por fim, a nossa aplicação permite a utilização de diferentes perfis predefinidos. Isto é, podemos definir quais as condições da nossa estufa, escolhendo os limites máximo e mínimo dos níveis de temperatura e humidade do ar e do solo. Podemos observar na figura 22 o exemplo do nosso Perfil de Tomates.

Dispositivo 1 (SN): 181392743461048



Perfil nº: 1



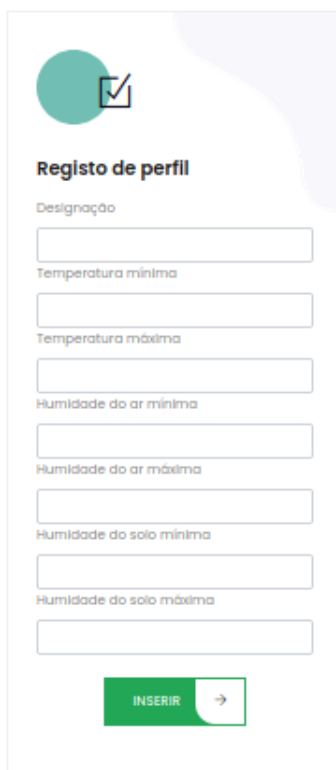
Designação: Perfil de Tomates
Temperatura mínima: 22
Temperatura máxima: 23
Humidade do ar mínima: 50
Humidade do ar máxima: 60
Humidade do solo mínima: 40
Humidade do solo máxima: 50



Figura 22: Página de perfis.

Após a criação do perfil, para ativar e utilizar as medidas definidas pelo mesmo, o utilizador selecciona quais os dispositivos nos quais o pretende usar - usando um botão *toggle* - e ativa o perfil clicando em "ativar", enviando esse perfil para o nosso Arduino através de uma publicação MQTT.

O utilizador poderá também criar novos perfis que pretenda utilizar. Para tal, criamos um registo de perfis, onde são introduzidas as condições pretendidas pelo utilizador, como vemos na figura 23.



Registo de perfil

Designação

Temperatura mínima

Temperatura máxima

Humidade do ar mínima

Humidade do ar máxima

Humidade do solo mínima

Humidade do solo máxima

INSERIR →

Figura 23: Página de registo de perfis.

4. Testes, Resultados e Discussão

4.1. Sessão inicial de configuração da rede doméstica

Como foi referido anteriormente, o dispositivo na primeira vez que iniciado ou caso seja detetado duplo *reset*, inicia um *webserver* com uma página para configuração da rede doméstica.

The screenshot shows a web interface titled "Configuration". It contains three main sections:

- Network Status:** A box showing three items with lock icons and percentages: [MEO-DIAS](#) (38%), [MEO-4E4930](#) (32%), and [MEO-WIFI](#) (28%).
- Wi-Fi Settings:** A box with four input fields: "SSID" (containing "SSID"), "Password" (containing "password"), "SSID1" (containing "SSID1"), and "Password" (containing "password1").
- Static IP Settings:** A box with five input fields, all containing "0.0.0.0": "Static IP", "Gateway IP", "Subnet", "DNS1 IP", and "DNS2 IP".

Figura 24: Página de configuração da rede.

4.2. Comunicação entre estufa e a plataforma

De forma a testar a comunicação entre o dispositivo e a plataforma, coloca-mos múltiplos *prints* nas zonas relevantes do código do dispositivo, sabendo assim exatamente onde passa quando há comunicação.

Na figura 25, estão os tópicos (MQTT) que serão utilizados pelo do dispositivo em questão.

```
Input topic:
  /augustocesarsilvamota@gmail.com/181392743461048/in
Output topic:
  /augustocesarsilvamota@gmail.com/181392743461048/out
```

Figura 25: Tópicos (MQTT) do dispositivo.

Na figura 26, verificamos o conteúdo do pedido POST que é enviado no registo do dispositivo.

```
Post: {"serial_number":"181392743461048","registcode":"d72830b1debe062ebbe7"}
```

Figura 26: POST do registo do dispositivo.

Na figura 28, verificamos o conteúdo do pedido POST que é enviado no envio das amostras.

```
Post: {"serial_number":"181392743461048","timest":"1610917962","temp":"18.30","hum_ai
r":"60.00","hum_earth":"42.00","luminosity":"44.13","states":"1"}
```

Figura 27: POST das amostras.

Na figura 28, verificamos o conteúdo do pedido POST que é enviado no envio das amostras.

```
Post: {"serial_number":"181392743461048","timest":"1610917962","temp":"18.30","hum_ai
r":"60.00","hum_earth":"42.00","luminosity":"44.13","states":"1"}
```

Figura 28: POST das amostras.

Agora vamos enviar um perfil novo para o dispositivo.

Na figura 29, verificamos as definições deste novo perfil.

Dispositivo 1 (SN): 181392743461048 🔴

Perfil n°: 1 +

Perfil n°: 2 +

Designação: Broculos
 Temperatura mínima: 10
 Temperatura máxima: 20
 Humidade do ar mínima: 70
 Humidade do ar máxima: 80
 Humidade do solo mínima: 60
 Humidade do solo máxima: 80

ATIVAR
→

Figura 29: Definições do perfil.

Na figura 30, verificamos a chegada do novo perfil, e de seguida por mera coincidência observamos também o processo de verificação do perfil e a ação regar.

```
Set Profile
tempArMax: 20.000000
tempArMin: 10.000000
humArMax: 80.000000
humArMin: 70.000000
humSoloMax: 80.000000
humSoloMin: 60.000000
Verificar temperatura e humidade
Verificar niveis de agua
Regar
=====Debug=====
Estufa fechada
Percentagem de luz: 46.54%
Temperatura do ar: 19.00°C
Percentagem de Humidade do ar: 59.00%
Percentagem de humidade da terra: 49.63%
=====
Post: {"serial_number":"181392743461048","timest":"16109186
```

Figura 30: Receção de perfil e atuação.

Por fim, vamos enviar um pedido de abertura através do botão abrir (Figura 31).

Broculos



Dispositivo (SN): 181392743461048

Temperatura: 18.60 °C

Humidade do ar: 60.00%

Humidade do solo: 47.69%

Luminosidade: 47.86%

Estado da estufa: Fechada



Figura 31: Ações da estufa.

Na figura 30, verificamos a abertura da estufa, e de seguida, novamente, por mera coincidência observamos também o processo de verificação do perfil, que fecha a estufa. Porque tanto a temperatura como humidade do ar estão abaixo dos limites do perfil descrito anteriormente.

```
Callback verificado
Abrir
Broker
Verificar temperatura e humidade
Fechar
Verificar niveis de agua
Regar
=====Debug=====
Estufa fechada
Percentagem de luz: 47.25%
Temperatura do ar: 18.60°C
Percentagem de Humidade do ar: 60.00%
Percentagem de humidade da terra: 50.53%
=====
```

Figura 32: Abertura e fecho da estufa.

4.3. Conteúdo da base de dados

Nas imagens 33, 34, 35 e 36 podemos visualizar o conteúdo das tabelas *User*, *Device*, *Perfil* e *rel_user_device* (representa uma estufa) da nossa base de dados e todos os campos respetivos de cada tabela após o serviço ser usado e terem sido feitos alguns testes:

```
mysql> SELECT * FROM User;
```

| id_user | username | mail | pass |
|---------|------------|-----------------------------|--|
| 1 | pedroar | pedro@gmail.com | \$2a\$08\$5QAhRI/12t9Vuzw7qWDpUefM4JNSt044zwlWnfINVSHOfGKJ6uAC |
| 2 | hugo | hugo@mail.com | \$2a\$08\$jrm1wr7F9hnN6qBDCqD68eAixf6ujo1Z2TH9WXvnY9bhOM8M1959a |
| 3 | pedroar2 | pedroar@gmail.com | \$2a\$08\$Rg9207sPeAMeW234o6H5FeA98WpNkDHJA2nqpNOGzoFfviSHGWDZ2 |
| 4 | joaoalonso | joaoalonsobarbosa@gmail.com | \$2a\$08\$axvQ.WG2o1Eo8bopByCLvOecPOUPI/Zj1uvfw7UVh..UwWr2VfBeC |
| 5 | telitos | telitos@mail.com | \$2a\$08\$QB7kh2NaxxK0Ic5NcSYQ...z012SJntoDY6ofjqIi9Sf7a5NhiXzdi |

```
5 rows in set (0.00 sec)
```

Figura 33: Tabela *User*.

```
mysql> USE GHD_DB
Database changed
mysql> SELECT * FROM Device;
```

| serial_number | resgistcode |
|-----------------|----------------------|
| 181392743461048 | d72830b1debe062ebbe7 |

```
1 row in set (0.00 sec)
```

Figura 34: Tabela *Device*.

```
mysql> SELECT * FROM Perfil;
```

| id_perfil | designacao | temp_min | temp_max | hum_air_min | hum_air_max | hum_earth_min | hum_earth_max |
|-----------|-------------------|----------|----------|-------------|-------------|---------------|---------------|
| 1 | Perfil de Tomates | 22 | 23 | 50 | 60 | 40 | 50 |
| 2 | Broculos | 10 | 20 | 70 | 80 | 60 | 80 |

```
2 rows in set (0.00 sec)
```

Figura 35: Tabela *Perfil*.

```
mysql> SELECT * FROM rel_user_device;
```

| id_rel_user_device | id_user | serial_number | designacao |
|--------------------|---------|-----------------|---------------|
| 1 | 1 | 181392743461048 | Estufa da GNZ |
| 2 | 2 | 181392743461048 | Broculos |
| 3 | 4 | 181392743461048 | Estufa |

```
3 rows in set (0.00 sec)
```

Figura 36: Tabela *rel_user_device*.

A tabela a seguir representada na imagem 37 é a que contém todos dados relativos às amostras enviadas periodicamente pelos dispositivos das estufas e possui os seguintes campos: *id_history*, *serial_number*, *timest*, *temp*, *hum_air*, *hum_earth*, *luminosity* e *states*.

| | | | | | | | | | | | | | | | |
|----|--|-----------------|--|------------|--|-------|--|-------|--|-------|--|-------|--|---|--|
| 41 | | 181392743461048 | | 1610819188 | | 17.70 | | 55.00 | | 0.00 | | 44.37 | | 0 | |
| 42 | | 181392743461048 | | 1610819308 | | 17.60 | | 55.00 | | 0.00 | | 42.56 | | 0 | |
| 43 | | 181392743461048 | | 1610819428 | | 17.70 | | 55.00 | | 0.00 | | 43.35 | | 0 | |
| 44 | | 181392743461048 | | 1610819549 | | 17.70 | | 56.00 | | 0.00 | | 42.52 | | 0 | |
| 45 | | 181392743461048 | | 1610819669 | | 17.70 | | 56.00 | | 0.00 | | 41.22 | | 0 | |
| 46 | | 181392743461048 | | 1610819789 | | 17.80 | | 57.00 | | 0.00 | | 42.64 | | 0 | |
| 47 | | 181392743461048 | | 1610819909 | | 17.80 | | 57.00 | | 0.00 | | 43.10 | | 0 | |
| 48 | | 181392743461048 | | 1610820030 | | 17.70 | | 57.00 | | 0.00 | | 42.59 | | 0 | |
| 49 | | 181392743461048 | | 1610820150 | | 17.80 | | 57.00 | | 0.00 | | 42.59 | | 0 | |
| 50 | | 181392743461048 | | 1610820270 | | 17.90 | | 57.00 | | 0.00 | | 40.05 | | 0 | |
| 51 | | 181392743461048 | | 1610820390 | | 17.90 | | 61.00 | | 0.00 | | 40.15 | | 0 | |
| 52 | | 181392743461048 | | 1610820510 | | 18.00 | | 62.00 | | 0.00 | | 40.59 | | 0 | |
| 53 | | 181392743461048 | | 1610820631 | | 18.10 | | 63.00 | | 0.00 | | 40.61 | | 0 | |
| 54 | | 181392743461048 | | 1610820789 | | 18.20 | | 61.00 | | 0.00 | | 42.20 | | 1 | |
| 55 | | 181392743461048 | | 1610820795 | | 18.30 | | 60.00 | | 0.00 | | 43.96 | | 1 | |
| 56 | | 181392743461048 | | 1610820800 | | 18.30 | | 60.00 | | 0.00 | | 42.93 | | 1 | |
| 57 | | 181392743461048 | | 1610820806 | | 18.30 | | 59.00 | | 0.00 | | 40.78 | | 1 | |
| 58 | | 181392743461048 | | 1610820811 | | 18.30 | | 60.00 | | 0.00 | | 41.37 | | 1 | |
| 59 | | 181392743461048 | | 1610820817 | | 18.30 | | 59.00 | | 0.00 | | 42.71 | | 1 | |
| 60 | | 181392743461048 | | 1610820823 | | 18.20 | | 59.00 | | 0.00 | | 42.56 | | 1 | |
| 61 | | 181392743461048 | | 1610820828 | | 18.60 | | 59.00 | | 0.00 | | 41.47 | | 1 | |
| 62 | | 181392743461048 | | 1610820834 | | 18.40 | | 59.00 | | 0.00 | | 42.64 | | 1 | |
| 63 | | 181392743461048 | | 1610820839 | | 18.30 | | 59.00 | | 0.00 | | 42.27 | | 1 | |
| 64 | | 181392743461048 | | 1610820845 | | 18.30 | | 58.00 | | 0.00 | | 40.22 | | 1 | |
| 65 | | 181392743461048 | | 1610820851 | | 18.20 | | 58.00 | | 0.00 | | 42.05 | | 1 | |
| 66 | | 181392743461048 | | 1610820857 | | 17.90 | | 57.00 | | 0.00 | | 41.00 | | 1 | |
| 67 | | 181392743461048 | | 1610820862 | | 18.30 | | 58.00 | | 0.00 | | 43.08 | | 1 | |
| 68 | | 181392743461048 | | 1610912737 | | 18.40 | | 62.00 | | 39.58 | | 51.48 | | 1 | |
| 69 | | 181392743461048 | | 1610912857 | | 18.40 | | 61.00 | | 39.94 | | 50.65 | | 1 | |
| 70 | | 181392743461048 | | 1610912977 | | 18.50 | | 61.00 | | 40.81 | | 51.50 | | 1 | |
| 71 | | 181392743461048 | | 1610913097 | | 18.50 | | 61.00 | | 41.36 | | 50.01 | | 1 | |
| 72 | | 181392743461048 | | 1610913217 | | 18.60 | | 61.00 | | 42.07 | | 48.79 | | 1 | |
| 73 | | 181392743461048 | | 1610913337 | | 18.50 | | 61.00 | | 42.55 | | 51.79 | | 1 | |
| 74 | | 181392743461048 | | 1610913457 | | 18.60 | | 61.00 | | 42.94 | | 48.64 | | 1 | |
| 75 | | 181392743461048 | | 1610913577 | | 18.50 | | 61.00 | | 43.52 | | 51.43 | | 1 | |
| 76 | | 181392743461048 | | 1610913697 | | 18.50 | | 61.00 | | 43.81 | | 48.64 | | 1 | |
| 77 | | 181392743461048 | | 1610913817 | | 18.60 | | 61.00 | | 44.07 | | 51.82 | | 1 | |
| 78 | | 181392743461048 | | 1610913937 | | 18.60 | | 61.00 | | 44.36 | | 50.21 | | 1 | |
| 79 | | 181392743461048 | | 1610914057 | | 18.60 | | 61.00 | | 44.94 | | 51.38 | | 1 | |
| 80 | | 181392743461048 | | 1610914230 | | 18.60 | | 61.00 | | 45.27 | | 51.28 | | 0 | |
| 81 | | 181392743461048 | | 1610914351 | | 18.60 | | 60.00 | | 45.56 | | 50.55 | | 0 | |
| 82 | | 181392743461048 | | 1610917722 | | 17.90 | | 62.00 | | 41.68 | | 84.98 | | 0 | |
| 83 | | 181392743461048 | | 1610917842 | | 18.20 | | 62.00 | | 41.00 | | 45.13 | | 1 | |
| 84 | | 181392743461048 | | 1610917962 | | 18.30 | | 60.00 | | 42.00 | | 44.13 | | 1 | |
| 85 | | 181392743461048 | | 1610918082 | | 18.40 | | 61.00 | | 45.40 | | 45.96 | | 0 | |

Figura 37: Tabela *History*.

5. Conclusão

Ao concluir este projeto, o grupo está satisfeito com o resultado final, tendo em conta que foi possível pôr em prática e aprofundar os conceitos lecionados na UC. De notar que este trabalho se revelou mais extenso do que o grupo esperava, o que impossibilitou a realização de um trabalho ainda mais completo. De qualquer das formas, foram cumpridos os objetivos a que nos propusemos.

A nossa aplicação é de simples utilização e contém todos os serviços necessários para controlar o ambiente de cultivo. A aplicação *Web* é apelativa, leve e *user friendly*.

Estamos satisfeitos com o nosso projeto porque é um projeto no qual podemos continuar a trabalhar depois da avaliação do professor e do fim da disciplina. Acima de tudo, é um projeto atual e com futuro, considerando que cada vez mais se dá importância a questões ambientais e cada vez se dá mais valor a produtos caseiros e biológicos.

6. Referências

- [1] **How to Grow Plants From Seeds Step by Step.** [online] Disponível em: <https://www.natria.com/learn/rose-flower/10-steps-growing-plants-seed> [Acedido em 29 Outubro 2020].
- [2] **Usinainfo. 2020. ESP32 Nodemcu Placa De Desenvolvimento Iot ESP-32S - Usinainfo.** [online] Disponível em: <https://www.usinainfo.com.br/nodemcu/esp32-nodemcu-iot-com-wifi-e-bluetooth-30-pinos-5147.html> [Acedido em 14 Novembro 2020].
- [3] **Blogmasterwalkershop.com.br. 2020.** [online] Disponível em: https://blogmasterwalkershop.com.br/wp-content/uploads/2017/05/img02_conhecendo_o_nodemcu-32s_esp32_esp-32_wifi_bluetooth_esp8266_arduino_esp-wroom-32.jpg [Acedido em 14 Novembro de 2020].
- [4] **Vaisala.com. 2020.** [online] Disponível em: <https://www.vaisala.com/sites/default/files/inline-images/capacitive-humidity-sensor.jpg> [Acedido em 14 Novembro 2020].
- [5] **Last Minute Engineers. 2020. Insight Into How DHT11 DHT22 Sensor Works Interface It With Arduino.** [online] Disponível em: <https://lastminuteengineers.com/dht11-dht22-arduino-tutorial/> [Acedido em 14 Novembro 2020].
- [6] **Lastminuteengineers.com. 2020.** [online] Disponível em: <https://lastminuteengineers.com/wp-content/uploads/arduino/NTC-Thermistor-Temperature-Resistance-Characteristic-Curve.png> [Acedido em 14 Novembro 2020].
- [7] **Lastminuteengineers.com. 2020.** [online] Disponível em: <https://lastminuteengineers.com/wp-content/uploads/arduino/NTC-Thermistor-Temperature-Resistance-Characteristic-Curve.png> [Accessed 14 November 2020].
- [8] **Lastminuteengineers.com. 2020.** [online] Disponível em: <https://lastminuteengineers.com/wp-content/uploads/arduino/NTC-Thermistor-Temperature-Resistance-Characteristic-Curve.png> [Acedido em 14 Novembro 2020].
- [9] **ArduinoModulesInfo. 2020. KY-018 Photoresistor Module - Arduinomodulesinfo.** [online] Disponível em: <https://arduinomodules.info/ky-018-photoresistor-module/> [Acedido em 14 Novembro 2020].

- [10] **Images-na.ssl-images-amazon.com. 2020. [online]** Disponível em: https://images-na.ssl-images-amazon.com/images/I/61U81clfr3L._SL1500_.jpg [Acedido em 14 Novembro 2020].
- [11] **2020. [online]** Disponível em: <https://www.digikey.com/catalog/en/partgroup/submersible-3v-dc-water-pump-horizontal-or-vertical-type/107104> [Acedido em 14 Novembro 2020].
- [12] **5.imimg.com. 2020. [online]** Disponível em: <https://5.imimg.com/data5/KQ/RZ/XM/SELLER-65880656/dc-3-6v-submersible-pump-mini-water-pump-500x500.jpg> [Acedido em 14 Novembro 2020].
- [13] **Jameco.com. 2020. Servo Motors Work | How Servo Motors Work. [online]** <https://www.jameco.com/jameco/workshop/howitworks/how-servo-motors-work.html> [Acedido em 14 Novembro 2020].
- [14] **Lastminuteengineers.com. 2020. [online]** Disponível em: <https://lastminuteengineers.com/wp-content/uploads/arduino/NTC-Thermistor-Temperature-Resistance-Characteristic-Curve.png> [Acedido em 14 Novembro 2020].
- [15] **Lastminuteengineers.com. 2020. [online]** Disponível em: <https://lastminuteengineers.com/wp-content/uploads/arduino/NTC-Thermistor-Temperature-Resistance-Characteristic-Curve.png> [Acedido em 14 Novembro 2020].
- [16] **Randomnerdtutorials.com. 2020. [online]** Disponível em: <https://randomnerdtutorials.com/what-is-mqtt-and-how-it-works/>
- [17] **Opc-router.com. 2020. [online]** Disponível em: <https://www.opc-router.com/what-is-mqtt/>
- [18] **Docente de Microdispositivos de RF para comunicação sem Fios, "Trabaho Prático",** Universidade do Minho, 2020