UNIVERSIDADE FEDERAL DO AMAZONAS
FACULDADE DE TECNOLOGIA
COORDENAÇÃO DE GRADUAÇÃO EM ENGENHARIA ELÉTRICA

IMPLEMENTAÇÃO E MONITORAMENTO DE UM SISTEMA DE IRRIGAÇÃO AUTOMATIZADO EM IOT UTILIZANDO MÓDULO ESP32 EM PLANTIO CASEIRO

Luiz Felipe Araújo Henriques

Monografia de Graduação apresentada a Coordenação de Graduação em Engenharia Elétrica, da Universidade Federal do Amazonas, como parte dos requisitos necessários à obtenção do título de Engenheiro em Engenharia Elétrica.

Orientador: Waldir Sabino da Silva Júnior

Manaus

Dezembro de 2021

Ficha Catalográfica

Ficha catalográfica elaborada automaticamente de acordo com os dados fornecidos pelo(a) autor(a).

Henriques, Luiz Felipe Araújo
H519i Implementação e monitoramento de um sistema de irrigação automatizado em IoT utilizando módulo ESP32 em plantio caseiro / Luiz Felipe Araújo Henriques . 2021
70 f.: il. color; 31 cm.

Orientador: Waldir da Silva Júnior
TCC de Graduação (Engenharia Elétrica - Telecomunicações) - Universidade Federal do Amazonas.

1. Protocolo MQTT. 2. IOT - Internet of Things. 3. Dashboard. 4. Sistema de Irrigação. I. Silva Júnior, Waldir da. II. Universidade Federal do Amazonas III. Título

IMPLEMENTAÇÃO E MONITORAMENTO DE UM SISTEMA DE IRRIGAÇÃO AUTOMATIZADO EM IOT UTILIZANDO MÓDULO ESP32 EM PLANTIO CASEIRO

Luiz Felipe Araújo Henriques

MONOGRAFIA SUBMETIDA AO CORPO DOCENTE DA COORDENAÇÃO DE GRADUAÇÃO EM ENGENHARIA ELÉTRICA DA UNIVERSIDADE FEDERAL DO AMAZONAS COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE ENGENHEIRO EM ENGENHARIA ELÉTRICA.

Aprovada por:

Prof. Waldir Sabino da Silva Júnior, D.Sc.

Prof. Carlos Augusto de Moares Cruz, D.Sc.

Prof. Diego Alves Amoedo, M.Sc.

MANAUS, AM – BRASIL

DEZEMBRO DE 2021

Agradecimentos

Quero agradecer primeiramente a Deus. Agradecer minha mãe, Lilian Fernandes, por sempre ser uma mãe dedicada, me apoiar nas minhas escolhas e por sempre sacrificar-se para me dar uma boa educação. Agradecer ao meu pai Antônio Moreira pela ajuda e incentivo aos estudos.

Agradecer aos amigos e colegas da UFAM pelo companheirismo e amizade ao longo da graduação em especial, Abdel Fadyl, Adelson Auzier, Alerrandro Souza, Ana Jéssica Ferreira, Felipe Santos, kevin Guimarães, Leopoldino Lourenço, Lucas Tribuzy, Rafael Furtado, Sandro Matos, Yasmim Brito, Yasmim Torres, Jones Castro Pinto.

Em especial agradeço ao Prof. Waldir Sabino por todos os ensinamentos, conselhos, orientações e suporte ao longo da graduação. Agradeço finalmente a toda a equipe de professores da UFAM que ajudaram a sedimentar essa formação.

Grande é o Senhor e muito digno de ser louvado; sua grandeza não tem limites.

Salmo 145:3

Resumo da Monografia apresentada à UFAM como parte dos requisitos necessários para a obtenção do grau de Engenheiro em Engenharia Elétrica

IMPLEMENTAÇÃO E MONITORAMENTO DE UM SISTEMA DE IRRIGAÇÃO AUTOMATIZADO EM IOT UTILIZANDO MÓDULO ESP32 EM PLANTIO CASEIRO

Luiz Felipe Araújo Henriques

Orientador: Waldir Sabino da Silva Júnior

Coordenação de Graduação em Engenharia Elétrica

Este trabalho consiste no desenvolvimento e implementação de um sistema de irrigação automatizada e fundamentada nos conceitos de Internet das coisas. O projeto proposto foi realizado através do uso do protocolo de comunicação entre máquinas MQTT e módulo microcontrolador Esp-wroom-32, para o monitoramento de dados de umidade do solo e temperatura ambiente coletados por sensores em plantio caseiro, transmitidos a plataforma IoT TagoIO para a exibições dos dados recebidos por meios de gráficos em *dashboard*.

Palavras-chave: Protocolo MQTT, IoT, Dashboard, Sistema de irrigação.

Abstract of Monograph presented to UFAM as a partial fulfillment of the requirements for the degree of Engineer in Eletrical Engineering

IMPLEMENTATION AND MONITORING OF AN IOT AUTOMATED IRRIGATION SYSTEM USING ESP32 MODULE IN HOME PLANTING

Luiz Felipe Araújo Henriques

Advisor: Waldir Sabino da Silva Júnior

Electrical Engineering Undergraduate

This work consists of the development and implementation of an automated irrigation system based on the concepts of the Internet of Things. The proposed project was carried out using the communication protocol between MQTT machines and the ESP-WROOM-32 microcontroller module, for monitoring soil moisture and ambient temperature data collected by sensors in home planting, transmitted to the TagoIO IoT platform for the views of data received through dashboard graphics.

Keywords: MQTT protocol, IoT, Dashboard, Irrigation system.

Sumário

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução

As hortaliças têm o seu desenvolvimento, bem como, seu rendimento drásticamente influenciados pelas condições de clima e de umidade do solo. Os agricultores encaram várias adversidades com questões relacionadas a falta da chuva por um tempo prolongado, pois a deficiência de água no solo é o fator mais relevante para a qualidade da colheita e até mesmo evitar perda de todo cultivo. Por outro lado é importante ressaltar que até mesmo o excesso de água pode ter um efeito nocivo para determinadas culturas de hortaliças, por isso os recursos tecnológicos têm sido de suma importância para suprir as necessidades e evitar o desperdício de água.

Com as evoluções tecnológicas, não só no que se refere a microeletrônica, mas também a revolução da computação e as tecnologias de comunicação digitais em geral, unidas com o objetivo de se interconectar aos mais diversos tipos de objetos e sistemas eletrônicos e mesmo os computacionais, um conceito muito importante e que vem ganhando espaço é o da Internet das coisas (IoT, do inglês, *Internet of things*). A tecnologia possui várias aplicações importantes, incluindo monitoramento de tráfego, chaves eletrônicas na hotelaria, abastecimento de água e etc.

A Internet das Coisas é uma das alternativas para fazer o controle automatizado de vários sistemas, inclusive de sistemas de irrigação. A comunicação *wireless* pode ser feita por meio de um protocolo, pois as implementações já existem e um dos protocolos mais conhecidos para troca de mensagens em IoT é o protocolo de transporte de filas de mensagens de telemetria (MQTT, do inglês, *message queue telemetry transport)*, que é um protocolo utilizado para a comunicação entre máquinas. No trabalho proposto o processo de implementação irá envolver desde coleta e

transmissão remota de dados em tempo real por meio de tecnologias de comunicação sem fio, armazenamento em nuvem e os dados captados poderão se observados e analisados no *dashboard* da TagoIO, mas simultaneamente ao envio de dados o sistema irá irrigar a horta caso seja coletado pelo sensor higrômetro um valor de umidade do solo abaixo de 50%.

1.1 Objetivo Geral

Aplicar o conceito de IoT utilizando o protocolo MQTT para fazer a comunicação do monitoramento remoto de temperatura e umidade de uma horta e assim irrigá-la se a umidade do solo estiver abaixo de 50%.

1.2 Objetivos Específicos

- Desenvolver um sistema de irrigação automatizado utilizando módulo ESP-WROOM-32, válvula solenoide, sensor DHT11, sensor higrômetro e módulo relé;
- Obter valores de temperatura ambiente e umidade do solo via monitor serial do ambiente de desenvolvimento integrado do ESP32;
- Aplicar técnicas do protocolo MQTT para envio de dados coletados para o *dashboard* TagoIO;
- Realizar testes de variação de temperatura, umidade do solo em terra seca, terra úmida e exibir os respectivos dados em gráficos no painel *dashboard* TagoIO;
- Acionar a irrigação do plantio em condição de umidade do solo menos que 50%;

1.3 Organização do Trabalho

A monografia está organizada da seguinte maneira:

- No Capítulo 2, são apresentados os fundamentos teóricos utilizados na Monografia. Primeiramente, conceitos relacionados à IoT e a comunicação via protocolo MQTT. Posteriormente, uma breve apresentação do módulo ESP-WROOM-32, com seus recursos e sua pinagem. É abordado também a configuração do sistema de irrigação com uma explanação acerca dos sensores de tempertura e umidade bem como o Dahsboard utilizado para a análise dos dados;

- O Capítulo 3 apresenta o desenvolvimento do trabalho, contém a metodologia do desenvolvimento do sistema de irrigação;

- O Capítulo 4 apresenta os experimentos e a análise dos resultados de desempenho do sistema;

- O Capítulo 5 apresenta as conclusões deste trabalho bem como sugestões para trabalhos futuros.

Capítulo 2

Fundamentação Teórica

2.1 Internet of Things

2.1.1 Conceito

Conforme o autor Kiran, *et al.*, a internet das coisas (IoT, do inglês, *Internet of things)*, pode ser compreendida como uma infraestrutura global que permite serviços avançados ao interconectar dispositivos físicos e virtuais, com base nas tecnologias de informação e comunicação interoperáveis [3]. Um determinado tipo de rede utilizada para conectar dispositivos a internet por meio de protocolos específicos capazes de reunir e de transmitir dados a depender da aplicação dos mais variados equipamentos de sensoriamento, para que possa ocorrer a troca de informações, com o intuito de estabelecer reconhecimento inteligente, sistema de localização em tempo real e rastreamento, por exemplo. Com uma infinidade de aplicações envolvento IoT, pode- se compreender vários pontos de vista a cerca da definição desta tecnologia.

Conforme o banco nacional de desenvolvimento econômico e social (BNDES), é um conjunto de elementos estruturais que habilitam serviços avançados por meio da interconexão entre componentes físicos e virtuais, com base nas tecnologias de informação e comunicação. Em sentido amplo, trata-se não apenas de conectar coisas, como veículos e eletrodomésticos, mas também de dotá-las do poder de processar dados, tornando-as inteligentes. Por isso, a Internet das coisas vem ganhando espaço não somente pelo surgimento de tecnologias disruptivas, mas também pela evolução de um conjunto de tecnologias já disponíveis, que estão se tornando mais acessíveis,

possibilitando sua adoção em massa [4].

A possibilidade de uma fluidez entre dispositivos (sensores, atuadores por exemplo), com as informações sem qualquer intervenção humana, ou seja, sistema automatizado em seu maior grau, é um próposito importante e até mesmo central dentro do conceito de Internet das coisas. Os dispositivos físicos são considerados apenas objetos na estrutua, ou seja, os *hardwares*, que são controlados por meio de codificações nomeadas de softwares, que são de origem virtual. A ideia para o surgimento do conceito de internet das coisas, deve-se a algumas áreas da engenharia, como: Microeletrônica, sistemas embarcados e sensoriamento remoto por exemplo. A interconexão entre os objetos e a internet está avançando na mesma medida que a capacidade computacional. Qualquer dispositivo em conexão com a internet pode ser controlado de forma a criar conexões e troca de dados entre provedores de serviços [5].

2.1.2 Pré-requisitos de uma Aplicação em IoT

Para que haja um sistema baseado em Internet das coisas, se faz necessário utilização de dispositivos conectados em uma rede comum, e os mesmos devem fazer uso de tecnologias (protocolos por exemplo) para que a comunicação entre si seja possível. É fundamental um mapeamento dos objetivos a serem executados por esse sistema e quais são as saídas de interesse. Os pré-requisitos irão depender de cada sistema, porém é possível estabelecer três pré-requisitos básicos comuns a todo sistema baseado em IoT [4].

1 º) Pré-requisito: Recebimento de dados digitais vindo de sensores e/ou indo para atuadores (como exemplo, sensor aferindo temperatura em um motor);

2 º) Pré-requisito: Conexão com uma rede;

3 º) Pré-requisito: Habilidade de realizar o processamento dados de forma automática (sem intervenção humana).

Conforme a autora Santos, *et al.*, existe uma estrutura com definições dos mecanismos associados no processo de apliçações de Internet das coisas, que possibilitam a interação dos dispositivos físicos com o ambiente virtual, são [5]:

a) **Reconhecimento:** Tem um papel fundamental para realizar a identificação única dos dispositivos que serão conectados a Internet. Algumas tecnologias

são amplamente empregas para tal necessidade, como: identificação por rádio frequência (RFID, do inglês, *radio frequency identification*), que realiza a identificação dos dispositivos através de ondas de rádio e a comunicação por campo de proximidade (NFC, do inglês, *near field communication*), que é uma tecnologia que faz a identificação de dispositivos, possibilitando a troca de dados entres estes, desde que sejam compatíveis com a tecnologia NFC.

b) **Sensores/Atuadores:** Estes dispositivos interagem com o ambiente recebendo informações desde e transmitindo os dados coletados, no cenário de internet das coisas esses dados podem ser transmitidos a um determinado banco de dados ou até mesmo para os diversos serviços em nuvem existentes.

c) **Comunicabilidade:** Refere - se as diversas técnicas e tecnologias empregadas para conexão dos dispositivos a serem utilizados na aplicação IoT, e até mesmo interferindo no consumo de energia destes, sendo considerados muito importantes por tal aspecto. As tecnologias amplamente empregadas são RFID, Wi-Fi, *Bluetooth*. Alguns fatores importantes para a comunicação desses dispositivos com a Internet, sendo necessários a utilização de protocolos de comunicação específicos para que a rede dos dispositivos atenda os pré-requisitos dos padrões da rede global.

d) **Processamento:** Refere-se a ao sistema de operação dos dispositivos que farão o processamento das informaçõees, tais como: microcontroladores (como por exemplo, arduino e ESP32) e processadores em geral, que executam os códigos ou algomritmos nesses dispositivos ditos inteligentes.

e) **Serviços:** Para o fornecimentos dos serviços em IoT, algumas áreas de destaque são de suma importância, que são: Reconhecimento, integração das informações, cooperação inteligente. O reconhecimento tem o papel de realizar o mapeamento dos dispositivos físicos de relevância para o usuário fazendo sua correspondência em ambientes virtuais, como por exemplo temperatura do ambiente físico com o seu valor, as coordenadas de localização desse sensor e circunstância da coleta de dados. Tendo assim a tarefa de coletar e agrupar esses dados.

A integração das informações é encarregada do tratamento dos dados e agregar esses dados coletados pelos dispositivos inteligentes a classificações de dados homogêneos e/ou heterogêneos.

e) **Semântica:** Tem a tarefa de retirar as informações dos dispositivos IoT, para poder definir os serviços que podem ser realizados com os dados obtidos. São utilizadas várias ténicas para essa finalidade, como: ou modo de representação e de descrição de recursos (RDF, do inglês, *resource description framework)*, ou no formato binário para troca de dados utilizando *efficient XML interchange*.

2.1.3 Arquitetura Geral em Sistemas IoT

Um fator de suma importância no que diz respeito às aplicações de sistemas IoT, é a arquitetura desse sistema, a maneira como os dispositivos desse sistema estão dispostos e como se comunicam. Dado a complexidade desse sistema uma análise das camadas pode ser útil para um melhor entendimento de todas as partes envolvidas. Diante de uma infinidade de soluções IoT, plataformas *cloud* IoT, arquiteturas e etc. Diversos cientistas e estudiosos da área, têm se empenhado em definir uma arquitura geral em IoT. Abaixo um exemplo de uma possível arquitetura IoT de conforme os conceitos dos autores Al-Fuqahaet, *et al.*, e Khan, *et al.*, que sugeriram uma divisão da estrutura IoT em 5 camadas [6] [7]:

- **Camada de Percepção:** Essa é a primeira camada e a mesma integra toda a parte física do sistema IoT, tais como: sensores e atuadores por exemplo. O principal objetivo dessa primeira camada é a coleta de dados por meio de sensores, os dados podem ser de umidade, localização, temperatura e etc. Os dados coletados são transmitidos para a segunda camada (camada de rede).

- **Camada de Rede:** Esta camda transmite as informações dos dispositivos (sensores) para o sistema que irá processará estas informações. tecnologia utilizada pode ser pode ser por exemplo Wi-Fi, utilizada em IoT. A tecnologia escolhida para o envio deverá ser compatível com o dispositivo. Dessa forma, esta camada transfere as informações da camada de percepção para a próxima camada (camada de *middleware*).

- **Camada de Middleware:** A finalidade dessa camada é gerenciar os serviços e estabelecer a conexão com determinada base de dados. As informações recebidas da camada de rede, são salvas na base de dados. A camada de *middleware* efetua o processamento das informações e toma decisões de acordo com as respostas do processamento.

- **Camada de Aplicação:** Entrega os serviços requisitados pelos usuários. A importância dessa camada para a IoT é que ela possui a capacidade de fornecer serviços inteligentes, com qualidade, com o objetivo de suprir as necessidades dos usuários.

- **Camada de Negócio:** Essa camada faz o gerenciamento dos serviços de um sistema IoT, a mesma gerencia os dados recebidos da camada de aplicação e viabiliza um suporte nos processos de tomada de decisão fundamentada em análise de *big data*. O gerenciamento das camadas inferiores é realizado nesta.

2.1.4 Aplicações em IoT

Com as constantes inovações tecnologicas, principalmente em conectividade e o desenvolvimento em processos envolvendo internet das coisas, existe a possibilidade das aplicações IoT se disseminarem por quase todos os setores da economia. Os elementos chaves para aceitação e utilização do conceito de IoT pela massa, que irá favorecer a economia e sociedade Brasileira, são: dispositivos eletrônicos mais eficientes e com maior velocidade de processamento e resposta, avanço no armazenamento e processamento de dados, difusão de redes de telecomunicações [8]. Com a Internet das coisas, as empresas podem fazer uma análise do funcionamento dos seus sistemas implantados em tempo real, podem coletar dados e automatizar as repostas de acordo com a saída, o que pode possibilitar melhorias interessantes na qualidade do serviço e auxilia na tomada de decisão por parte da empresa. Abaixo alguns exemplos de aplicações do IoT [9]:

- Monitoramento de tráfego de veículos;

- Gestão de frotas;

- Fazendas inteligentes;

- Controle de irrigação na agricultura;
- Chaves eletrônicas na hotelaria.

2.2 Comunicações *Wireless*

2.2.1 Conceito de *Wireless*

A tecnologia capaz de transmistir dados sem a necessidade de cabeamento, ou seja, sem fio (*Wireless*, do inglês), viabiliza a transmissão de dados em conexão entre pontos distantes sem precisar usar fios [10]. No wireless a transmissão das informações acontece por meio do ar via ondas de rádio,que se irradiam e se propagam pelo espaço [11]. O *wireless* envolve vários outras tecnologias, como: Wi-Fi, IrDA, *Bluetooth* entre outras.

2.2.2 *Wireless-Fidelity (Wi-Fi)*

É corriqueira a ideia de que wireless e Wi-Fi são a mesma tecnologia, porém, conforme os autores Engst e Fleishman, *et al.*, o Wi-Fi tem sua origem baseada no padrão 802.11, que foi um padrão definido pelo instituto de engenheiros eletricistas e eletrônicos (IEEE, do inglês, *institute of electrical and electronics engineers*), o padrão referido foi homologado ainda nos anos 90. A conexão via Wi-Fi surgiria a partir de uma conexão vi cabos juntamente com um transmissor que irá emitir o sinal de internet por meio de ondas eletromagnéticas via ar [12].

Conforme o autor Mendes, *et al.*, o padrão 802.11 define uma série de especificações criadas pelo IEEE para redes *wireless*, o objetivo do instituto IEEE é padronizar os equipamentos que utilizam a tecnologia *wireless* para que esses equipamentos possam sempre ter compatibilidade. A especificação do padrão 802.11 foi aceita em 1997, e define uma interface entre um computador sem fio e o seu ponto de acesso, e entre dois computadores sem fio. O padrão 802.11 define a camada de controle de acesso ao meio (MAC, do inglês, *Media access control*), para transmissões de dados em redes sem fio.

O Wi-Fi faz a interface da camada física/ enlace, que são as primeiras camadas do modelo de interconexão de sistemas abertos (OSI, do inglês, *open systems*

interconnection). Conforme o autor Mendes, *et al.*, essas camadas teriam como intuito a redução do nível de complexidade das redes, sendo atribuída a cada uma dessas camadas serviços específicos e complementares às funções umas das outras de modo a estarem interconectadas. Dessa forma, foi criado na década de 1970, pela organização internacional de normalização (ISO, do inglês, *(international standards organization*) o padrão OSI, que contém sete camadas interligadas: física, enlace, rede, transporte, sessão, apresentação e aplicação [13].

2.2.3 O que são Protocolos de Rede?

Os protocolos de redes podem ser compreendidos como diretrizes para que vários dispositivos conectados à internet consigam realizar a comunicação entre si. Esses protocolos são responsáveis por transmitir as informações por meio de pacotes, cada capote transporta consigo informações da origem da informação e o destino da mesma [14].

2.2.4 Camadas do Modelo OSI

O modelo de Interconexão de Sistemas Abertos (OSI, do inglês, *open systems interconnection*), tem como principal função padronizar e definir uma estrutura para os protocolos de comunicação utilizados entre os mais diversificados tipos de sistema, possibilitando uma comunicação *end-to-end*. Com o modelo OSI é possível viabilizar a comunicação entre máquinas distintas e definir instruções gerais para a elaboração de redes de computadores independente da tecnologia empregada, sejam redes de curta, média ou longa distância. A arquitetura deste modelo é divido a rede de computadores em 7 camadas interligadas [13]:

1 º) camada: Física;

2 º) camada: Dados;

3 º) camada: Rede;

4 º) camada: Transporte;

5 º) camada: Sessão;

6 º) camada: Apresentação;

7 º) camada: Aplicação.

2.2.5 Aplicações dos Principais Protocolos de Rede

Para que haja uma interação de comunicação entre computadores ou computador e dispositivo, o pré-requisito é que os mesmos devem possuir configurações no mesmos parâmetros e que utilizem iguais padrões de comunicação. A rede de computadores é divida em camadas, onde cada camada realiza uma função bem definida. Os mais variados protocolos de rede são usados conforme o tipo de serviço a ser implementado e de sua camada correspondente. Abaixo alguns dos principais protocolos em suas respectivas camadas:

- camada física: Ethernet, FDDi, Modem;
- camada de rede: IPv4, IPv6;
- camada de transporte: TCP, DCCP, UDP, RTP;
- camada de aplicação: WWW, HTTP, FTP , Telnet, SMTP, DNS, PING

2.3 Protocolo MQTT

Um sistema que utiliza o protocolo MQTT, é voltado para comunicação entre máquinas (M2M, do inglês, *machine-to-machine).* Esse protocolo realiza a comunicação *publish/subscribe*, com taxas de transmissão que possibilitam a utilização até de hardwares com baixo poder de processamento, porém com um relativo grau de entrega das informações e até mesmo da confiabilidade. Tais características tornam o protocolo MQTT, uma das melhores alternativas para projetos que envolvem IoT. O protocolo é amplamente utilizado por ser leve e exigir pouco dos hardwares, por ter baixo consumo de dados e também por ter uma comunicação bilateral, ou seja, do cliente para o servidor e vice-versa [15].

2.3.1 Conceito de MQTT

Consiste em um protocolo de transporte de mensagens de publicação/assinatura do *Client Server*. É leve, aberto, simples e projetado para ser fácil de implementar [1]. Na década de 90 a IBM, desenvolveu um protocolo de mensagens conhecido como transporte de filas de mensagem de telemetria (MQTT, do inglês,

message queuing telemetry transpor). O MQTT surgiu da necessidade de desenvolver um protocolo simples e leve que realizasse a comunicação entre máquinas. Hoje o protocolo é padronizado pela organização para o avanço de padrões de informação (OASIS, do inglês, *organization for the advancement of structured information standards*), é um dos protocolos mais utilizados em Internet das coisas (IoT), pois pode ser usado em uma infinidade de aplicações e nos diversos segmentos da indústria, como: Automotiva, manufatura, telecomunicações por exemplo [16].

2.3.2 Definição de Broker MQTT

É definido como o servidor intermediário *(broker, do inglês)*, na comunicação entre os dispositivos MQTT, gerenciando informações recebidas dos clientes que publicam dados (sensores por exemplo) e transmitindo aos clientes que requisitam esses dados [17].

2.3.3 Modelo Publicação/Assinatura e Tópicos em MQTT

O modelo *publish/subscribe* ou publicação/assinatura, separa o cliente que publica a mensagem do cliente que requisita a mensagem. Os clientes não precisam ter uma comunicação direta e além disso o modelo pode ter inúmeros clientes que publicam ou recebem informações. Às informações são enviadas por meio de tópicos [18] que podem ser entendidos como endereço para qual uma mensagem será direcionada, um cliente que publica, cria este endereço e o cliente que assina se inscreve naquele endereço para requisitar as informações, um assinante poderá se inscrever em vários tópicos simultâneos, por outro lado um cliente que publica as informações poderá criar diversos tópicos e Isso é possível por conta de um servidor intermediário conhecido como *broker*, que é um servidor intermediário, que recebe os dados e transmite para os assinantes de maneira organizada, pois o mesmo classifica os dados quando os coloca em tópicos correspondentes [19].

Um exemplo seria a organização de classe, o tópico pressao_ar_data publicado por um sensor que afere a pressão atmosférica, irá conter apenas dados referentes a pressão do ar e tópico de umidade_data publicado por um sensor de umidade, apenas dados de umidade. Os dados serão requisitados por um cliente *subscriber* (um app ou *dashboard*, por exemplo) que assinou ambos os tópicos.

2.3.4 Transmissão de Dados MQTT

Conforme a Organização para o Avanço de Padrões de Informação Estruturada (OASIS), o protocolo de comunicação entre máquinas MQTT, opera realizando a troca de uma série de pacotes de controle de maneira definida. Dentre as partes que compõem a estrutura dos pacotes de controle, uma que merece especial atenção é o cabeçalho fixo, pois está presente em todos os pacotes de controle MQTT [1] .

Cabeçalho Fixo de um Pacote em MQTT								
bits	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
1o. Byte	Tipo de mensagem (escrito com 4 bits)				DUP(Entrega duplicada)	Nível QOS (Qualidade do serviço)		Retenha
2o. Byte	Representa o número de bytes restantes na mensagem atual							

Tabela 2.1: Formato de um cabeçalho fixo de um pacote em MQTT [1].

No cabeçalho fixo o primeiro byte sempre tem 4 bits para qual o tipo de pacote de controle deve ser enviado ou recebido pelo cliente MQTT, e o *broker*, 1 bit para sinalizar o reenvio de um pacote pelo cliente MQTT (entrega duplicada), se tiver a necessidade de reenvio, 2 bits para determinar o nível de qualidade do serviço, 1 bit para o *broker* guardar a última informação para que o cliente que assinou o tópico após o envio da mensagem possa visualizá- la.

O segundo byte irá informar o tamanho do cabeçalho variável se houver e o *payload*, que seria a informação pura. No campo do comprimento restante pode ter de 1 a 4 bytes, o mesmo utiliza codificação binária de comprimento variável, mais conhecida como UTF-8, técnica que possibilita a transmissão de mensagens de até 256 bits [1] [20].

Na Tabela 2.2 há uma relação de todas as mensagens possíveis do protocolo MQTT, mensagens de comunicação entre clientes MQTT e servidor *broker* MQTT. cada mensagem está associada à respectiva direção do fluxo de dados correspondentes e ainda uma definição de função dessas mensagens.

Mensagens características de um pacote de controle em MQTT			
Comunicação	**Valor decimal**	**Direção do fluxo de informação**	**Descrição de estado**
RESERVADO	0	Proibido	Reservado
CONNECT	1	Cliente-Broker	Cliente solicita conexão com ao Broker
CONNACT	2	Broker-Cliente	Reconhecimento de conexão
PUBLISH	3	Bidirecional	Publicar mensagem
PUBACK	4	Bidirecional	Reconhecimento de publicação
PUBREC	5	Bidirecional	Publicação recebida
PUBREL	6	Bidirecional	Publicação lançada
PUBCOMP	7	Bidirecional	Publicação completa
SUBSCRIBE	8	Cliente-Broker	Cliente solicita assinatura
SUBACK	9	Broker-Cliente	Reconhecimento de assinatura
UNSUBSCRIBE	10	Cliente-Broker	Pedido de cancelamento de assinatura
UNSUBACK	11	Broker-Cliente	Reconhecimento de cancelamento de ass.
PINGREQ	12	Cliente-Broker	Pedido de PING
PINGRESP	13	Broker-Cliente	Resposta de PING
DISCONNECT	14	Cliente-Broker	Cliente está desconectado
RESERVADO	15	Broker-Cliente	Reservado

Tabela 2.2: Tipos de mensagens de um pacote de controle em MQTT [1].

2.3.5 Definição de Qualidade do Serviço

Dentre inúmeras perspectivas existentes para uma definição de qualidade do serviço (QoS, do inglês, *quality of service*), a que é mais interessante para o campo da tecnologia é a vinda da ótica das telecomunicações, que de acordo com E.800 (ITU, *et al.*), sendo o resultado da união e desempenho do serviço que define o nível de satisfação do usuário deste serviço [21].

2.3.6 Qualidade de Serviço em MQTT

As conexões em geral determinam um nível de QoS, que basicamente descreve como será a qualidade do serviço entre elementos de uma conexão [22]. Em MQTT, o nível de qualidade de serviço pode ser comparado a um acordo entre o cliente que enviará a informação *(Publisher)* e o *broker* MQTT. O objetivo é garantir a entrega da mensagem. Há a possibilidade de três níveis, que são: QoS 0, QoS 1, QoS 2.

a) **QoS 0 - No máximo uma vez:** Este nível de serviço garante uma entrega conhecida como melhor esforço. Nela não há um *feedback* de entrega da mensagem por parte do *broker* MQTT, e além disso a mensagem não fica

armazenada ou até mesmo é transmitida novamente, ou seja, a mensagem se perde. Este nível é comumente denominado como dispare e esqueça, o nível de garantia é o mesmo do protocolo TCP.

b) **QoS 1 - Pelo menos uma vez:** Neste nível de serviço temos uma garantia de que uma mensagem será entregue pelo menos uma vez ao *broker* MQTT (receptor). O emissor da mensagem armazena a mensagem até ter como retorno um pacote de sinalização PUBACK do *broker* MQTT, que confirma o recebimento da mensagem. Uma única mensagem pode ser enviada diversas vezes até que o Broker indique o recebimento. O cliente MQTT que publica a mensagem, envia junto com a mensagem um identificador de pacote associado ao pacote correspondente de *feedback* do *broker*, pacote de garantia de recebimento da mensagem, mais conhecido como Pacote PUBACK. Caso o emissor da mensagem não receba um pacote informando o recebimento da mensagem (PUBACK),em um tempo determinado, o emissor enviará novamente a publicação e quando o emissor da mensagem reenviar a mensagem, o mesmo utilizará uma sinalização conhecida como DUP, apenas para informar o *broker*, mas de qualquer forma o *broker* irá enviar um pacote PUBACK.

c) **QoS 2 - Exatamente uma vez:** O serviço de QoS 2, garante que toda mensagem seja recebida apenas uma vez pelo *broker* MQTT. Este nível tem a maior confiabilidade, porém mais lento que os outros níveis. O nível de confiabilidade é feito em basicamente dois fluxos de relação solicitação e resposta, feito entre cliente emissor e *broker*. Ambos utilizam um identificador de pacote da mensagem original que foi enviada para administrar a entrega desta mensagem. Neste serviço o *broker* MQTT recebe a mensagem enviada pelo cliente *publisher* e retorna a este um pacote conhecido como PUBREC, que reconhece o pacote do tipo *publish*. Caso o cliente emissor da mensagem não tenha um retorno (receba o pacote PUBREC) do *broker*, o emissor reenviará o pacote *publish* com a sinalização DUP, até que o mesmo receba um *feedback* de confirmação. Após o recebimento do pacote PUBREC do *broker* MQTT, o emissor da mensagem poderá desprezar a mensagem original e além disso ele armazena o pacote PUBREC e responde a esse pacote com outro pacote conhecido como PUBREL. Quando o Broker recebe o pacote PUBREL,

despreza os pacotes armazenados e responde com um pacote conhecido como PUBCOMP e guarda a referência ao identificador da mensagem original. logo após esse processo o emissor recebe o pacote PUBCOMP, nessa etapa o pacote da mensagem original pode ser reutilizado e ao fim do processo o cliente MQTT *publisher* tem a confirmação da entrega da mensagem.

2.4 Módulo ESP-WROOM-32

2.4.1 Definição de ESP32

O ESP32 foi desenvolvido pela *espressif systems* e começou a circular no mercado em meados de 2016 [23] . O microcontrolador possui uma alta velocidade de processamento dos dados e conectividade via Wi-Fi, é na atualidade um dos controladores mais comercilizados até mesmo por sua capacidade de armazenamento, que é superior a outros controladores amplamente utilizados, como o arduino por exemplo [24].

O módulo ESP32 é um módulo de MCU que integra um microcontrolador da série ESP32, Wi-Fi e *Bluetooth*. Este microcontrolador é amplamente utilizado em projetos envolvendo Internet das coisas, pois é capaz interagir com as mais variadas bibliotecas e módulos (sensores), em uma infinidade de linguagens de programação, sendo a mais comum a linguagem C++. No caso do protocolo de comunicação MQTT, o ESP32 é um cliente MQTT,que utiliza a IDE do arduino como cliente pub/sub em MQTT. Pode-se destacar alguns aspectos positivos do dispositivo ESP32: Tem ótimo desempenho de potência, pouco consumo de energia, amplificador de baixo ruído. Abaixo algumas especificações do modelo ESP32-WROOM-32:

- CPU: Xtensa Dual-Core 32-bit LX6;
- ROM: 448 KBytes;
- RAM: 520 KBytes;
- Flash: 4 MB;
- Clock máximo: 240MHz;

- Wireless padrão 802.11 b/g/n;
- Bluetooth BLE 4.2.

As especificações de operação são:

Parâmetros	Símbolo	Mínimo	Ideal	Máximo	Unidade
Temperatura de Operação	-	- 40	20	85	°C
Tensão de Alimentação	VDD	2,2	3,3	3,6	V
Corrente de Operação	IvDD	0,5	-	-	A

Tabela 2.3: Especificações de Operação do ESP32-WROOM-32 [2].

Os periféricos do módulo ESP32-WROOM-32 possuem:

- 18 Pinos referentes a função de Conversor de analógico para digital (ADC);
- 3 Pinos de interfaces SPI;
- 3 Pinos de interfaces UART (comunicação serial);
- 2 Pinos de interfaces I2C;
- 16 Pinos referentes a canais de saída PWM(modulação por largura de pulso);
- 2 Pinos referentes a função de Conversores Digital-Analógico (DAC);
- 2 Pinos de interfaces I2S;
- 10 Pinos GPIOs com detecção capacitiva (Touch).

2.4.2 Ambiente de Desenvolvimento Integrado (IDE)

Ambiente de Desenvolvimento Integrado (IDE, do inglês, *Integrated Development Environment)*, é um *software* utilizado para desenvolver aplicações que reúne um conjunto ferramentas comuns de desenvolvimento em uma só interface gráfica do usuário (GUI). Um IDE geralmente constitui-se de [25]:

- Editor de código-fonte: é um editor de texto que contribui na desenvolvimento de um código de software viabilizando algumas funções, como por exemplo:

destaque da sintaxe com indicadores, ferramenta de preenchimento automático específico da linguagem e uma das mais importantes, a verificação e indicação de bugs (erros) durante o desenvolvimento do *software*.

- Automação de compilação local: Os compiladores que automatizam funções menos complexas e repetitivas ao longo da criação de uma compilação local do software usada pelo desenvolvedor. São funções, como: compilação de código-fonte em código binário, desenvolvimento de pacotes de código binário e ainda a realização dos testes automatizados.

- Debugger: É basicamente um software que possui a função de testar outros softwares e mostrar graficamente o local exato onde está ocorrendo o bug no código.

Os ambientes de desenvolvimento integrado foram criados para dar suporte aos desenvolvedores de *software*, organizando o fluxo de trabalho. Os IDEs verificam o código fonte no momento em que está sendo escrito. Dessa forma, os bugs (erros) encontrados podem ser corrigidos em tempo real.

2.5 Sensoriamento em IoT

2.5.1 Definição de Sensores

Os sensores são dispositivos interagem aos estímulos físicos ou químicos, ou seja, alguma forma de energia do ambiente, tal como: térmica e luminosa. Sempre correlacionando a grandezas físicas, por exemplo: velocidade, posição, temperatura, umidade. Após a captação os dados são convertidos em sinais elétricos. Normalmente, a resposta desses sensores quando interagem com o meio pode ser um nível de tensão, corrente ou até mesmo uma variação na resistência elétrica do mesmo, geralmente esses dados são transmitidos para microcontroladores para que esse faça o processamento dos dados [26]. Abaixo alguns exemplos de sensores:

- Sensores de pressão: barômetro;

- Sensores de umidade do solo: higrômetro;

- Sensores de corrente elétrica: galvanômetro, amperímetro;
- Sensores de temperatura: termostatos,termômetros, termopares, termístores.

2.5.2 Sensoriamento IoT na Indústria

O Sensoriamento IoT na indústria ganhou força com uma nova ideia de indústria, que é conhecida como indústria 4.0, termo amplamente disseminado e que foi utilizado pela confederação nacional da indústria (CNI), em 2016, mas alguns estudiosos do assunto definem a indústria 4.0 como a quarta revolução industrial. Conforme a CNI, o conceito indústria 4.0 é usado para classificar o modelo de produção quando há integração e monitoramento de um determinado processo industrial por meio da interconexão em rede de diversos equipamentos que possuem sensores agregados e unindo mundo real e virtual, surgindo então os sistemas ciberfísicos e além disso possibilitando o sistemas de inteligência artificial. O conceito de indústria 4.0 pode descrever a aplicação dos conceitos de IoT na indústria, nos diversos segmentos da atividade insdustrial, normalmente empregado no contexto de manufatura de bens de consumo, sabendo que a fábrica inteligente é fundamental para a indústria 4.0.

Conforme o autor Greengard, *et al.*, as habilidades sensoriais e de conectividade de produtos oriundos de uma fábrica inteligente tornam o processo mais dinâmico para às empresas, o que possibilita relizar uma análise a respeito das preferência dos consumidores e usar essa análise para realizar estratégias para a melhoria de um produto, para que esse tenha o sucesso desejado [27]. A seguir alguns exemplos da implementação de IoT em vários segmentos da economia:

a) **Carro inteligente:** Devido ao surgimento da rede 4G e 5G, a evolução da IoT deverá sem bem rápida e a mesma trará inovações nos diversos setores, em vários aspectos. principalmente dentro de uma das áreas profundamente afetadas pela IoT, a industrial, nela encontra-se o ramo automobilístico. Atualmente viabilização da telemática para veículos passou a ganhar força ao ser agregado aos veículos conectividade *wireless*, montadoras disponibilizam uma série de funcionalidade aos veículos, tais como [28]:

   GPS: O sistema de posicionamento global (GPS, do inglês, *global positioning*

system, no carro inteligente é utilizado para permitir que o sistema verifique as melhores rotas e que o veículo leve o motorista, sem a necessidade do motorista dirigir;

Radar: São sensores distribuidos por todo veículo para fazer monitoramento em torno deste em tempo real, sua principal função é fazer um mapeamento referente a localização de outros veículos na estrada;

Lidar: O Lidar funciona como radar, mas o lidar é utilizado para monitoramento de objetos em torno do carro;

câmeras: As câmeras no carro inteligente utilizam tecnologias de inteligência artificial, são capazes de reconhecer placas e sinais de trânsito e até mesmo pedestres nas ruas.

Conforme o autor Behmann, *et al.*, todas essas funcões vão desde serviços básicos relacionados a segurança e conectividade (como por exemplo colisão de veículos e a notificação aos interessados após a ocorrência), até a interação entre dois veículos ou até mesmo interação entre veículo e infraestrutura da rodovia [29] quando o veículo inteligente é dotado de muitos sensores, o mesmo não depende mais do motorista e passa a ser autônomo, pois cumpre a rota de forma automática, sem necessidade de intervenção humana. DesUnsa forma, o gerenciamento de trafego em rodovias nos Estados unidos, definiu níveis de automação em um veículo:

Nível 0: O motorista é totalmente responsável pelo veículo, não fazendo uso de nenhum recurso automatizado;

Nível 1: O sistema automatizado presente no veículo já realiza algumas funções básicas dando suporte ao motorista;

Nível 2: O sistema automatizado presente no veículo poderá realizar determinadas tarefas, como assumir a direção por exemplo;

Nível 3: O sistema pode realizar várias funções (como sensoriamento ambiente, assumir a direção) com supervisão do condutor;

Nível 4: O sistema pode realizar várias funções (como sensoriamento ambiente, assumir a direção) sem supervisão do condutor, mas apenas em algumas situações;

Nível 5: O sistema automatizado presente no veículo é capaz de realizar todas funções de forma autônoma, sem necessidade nenhuma de intervenção humana, podendo assumir a direção em qualquer circunstância.

O carro autônomo é uma ideia muito importante e que poderá até mesmo salvar vidas, pois acidentes podem acontecer geralmente por falha humana (imprudência por exemplo).

b) **Casa Inteligente:** A casa inteligente pode ser compreendida como uma residência onde diversos dispositivos (sensores e atuadores por exemplo) estão interligados em uma rede comum, com o intuito de ofertar serviços de maneira inteligente e automatizadas aos moradores da residência [30]. Dessa forma oferecem maior comodidade, otimizam tarefas da casa, podem oferecer maior segurança e etc. Há inúmeras aplicações de automação em IoT para um casa inteligente, como: Automatizar chuveiro, lâmpada, segurança eletrônica da casa. A baixo algumas das funcionalidades da casa inteligente [31]:

Lâmpadas inteligentes: As lâmpadas poderão ser controladas por meio de apps de smartphone, as lâmpadas inteligentes poderão ser acionadas tanto para apagar como acender em horários pré-definidos, tudo isso pode ser realizado até mesmo de forma remota, quando o usuário está longe do ambiente onde estão as lâmpadas, trazendo comodidade, conforto e segurança;

Controle de Acesso: É o controle de fechadura automatizada remotamente, possibilita que o usuário feche ou abra as portas via aplicativo de um celular. Assim tendo o controle da entrada de pessoas. Podendo até mesmo receber notificação do alarme caso alguém queira entrar sem permissão;

Eletrodomésticos: Os eletrodomésticos serão controlados de forma remota e todas as tarefas vao ser automatizadas. Ou seja, pode-se controlar tudo pelo celular. Alguns deles já poderão ser programados de maneira que sejam interconectados a redes de prestadores de serviço, como por exemplo supermercados, fazendo com que o pedido seja feito de imediato, criando uma lista de itens e quando estiverem acabando, o eletrodoméstico encarregado envia a lista para o supermercado.

2.5.3 Sensoriamento IoT na Agricultura

Conforme o autor Sundmaeker, *et al.*, no agronegócio, a Internet das coisas pode ser utilizada como uma ferramenta muito importante, para o avanço da tecnologia na agricultura e até mesmo para a melhoria da economia nesse setor. As aplicações de IoT na agricultura podem contribuir para ações, como: detectar e afastar predadores através de sensores de presença e som, monitorar a produção (plantio), analisar o crescimento das diferentes culturas da produção, controlar o desenvolvimento dos animais da fazenda, realizar irrigação automática, rastrear animais e o escoamento do produto [32].

Conforme a Empresa Brasileira de Pesquisa Agropecuária (Embrapa), as máquinas poderão interagir sem intervenção humana, trocando informações pela rede, e essas inovações já estão sendo implementadas no campo. E em pouco tempo os equipamentos utilizados na agricultura estarão cada vez sendo conectados a internet. A automatização inteligente empregada na agricultura poderá otimizar os processos produtivos existentes e dar suporte na tomada de decisão do agricultor, possibilitando assim, menores custos na produção e mais lucro para o mesmo. O plano nacional de Internet das Coisas, lançado pelo governo no final de 2017, almeja intensificar o avanço da implementação da internet das coisas em quatro áreas prioritárias: cidades inteligentes, saúde, agricultura e indústria. Estima-se que a implementação gere um impacto de até de 21 bilhões de dólares na agricultura até 2025. Abaixo alguns exemplos de aplicações de sensoriamento IoT na agricultura [33]:

Controle de Pragas e Doenças: O controle é feito por meio de sensores agregados a drones possibilitam o levantamento de dados nas fazendas, os dados coletados são transmitidos a plataformas computacionais que irão dar suporte na identificação de pragas ou doenças presentes nas diversas culturas de plantas existentes nas fazendas. Após o recebimento dos dados levantados por meio do mapeamento da área, as plataformas usam índices de plantas saudáveis como parâmetro para a classificação da saúde da área verificada, sendo esses dados enviados aos equipamentos para a tomada de ação na melhoria das áreas observadas. O manejo verificado por área permite melhor aplicação de agrotóxicos e até mesmo realizam uma economia na quantidade despejada dessas substâncias, ocasionando um maior lucro aos fazendeiros. Um dos pontos mais importantes é o de as ações serem automatizadas

e acontecerem em tempo real.

Rastreabilidade: Implementada geralmente na agropecuária, as tecnologias de internet das coisas, como chips de RFID e código de repostas rápida (QRcode, do inglês, *quick response*), são ferramentas importantes para os fazendeiros, pois possibilitam a coleta de dados referentes a localização e dados de referentes a ração destes além do monitoramento de animais doentes.

Capítulo 3

Metodologia

Este capítulo descreve os procedimentos técnicos utilizados na implementação e monitoramento do sistema de irrigação proposto bem como a obtenção dos dados de interesse via interface gráfica e a finalidade de cada etapa do sistema implementado.

3.1 Sistema Proposto

O projeto proposto consiste na implementação e monitoramento de um sistema de irrigação automatizado utilizando os conceitos de internet das coisas, onde foram utilizados sensores, microntroladores e atuadores, conectados em uma mesma rede wi-fi, onde os dados coletados foram transmitidos via protocolo de comunicação MQTT. A irrigação no sistema realizou-se de forma automática com condição estabelecida em código de umidade do solo menor 50%, mas é importante destacar que a escolha deste valor para a realização da irrigação foi apenas para este projeto proposto, não sendo utilizado a referência de umidade de nenhuma cultura de planta em específico, pois para cada cultura de plantas existe um valor de umidade adequado para irrigação bem específico e o monitoramento observou-se de forma remota, com a exibição dos dados por meio de interface gráfica em *dashboard*.

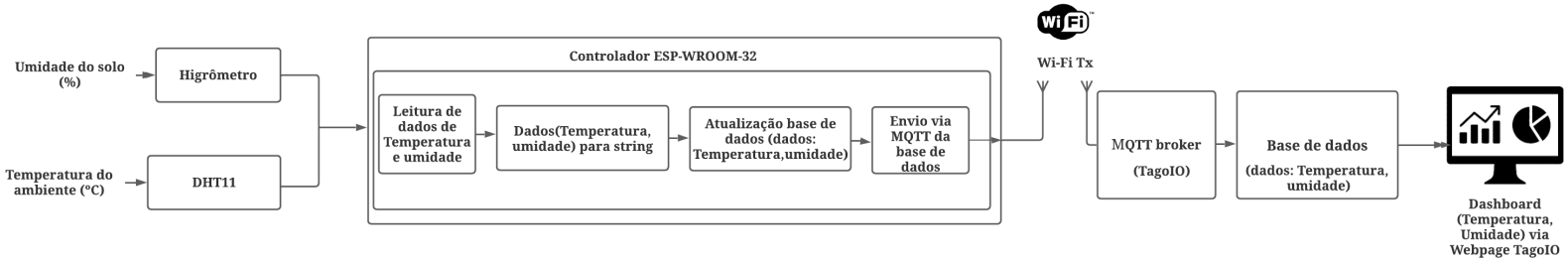

Figura 3.1: Diagrama de blocos do sistema de irrigação proposto.
Fonte: Próprio Autor.

A Figura 3.1 ilustra as etapas do sistema, que ao todo possui cinco blocos que serão detalhados enfatizando as suas particularidades nas próximas subseções.

3.1.1 Objetivos do Sistema Proposto

O sistema proposto tem por objetivo realizar a irrigação automática do plantio apenas com parâmetro de umidade do solo no local e fazer monitoramento dos dados de temperatura ambiente e umidade do solo através de *webpage*, para que o usuário possa ter o controle de seu plantio caseiro de forma remota e receba os dados em tempo real, tendo a possibilidade da tomada de decisões. No projeto proposto foi considerado como tempo real um tempo de latência de até 5 segundos, ou seja, o tempo desde que acontece a leitura e o processamento dos dados captados pelos sensores pelo microcontrolador até a exibição das informações por meios dos gráficos.

3.1.2 Dados de Entrada e Saída do Sistema de Irrigação

Para o bloco dos sensores no sistema, implementou se dois sensores, sensor de temperatura DHT11 e sensor de umidade do solo ou como também é conhecido, higrômetro. A entrada no sensor DHT11 foi a temperatura captada pelo mesmo em ambiente interno e externo, valores analógicos. A saída do sensor DHT11, foram dados de temperatura digitalizados. Para o higrômetro, a entrada do sensor de umidade do solo foi a a umidade do solo captada em solo úmido e solo seco, valores analógicos na entrada e saída deste. Para o bloco do microcontrolador ESP32, as entradas do mesmo foram as saídas dos sensores implementados, ou seja, as entradas foram dados de temperatura ambiente digitalizados vindos do sensor DHT11 e dados analógicos de umidade do solo úmido e seco vindos do sensor higrômetro. A saída foram dados de umidade e tempratura convertidos em strings após o tratamento de dados realizado via IDE do ESP32-WROOM-32. Para o bloco do *broker* MQTT TagoIO, a entrada do bloco foi a saída do microcontrolador ESP32, ou seja, os dados convertidos em strings e anexados em tópicos recebidos pelo *broker* via protocolo MQTT. A saída deste bloco foram os dados por meio de tópicos para vizualização via webpage. No bloco de *webpage* as entradas são os tópicos referentes a umidade e temperatura vindos do servidor intermediário (*broker* MQTT TagoIO) e a saída é

a exibição de gráficos dos dados recebidos.

3.1.3 Visão Geral dos Cenários de Testes

Os teste realizados para a temperatura foram elaborados para a verificação da temperatura em tempo real por meio de gráficos em Dashborad. Os cenários de teste para experimentos realizados com o sensor de temperatura DHT11, aconteceram por meio da aferição da temperatura ambiente em local de interesse (ambiente aberto) no primeiro experimento. No segundo experimento foi realizado em ambiente aberto com uma ferramenta de ar quente (secador de cabelo). Os testes realizados para umidade foram realizados com objetivo de verificar o valor de umidade monitorada em tempo real via gráfico no *dashboard* e executar a irrigação caso a umidade do solo estivesse abaixo de 50% . O primeiro cenário de teste aconteceu em ambiente fechado sendo realizado em um copo de água. No segundo cenário em solo úmido e solo seco do plantio caseiro em ambiente aberto.

3.2 Sensores Clientes MQTT

3.2.1 Sensor DHT11

A função do sensor DHT11 no sistema proposto foi de fazer a captação e monitoramento da temperatura ambiente do plantio caseiro e transmitir os dados captados ao microcontrolador ESP32, onde o dados foram tratados e enviados por meios de tópicos para exibição das informações por meio de gráficos de temperatura. O sensor DHT11 utilizado possui um componente interno para a medição de temperatura, um termistor do tipo NTC, ou seja, um resistor sensível as variações de temperatura e este componente é controlador por um microcontrolador interno do tipo MCU. Os valores de temperatura aferidos pelo NTC foram digitalizados pelo chip MCU, ou seja, os dados de temperatura transmitidos ao ESP32 eram digitais, por tal motivo a leitura do DHT11 não precisou ser realizada em um pino com função ADC do microcontrolador ESP32. O uso deste componente contribuiu para que o usuário estivesse acompanhando o monitoramento da temperatura de forma remota devido aos conceitos de IoT utilizados e em tempo real porque o sensor possui um

baixo tempo de resposta, ou seja, tempo de envio das informações. Não foi necessário qualquer tipo de calibragem para que o sensor DHT11 utilizado funcionasse da forma esperada, vale ressaltar que o mesmo faz captação apenas de temperaturas positivas, o que está de acordo com objetivo deste no projeto.

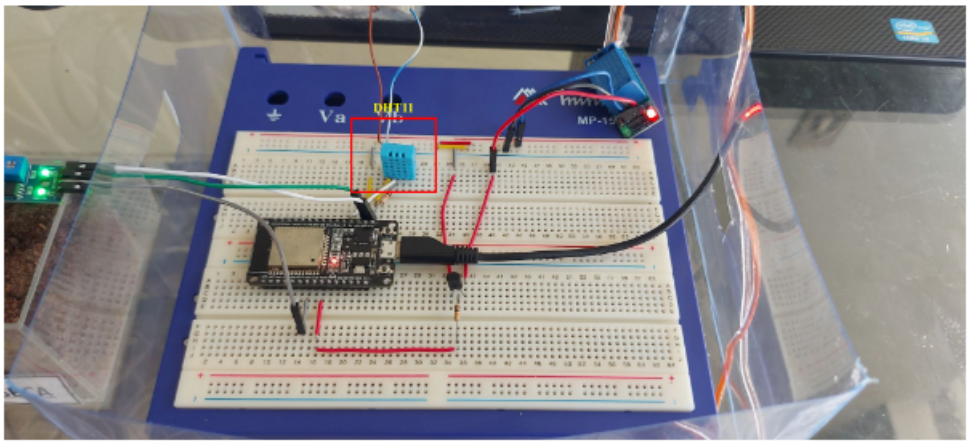

Figura 3.2: Sensor de temperatura DHT11 utilizado no projeto.
Fonte: Próprio Autor.

No sistema proposto o DHT11 é um cliente MQTT de publicação, pois o mesmo apenas fez a transmissão dos dados de temperatura coletados no local. Para que o sensor fosse capaz de captar a temperatura ambiente e realizar a transmissão para o microcontrolador, foi necessário o DHT11 ser alimentado com um nível de tensão de 3,3V DC, da placa do microcontrolador ESP32. Além de ser implementando um código na linguagem C++ no ambiente de desenvolvimento integrado IDE do ESP32, para que o microcontrolador fizesse a leitura dos dados coletados. O código bem como a leitura deste pelo microcontrolador será detalhada nas próximas seções.

3.2.2 Sensor Higrômetro

O sensor higrômetro utilizado é um cliente MQTT de publicação de suma importância no projeto. A função deste sensor no sistema é de captar os dados de umidade do solo e transmiti-lás ao microcontrolador para que a partir da verificação das condições de umidade feitas no ESP32, a irrigação aconteça de forma totalmente

automatizada e que os dados possam ser visualizados de forma remota.

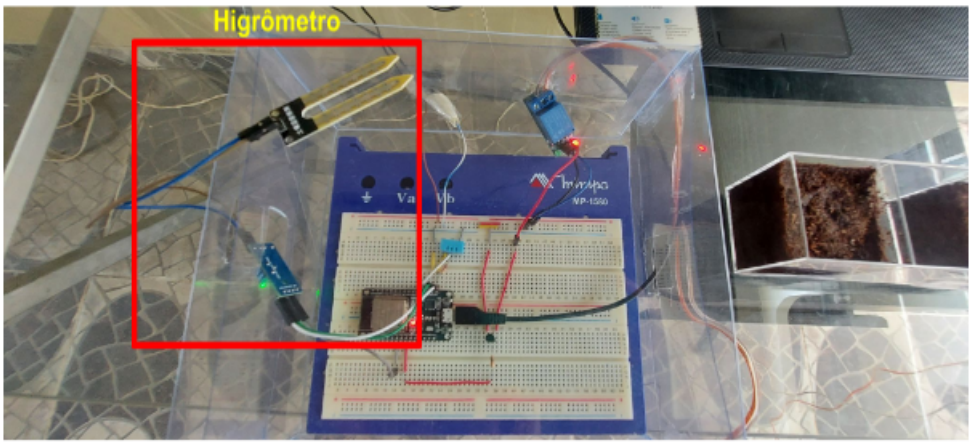

Figura 3.3: Sensor Higrômetro no protótipo utilizado no projeto.

Fonte: Próprio Autor.

No referido sensor foi feito apenas uso de sua pinagem analógica, ou seja, a informação de entrada do sensor higrômetro é analógica e a saída igualmente, pois o mesmo não possui conversor DAC na sua parte interna, sendo transmitidos ao microcontrolador dados puramente analógicos. O sensor utilizado no projeto não estava calibrado para utilização deste pino (analógico), portanto foi necessário a calibragem analógica deste, sendo feita por meio de código na linguagem C++ implementada no ambiente de desenvolvimento IDE do microcontrolador ESP32.

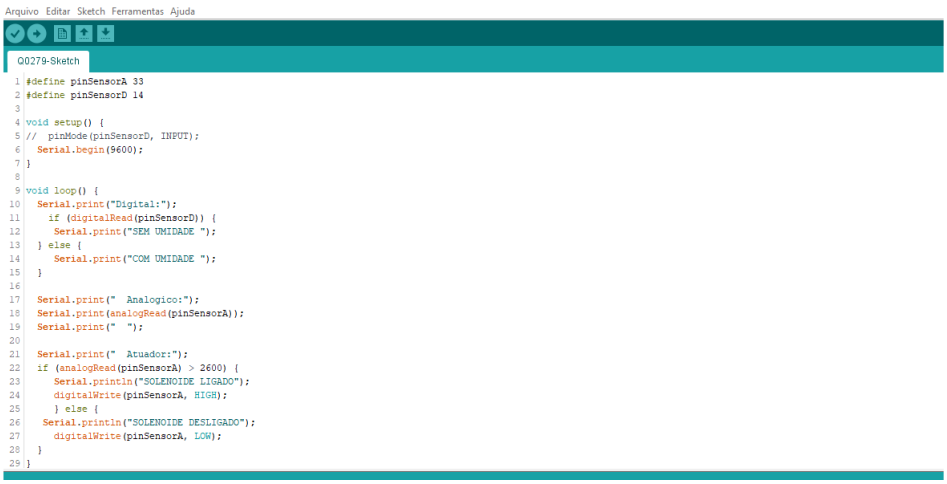

Figura 3.4: Código de calibragem do sensor Higrômetro utilizado no projeto.

Fonte: Próprio Autor.

O código na imagem acima foi utilizado para fazer a calibragem do sensor higrômetro tanto no primeiro experimento (copo com água) como no segundo experimento (solo úmido e seco).

3.3 Atuadores do Sistema de Irrigação

No sistema de irrigação implementado existem três atuadores, são: Módulo relé, válvula solenoide, aspersores para irrigação. Esses dispositivos de acionamento e controle foram utilizados no sistema fazendo o chaveamento da passagem da água, fazendo o controle desta.

3.3.1 Módulo Relé utilizado no Projeto

O módulo relé utilizado foi um módulo digital de 1 canal e bobina de 5VDC. O módulo relé foi acionado por meio de uma fonte de alimentação externa de 5VDC, para que o mesmo trocasse a posição do contato, que era normalmente aberta para normalmente fechada e simultaneamente foi conectado a um pino digital do microcontrolador ESP32 para receber o sinal deste quando a condição de umidade do solo fosse menor que 50%, por meio de código implementado em C++ em ambiente IDE.

3.3.2 Válvula Solenoide utilizada no Projeto

A válvula utilizada no sistema de irrigação é do tipo normalmente fechada e com bobina de 12V DC. Para seu acionamento a mesma foi alimentada com uma fonte externa, uma fonte de bancada com o valor de tensão de 12,5V DC. A válvula foi conectada ao módulo relé e a uma mangueira cristal com água da rua e passagem da torneira já aberta. A função da válvula solenoide no projeto é o controle da água, liberando a passagem caso a condição de umidade menor que 50% fosse satisfeita, assim o relé fecha o contato e a mesma recebe o nível de tensão da fonte de bancada.

Figura 3.5: Válvula solenoide utilizada no projeto.

Fonte: Próprio Autor.

Acima a imagem da válvula solenoide com os engates para mangueira, onde uma mangueira está conectada a torneira e as outras duas nos dois aspersores.

3.3.3 Aspersores utilizados no Sistema

A função dos aspersores no projeto implementado é fazer a água alcançar maior distância e fazer uma irrigação bem distribuída. O acionamento destes no projeto se deu junto com a válvula solenoide, estando os dois dispositivos integrados.

Figura 3.6: Aspersores utilizados no projeto.

Fonte: Próprio Autor.

3.4 Esquemático do Circuito

O circuito do projeto teve o protótipo montado em *protoboard*, dispositivos montados e testados primeiramente de forma individual para ter a validação individual de cada dispositivo antes da integração final. Os componentes montados em protoboard, são:

- Módulo ESP32-WROOM-32;
- Sensor de temperatura DHT11;
- Sensor de umidade do solo higrômetro;
- Módulo relé de 1 canal de 5V;
- Transistor BC547;
- 2 Resistores de 10k ohms.

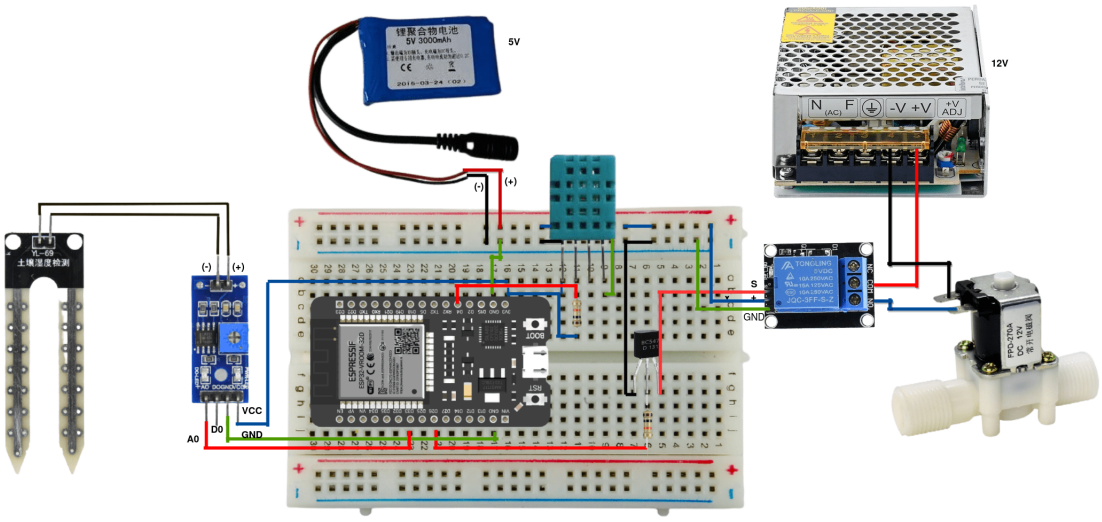

Figura 3.7: Esquemático do sistema de irrigação proposto.

Fonte: Próprio Autor.

Abaixo o detalhamento das conexões de cada dispositivos utilizados e que foram implementados em protoboard:

a) **Sensor higrômetro:**

- O sensor utilizado teve o seu pino analógico (A0) conectado ao pino 33 (pino com função ADC) do módulo ESP32, para que o microcontrolador fizesse as leituras de umidade através do pino 33;
- Pino GND do higrômetro conectado ao pino GND da placa;
- Pino vcc do higrômetro alimentado com 3,3V da placa.

b) **Sensor DHT11:**

- O sensor DHT11 utilizado teve um resistor de *pull-up* no valor de 10k ohms associado ao seu pino data, para evitar a indeterminação do nível lógico nessa entrada, mantendo assim a mesma em nível lógico alto;
- O pino data foi conectado ao pino de alimentação de 3,3V da placa e ao pino 4 para a transmissão dos dados de temperatura ao microcontrolador;
- O pino GND do DHT11 conectado ao negativo da fonte externa de 5V.

c) **Módulo Relé:**

As conexões de acionamento módulo relé, são:

- O pino 26 da placa utilizado para a comutação do relé foi conectado a um resistor de 10 k ohms, que estava associado a base do transistor Bc 547;
- Pino do data do relé ao emissor do transistor Bc 547;
- Positivo do relé ao positivo e negativo do relé conectados a fonte externa de 5V.

Conexões de saída do relé utilizado:

- Pino COM do relé conectado ao positivo da fonte de bancada com valor de tensão em 12,5V DC;
- Pino normalmente aberto (NA) conectado a um dos bornes da válvula solenoide e a mesma fechava o circuito estando conectada com o seu outro borne ao negativo da fonte de bancada.

d) **Módulo ESP32:**

- O módulo ESP32 utilizado foi alimentado com cabo micro USB conectado ao Notebook;
- Os pinos utilizados foram pino 33 (higrômetro), pino 26 (relé), pino 4 (DHT11), GND e VCC.

3.5 *Broker* MQTT TagoIO

A TagoIO foi a escolhida para ser a plataforma de aplicação com serviço de *broker* mqtt, por apresentar inúmeros recursos e possuir uma menor limitação quando comparada com outras plataformas, como a ProIoT por exemplo, mesmo quando utilizada na sua versão gratuita, que foi o caso deste projeto e por se uma plataforma voltada para projetos que envolvem protocolo de comunicação MQTT. O nível de qualidade do serviço utilizado foi o QoS 0, que é o nível de qualidade do serviço padrão para conta gratuita.

A TagoIO é uma plataforma em nuvem do tipo software de código aberto muito utilizada para soluções envolvendo Internet das coisas e a mesma é uma da várias plataformas que prestam suporte como *broker* MQTT, que como mencionado anteriormente é um servidor intermediário entre clientes MQTT (*publisher/subscriber*). A TagoIO combina o seu suporte MQTT com vários recursos existentes na plataforma, como por exemplo a criação de painéis *(dashboards)*, que podem ser entendidos como a interface para a exibição das informações em uma comunicação e principalmente em sistemas que utilizam o protocolo MQTT, além disso os Dashboards possibilitam o monitoramento dos dados, notificações e relatórios acerca dos mesmos. e por se uma plataforma voltada para projetos que envolvem protocolo de comunicação MQTT.

3.5.1 Interface Gráfica TagoIO

Foi feito o cadastro de usuário na plataforma TagoIO, a mesma foi utilizada na versão gratuita e primeiro foi criado um dispositivo no perfil cadastrado. Na imagem Abaixo é possível ver a página inicial da plataforma com os botões para criação, configuração de dispositivos e criação de painel *dashboard*.

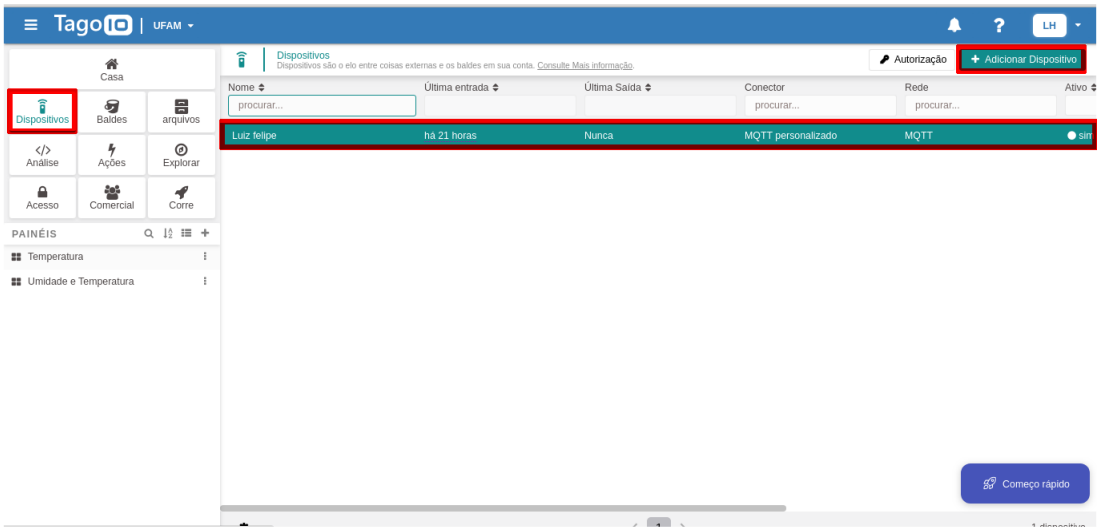

Figura 3.8: Criando device na plataforma TagoIO.

Fonte: Próprio Autor.

O primeiro passo para criar o *device* na plataforma foi por meio do botão dispositivos e em seguida adicionar dispositivo. Após isso foi dado um nome ao dispositivo (Luiz Felipe) e ao clicar no nome deste é possível verificar algumas informações muito importantes como o token por exemplo, que foi utilizado posteriormente no código, com objetivo da identificação do usuário na integração (ESP32/TagoIO), por meio do código em C++ no ambiente de desenvolvimento.

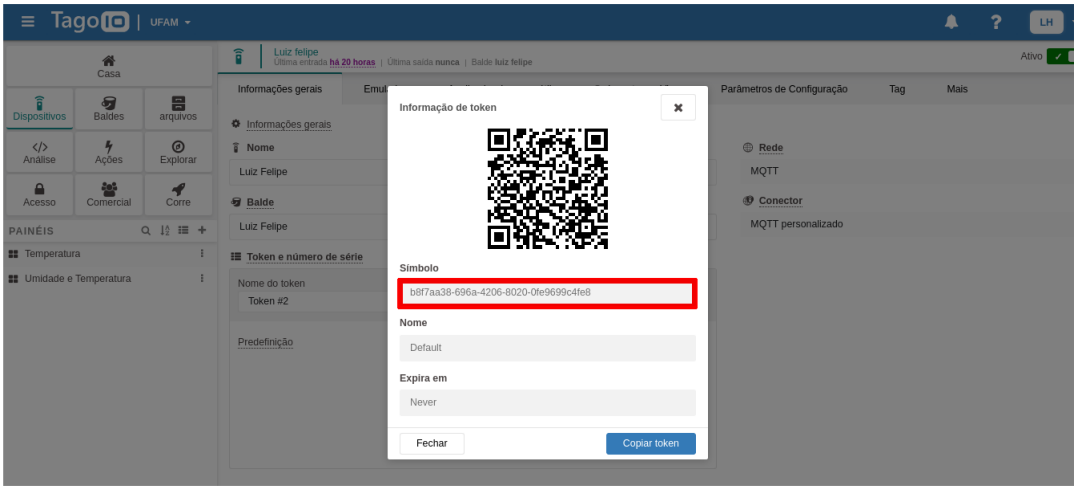

Figura 3.9: Token de usuário TagoIO criado no projeto.

Fonte: Próprio Autor.

3.5.2 Configuração do *Device*

Para a exibição das informações recebidas via protocolo MQTT pelo dispositivo criado na plataforma TagoIO, foi criado um painel dashboad nomeado de umidade e temperatura e dentre os diversos gráficos foi escolhido e utilizado no projeto o modelo de gráfico *solid*.

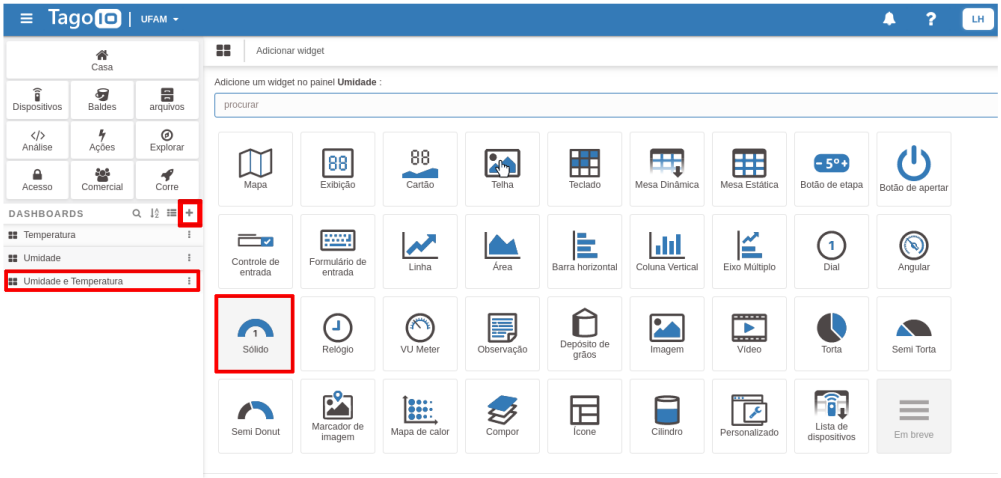

Figura 3.10: Painel dashboard umidade e temperatura com gráfico solid.

Fonte: Próprio Autor.

No projeto foram construídos dois gráficos um para cada variável (umidade e temperatura). Após a escolha do gráfico foram feitas as configurações nos gráfico e inseridas às variáveis correspondentes aos gráficos. Na imagem abaixo é possível visualizar o nome do dispositivo (Luiz Felipe) e a variável correspondente (temperatura) ao gráfico que estava sendo criado. Todos os gráficos foram configurados para que os dados exibidos enquanto os sensores estivessem funcionando e armazenado o último valor exibido, além disso o tamanho das variáveis para 100 em Y. É importante ressaltar que as variáveis anexadas aos gráficos no *dashboard* foram escritas com os caracteres de forma igual no código em c++.

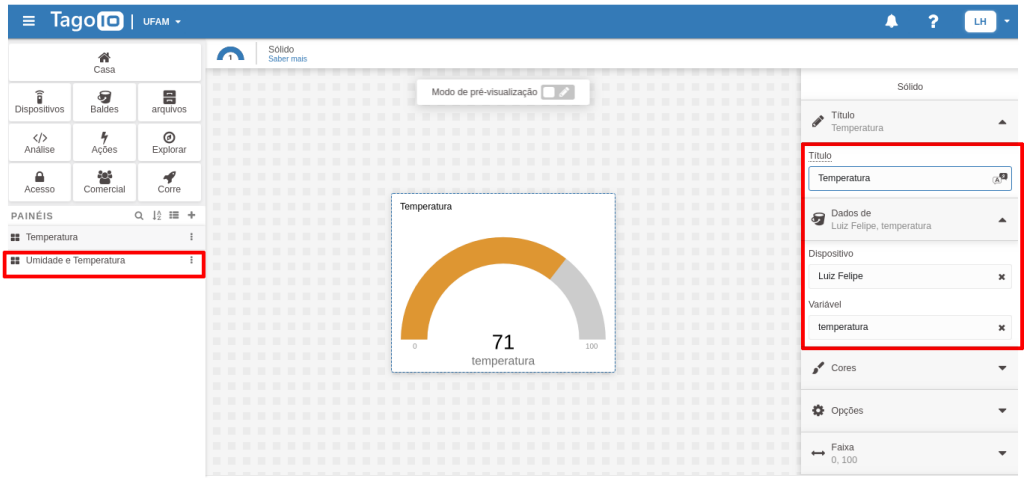

Figura 3.11: Configuração dos gráficos de umidade e temperatura e integração da variável.

Fonte: Próprio Autor.

A imagem abaixo mostra o *dashboard* temperatura e umidade após o primeiro experimento com quatro gráficos criados, dois do tipo solid e dois em linha com os últimos valores exibidos de umidade e temperatura.

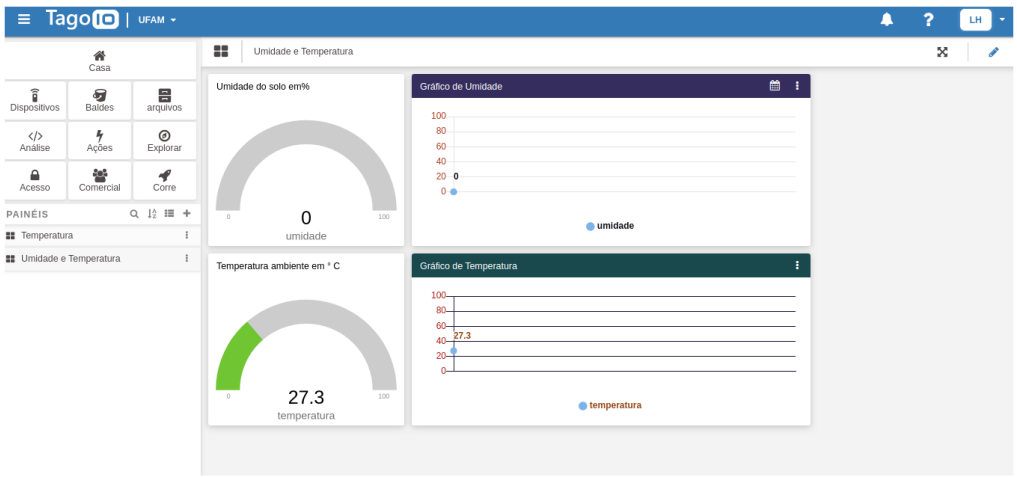

Figura 3.12: Painel umidade e temperatura.

Fonte: Próprio Autor.

3.6 Integração do Código na IDE

3.6.1 Requisição de Dados via Código na IDE

No sistema de irrigação foram utilizados três códigos, um para o higrômetro, para o DHT11 e para o módulo relé. Para a leitura do dados de temperatura por meio do controlador ESP32, primeiramente foi incluida a biblioteca através de include DHTesp.h do DHT11 no ambiente de desenvolvimento integrado IDE e após isso foi definido DHTpin 4, pois o pino data do DHT11, estava conectado ao pino 4 do ESP32, para fazer a transmissão do sinal por meio desse pino. Para o sensor higrômetro não foi utilizada nenhuma biblioteca no código e o pino analógico do mesmo foi lido pelo microcontrolador através de define pinSensorA 33. Para o relé foi utilizada a porta 26 do ESP32, para realizar seu acionamento diante da condição de umidade menor que 50%, ou seja, quando a umidade do solo captada pelo sensor Higrômetro fosse menor que 50% o relé iria comutar.

3.6.2 Tratamento de Dados do Sistema

O tratamento de dados realizado foi necessário para fazer o envio dos dados para a plataforma TagoIO, pois há uma limitação no protocolo de comunicação MQTT em receber mensagens apenas do tipo string e simultaneamente foi checado no monitor serial se o dados estavam sendo lidos e enviado corretamente. Após os sensores captarem os dados e transmit-los ao microcontrolador, os dados foram armazenados em variáveis do tipo *float* no ambiente IDE, então os dados de temperatura foram armazenados em *float* temperatura e dados de umidade em *float* umidade, pois aceita números com casa decimais.

Após esse processo os dados das variáveis do tipo *float* umidade e temperatura foram armazenado em variáveis do tipo *string*. Dessa forma, foi escrito str_ umi e str_ temp e por último as essa variáveis foram armazenas em tópicos MQTT. Na imagem abaixo é possível ver os valores de umidade e tempratura no monitor serial e simultaneamente os dados enviados a plataforma TagoIO.

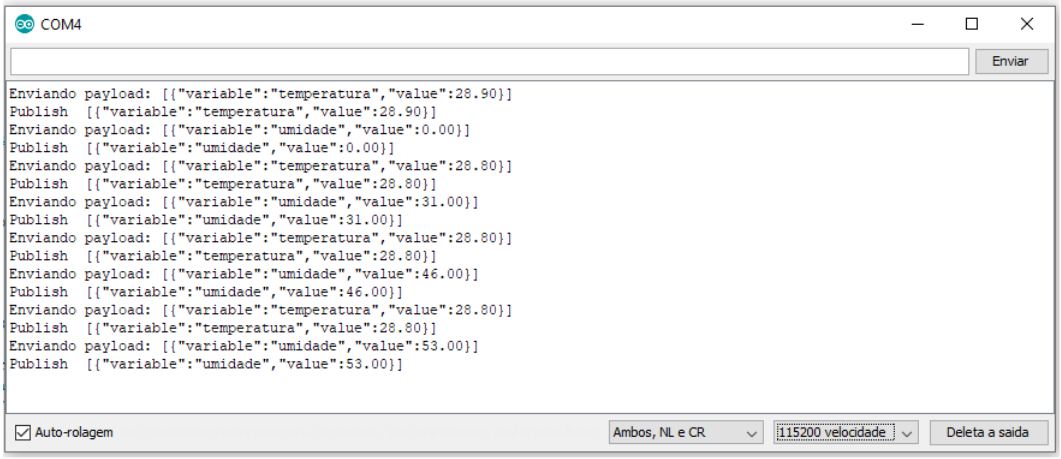

Figura 3.13: Monitor serial do IDE com valores de umidade e temperatura.

Fonte: Próprio Autor.

3.6.3 Integração dispositivo/broker MQTT

Para a integração do microcontrolador ESP32 com a Internet, o dispositivo foi conectado a rede local através de comandos no código em C++. O ESP32 foi testado em conexão via wi-fi residencial e wi-fi de Internet móvel. Após a configuração da conexão com a Internet foi feita a integração dispositivo *broker* MQTT TagoIO.

Para a comunicação com o servidor Mqtt TagoIO, foi utilizado o host mqtt.tago.io, a porta TCP/IP utilizada foi a porta 1883, umas das portas padrão do protocolo MQTT. Para a comunicação com o usuário TagoIO, MQTT *user*, foi utilizado *default*, que é o *user* padrão ao ser criado um dispositivo na plataforma TagoIO.

A chave de comunicação do dispositivo com o usuário, MQTT *password*, foi anexado o token do dispositivo criado, para a comunicação deste com a TagoIo. Para a publicação o modelo de envio do topico utilizado foi o "*tago/data/post*", que é o padrão TagoIO e Para a assinatura do topico o modelo utilizado foi *tago/my_ topic*. Todos estes dados inseridos em código c++ implementados no ambiente IDE, foram retirados da documentação da plataforma TagoIO.

Capítulo 4

Cenários de Teste e Resultados

Neste capítulo serão apresentados os experimentos realizados para umidade e temperatura do sistema proposto, assim como os resultados obtidos com intuito de avaliar o desempenho e características do sistema implementado.

4.1 Cenário 1 - Testes de Calibragem do Sensor de Umidade em Água

No teste foi utilizado o sensor de umidade do solo em ambiente fechado e foi utilizado um copo com água para que o sensor fosse introduzido no mesmo. O experimento foi realizado em duas etapas, são: sensor totalmente fora da água e totalmente imerso em água.

O objetivo do experimento foi de realizar a calibragem do sensor higrômetro, verificar se os dados de umidade eram proporcionais ao nível de sonda introduzida em água e visualizar os dados de umidade no monitor serial.

Os resultados esperados eram que calibragem fosse válida tanto para água como para o solo e que os dados coletados fossem coerentes com o nível do sensor introduzido em água.

- **Primeira etapa da calibragem:** Na primeira etapa o código descrito na seção 3.2.2, foi compilado no IDE do ESP32, para que fosse possível visualizar qual seria o valor referência da umidade exibida no monitor serial, estando o sensor totalmente fora da água.

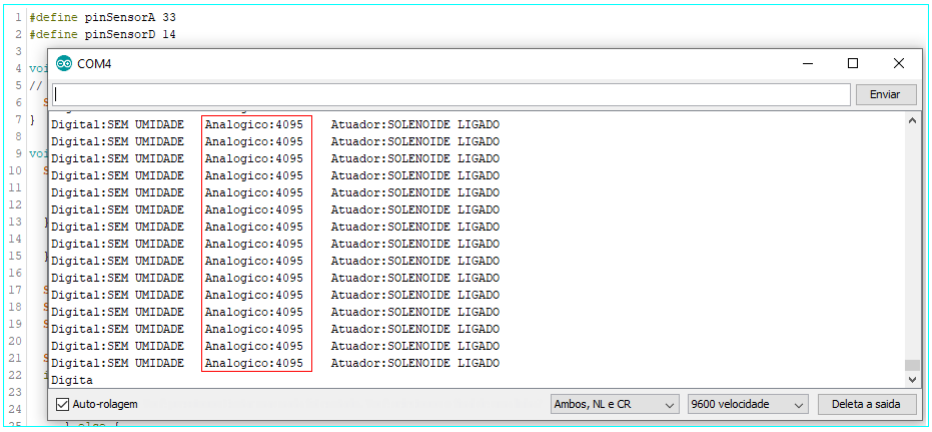

Figura 4.1: Valor referência de umidade da primeira etapa via monitor serial .

Fonte: Próprio Autor.

O valor exibido no monitor serial da IDE, pela captação do pino analógico do sensor de umidade teve como referência o valor 4095, estando o sensor completamente fora da água. O pino que foi utilizado para a transmissão e leitura dos dados, pino 33, possui a função de conversor do tipo ADC, por isso possui uma resolução de 12 bits, então a faixa total de leitura é de 0 a 4095 valores. O objetivo da etapa era ter um valor mínimo de referência e esse valor foi 4095.

1. **Segunda etapa da calibragem:** Na segunda etapa foi realizada a experiência para obter o valor referência de umidade quando o sensor estivesse totalmente imerso em água.

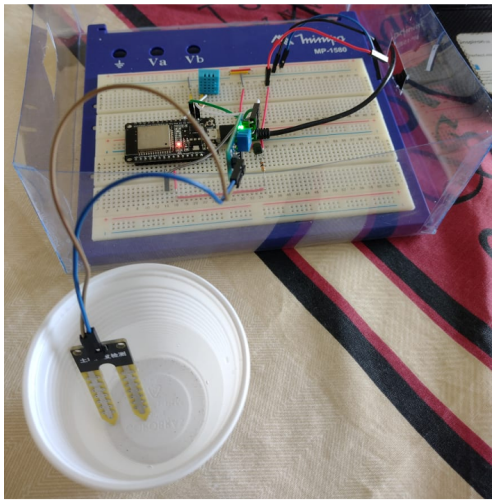

Figura 4.2: Sensor higrômetro completamente imerso em água .

Fonte: Próprio Autor.

Observou-se que na medida que o sensor de umidade era imerso na água os valores de referência exibidos nos monitor serial diminuíam, o que aconteceu até que o mesmo estivesse completamente envolvido em água. O objetivo era ter valor de referência máximo para depois converter esses valores de máximo e mínimo em em valores percentuais de umidade.

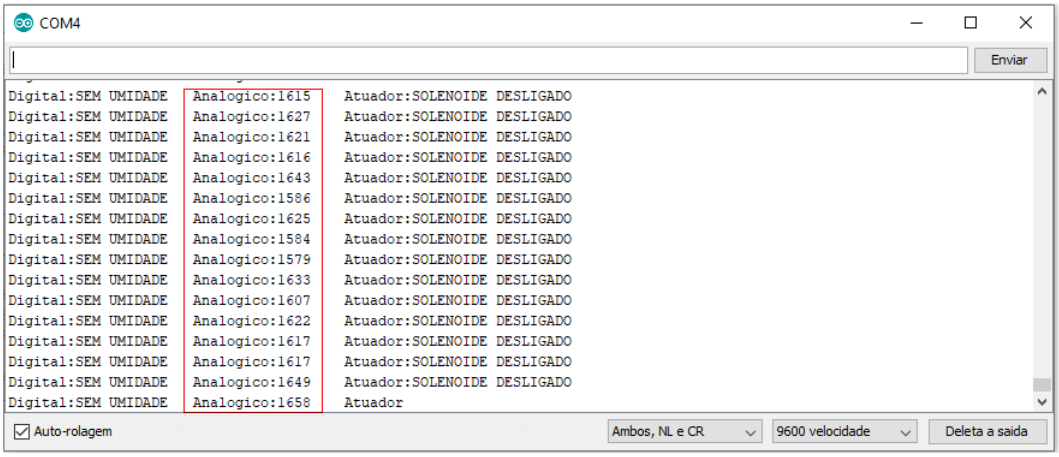

Figura 4.3: Valor referência de umidade da segunda etapa via monitor serial .

Fonte: Próprio Autor.

Após a verificação dos dados no monitor serial foi feita uma média de 5 valores mínimos de referência para realizar a conversão dos dado em valores percentuais. Então foi utilizada uma função implementada no código principal em C++, a função map para converter em valores percentuais. Então foi novamente verificado como os dados estavam sendo exibidos no monitor serial.

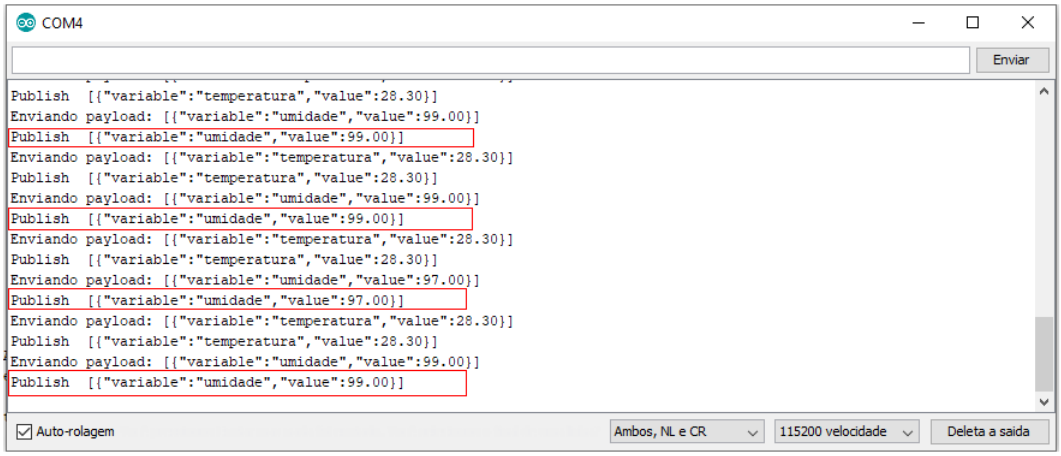

Figura 4.4: Exibição dos valores percentuais de umidade via monitor serial .

Fonte: Próprio Autor.

Na imagem acima é possível ver que os dados percentuais de umidade e valores de temperatura já estão sendo publicados na plataforma TagoIO. Os dados são enviados por meios dos tópicos de temperatura e umidade, que carregam as informações através da carga útil (*payload*, do inglês), que é a real mensagem pretendida.

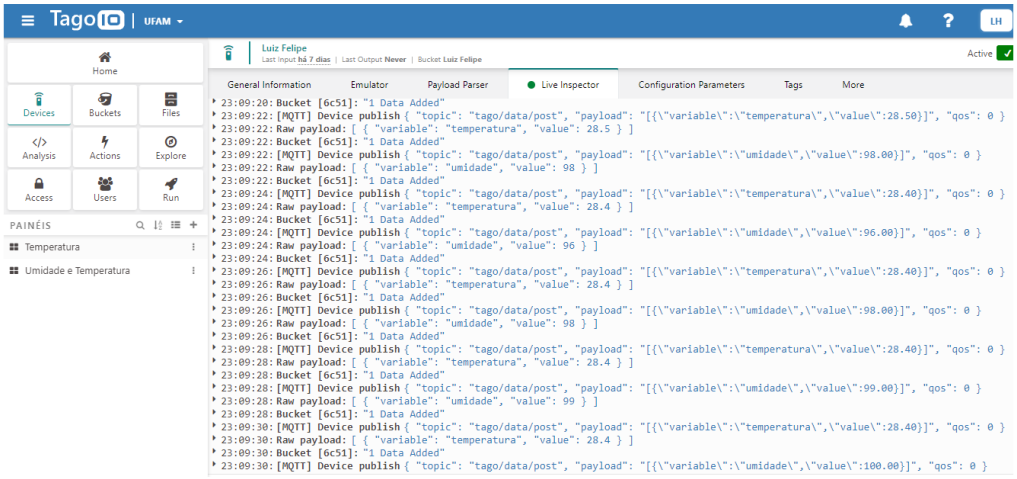

Figura 4.5: Tópico sendo recebido na plataforma TagoIO.

Fonte: Próprio Autor.

Na imagem acima pode-se ver o tópico sendo recebido por meio da recurso live inspector. O tópico é mostrado com o seu formato de envio *tago/data/post* e o *paylod* que representa a informação real de umidade e temperatura, além disso é possível ver

o nível de qualidade de serviço, que no caso foi QoS 0. Neste experimento os dados obtidos foram visualizados apenas no monitor serial e e no recurso live inspector para verificação dos tópicos sendo recebidos em tempo real.

4.2 Cenário 2 - Testes de Irrigação do Solo

O experimento proposto foi realizado em ambiente aberto e no local de interesse, ou seja, no plantio caseiro e foi analisado mediante duas possibilidades, solo seco e solo úmido.

É importante ressaltar que nesse experimento todos os dispositivos presentes no projeto de irrigação fizeram parte do teste e foram utilizados e além disso foi feita uma nova calibragem, mas nesse experimento foi diretamente em solo seco e úmido e os resultados da calibragem em solo foram muito próximos dos realizados em água.

O sensor de umidade foi utilizado para fazer captação da umidade no solo e transmiti-las ao microcontrolador e por conseguinte ao dashboard TagoIO e simultaneamente havia uma condição sendo constantemente verificada pelo microcontrolador, que foi implementada em código C++ para que os atuadores fossem acionados caso o valor de umidade lido pelo higrômetro fosse menor que 50%, nesse momento o relé iria ser acionado e iria comutar de normalmente aberto para normalmente fechado, ao comutar o relé iria acionar a válvula solenoide e por conseguinte os dois aspersores para realizar a irrigação.

Neste experimento não teve qualquer tipo de condição estabelecida para o sensor de temperatura DHT11, o mesmo apenas fez a captação de dados do ambiente e transmitiu esses cados ao microcontrolador e por conseguinte a plataforma TagoIO. O objetivo do teste é realizar uma irrigação automatizada no plantio caseiro e o seu monitoramento de temperatura e umidade dentro dos conceitos de internet das coisas via procolo MQTT, por meio de seu modelo de publicação/ assinatura e condição predeterminada em código c++, tendo assim um monitoramento remoto do plantio caseiro por meio da verificação dos dados de temperatura e umidade do solo em dashboard para uma possível tomada de decisão do usuário.

A saída esperada deveria ser uma leitura de umidade do solo maior que 50%

de e umidade em terra úmida, 0% de umidade em terra seca e a irrigação deveria acontecer quando o sensor estivesse imerso em terra seca, por satisfazer a condição de umidade $< 50\%$, que foi pré estabelecida no código utilizado no projeto e os dados de temperatura e umidade lidos pelos sensores deveriam ser exibidos constantemente no Dashboard TagoIO, independente da condição estabelecida.

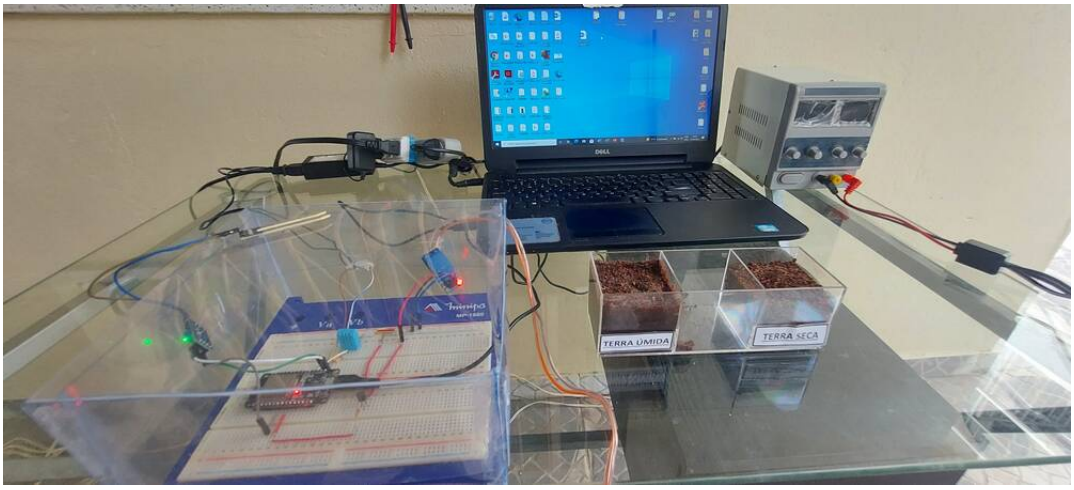

Figura 4.6: Cenário de verificação da umidade do solo.
Fonte: Próprio Autor.

A imagem acima mostra o cenário onde foi feito a verificação do teste de umidade do solo. Neste cenário, o circuito estava todo integrado, com sensores os sensores DHT11 e higrômetro energizados pelo microcontrolador ESP32, porém o código não havia sido compilado para a leitura e transmissão das informações ao microcontrolador.

O atuador módulo relé estava energizado pela fonte de 5V colocada na régua de tomada e o atuador válvula solenoide energizado com 12,5V DC da fonte de bancada. Foi colocado um recipiente com terra úmida e terra seca, onde o sensor primeiramente foi completamente introduzido em terra úmida, sendo captados os valores de umidade e temperatura ambiente onde seriam transmitidos em tempo real a plataforma TagoIO.

Figura 4.7: Sensor higrômetro completamente imerso em terra úmida.

Fonte: Próprio Autor.

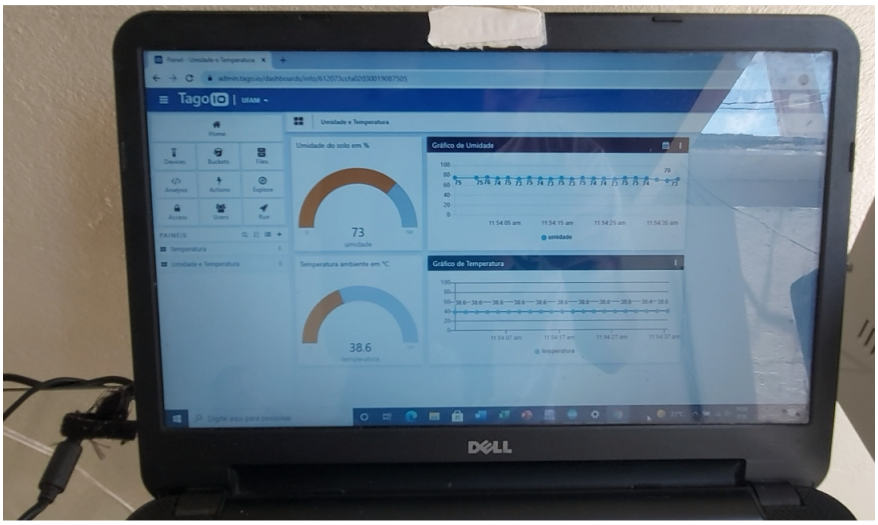

Figura 4.8: Dados via dashboard TagoIO do higrômetro completamente imerso em terra úmida.

Fonte: Próprio Autor.

Para a primeira possibilidade foi exibido o valor 73% de umidade no gráfico do tipo solid e uma variação 73% a 75% no gráfico de linha. além disso a temperatura ambiente registrada no momento do teste foi de 38,6 °C no gráfico solid e praticamente linear em 38,6 °C no momento da medição. Logo após a verificação dos gráficos da primeira possibilidade, o sensor higrômetro foi colocado na terra seca para constatar a diminuição da umidade até 0%, e novamente foi observado *dashboard* para a verificação dos dados em tempo real.

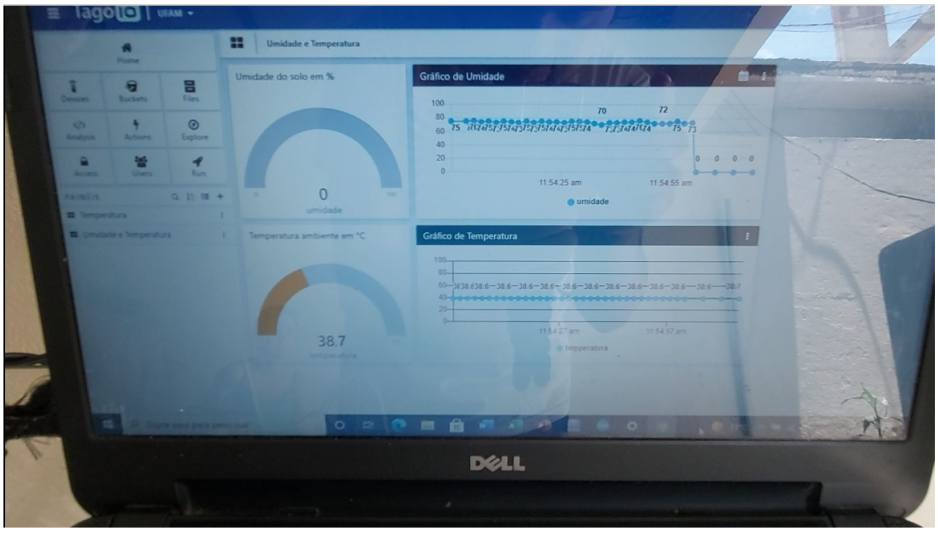

Figura 4.9: Dados via *dashboard* TagoIO do higrômetro completamente imerso em terra seca .

Fonte: Próprio Autor.

No momento em que o microcontrolador processa os dados e verifica que a condição de umidade implementada em código c++ no ambiente IDE está sendo satisfeita, o módulo relé é acionado por meio de código integrado e fecha o contato de seu circuito interno.

O relé permanece acionado por 8 segundos, condição que foi pré determinada em seu algoritmo. No processo a válvula solenoide é acionada pelo relé quando o mesmo fecha seu contato e permite que a válvula receba em seus bornes os 12,5V DC da fonte bancada, nesse momento a válvula permite a passagem da água até os aspersores, onde acontece a irrigação.

Figura 4.10: Implementação da irrigação devido a baixa umidade captada pelo sensor. Fonte: Próprio Autor.

4.3 Cenário 3 - Teste de Variação de Temperatura

O teste foi realizado com sensor de temperatura DHT11 em ambiente externo com um secador de cabelo ventilando ar quente no sensor por aproximadamente 15 segundos. O objetivo deste experimento foi de verificar o tempo de atraso no envio dos dados da elevação da temperatura captada e visualizar a variação da temperatura no *dashboard* TagoIO em tempo real.

A saída esperada deveria ser a visualização da variação de temperatura em tempo real sendo exibida no painel *dashboard* TagoIO com o mínimo de atraso possível.

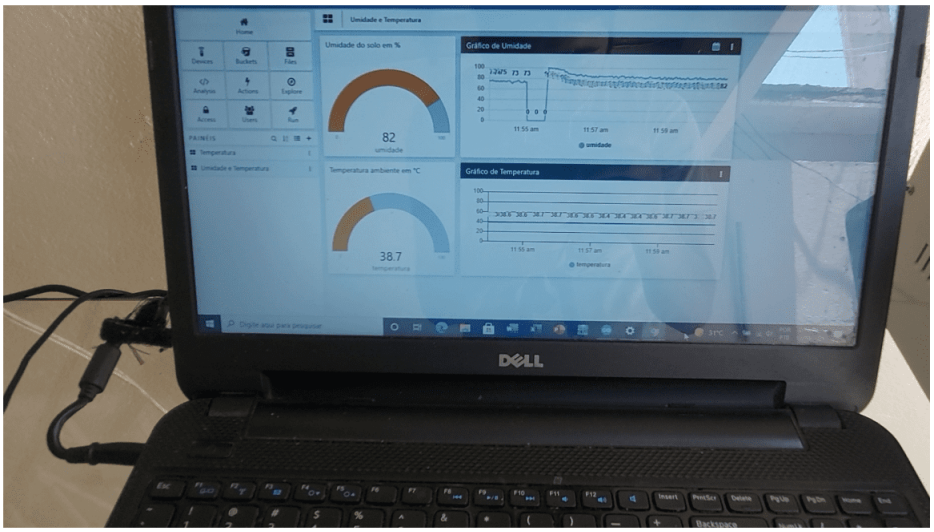

Figura 4.11: Gráfico de temperatura ambiente antes do experimento com ar quente.
Fonte: Próprio Autor.

Na Figura 4.11 é possível visualizar os gráficos de umidade e temperatura, mas nesse cenário o foco do teste em si foi apenas para a temperatura. No gráfico do tipo solid é possível ver uma de temperatura quase que constante de 38,7 °C no momento que foi aferido, antes do sensor DHT11 receber o ar quente do secador de cabelo. Após a verificação da temperatura ambiente por meio do painel dashboard TagoIO em tempo real, foi colocado o secador de cabelo ventilando ar quente.

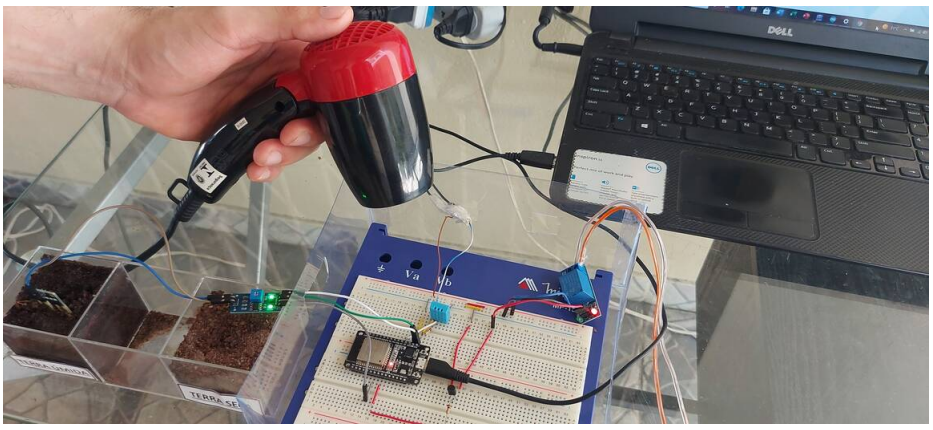

Figura 4.12: Representa a aplicação de teste de ar quente.
Fonte: Próprio Autor.

o ar quente foi ventilado por aproximadamente 15 segundos e nesse tempo os valores de temperatura foram aumentando de acordo com o tempo de exposição

do sensor DHT11 e os dados foram todos exibidos no painel dashboard TagoIO em tempo real. Na imagem abaixo o gráfico solid de temperatura estava exibindo uma temperatura de 44,6 °C, após alguns segundos de teste.

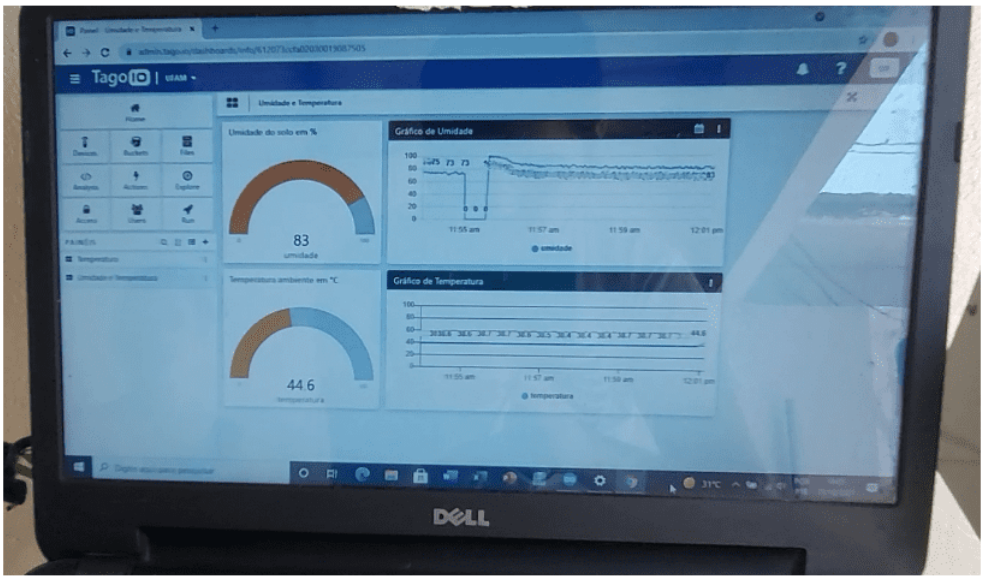

Figura 4.13: Temperatura registrada após segundos de teste com ar quente.
Fonte: Próprio Autor.

A temperatura registrada e exibida por meio dos gráficos ao no final do tempo estipulado para o teste foi de 46,5 °C, exibidos em tempo real.

Capítulo 5

Conclusão

Neste trabalho apresentou-se um sistema que realiza uma irrigação automatizada no plantio caseiro e monitoramento de umidade e temperatura deste, baseados nos conceitos de Internet das coisas, que foram de suma importância no projeto, pois possibilitou um en tendimento a cerca da interação dos dispositivos utilizados.

O protocolo de comunicação de máquinas MQTT, mostrou-se eficaz na comunicação dos dispositivos envolvidos no projeto, como: sensores, atuadores e usuário (via plataforma TagoIO). Pois, o seu modelo de publicação e assinatura permitiu a trasmissão e recepção dos dados de interesse, ou seja, dados temperatura ambiente e umidade do solo de forma agil, prática, com baixo consumo de banda e baixa necessidade de processamento na transmissão das informações a serem enviadas.

O módulo ESP32-WROOM-32 utilizado mostrou-se uma ótima alternativa para a implementação do sistema proposto, onde foi utilizado como cliente de publicação QMTT. Pois o mesmo possui módulo wi-fi de alta performance, tem um baixo consumo de energia e além disso é um disposivito de baixo custo.

A plataforma IoT TagoIO, viabilizou a utilização do serviço como broker MQTT, ou seja, como servidora intermediária de publicação e assinatura, pois a plataforma possibilitou a exibição de dados dos sensores através de gráficos em tempo real, o que viabiliza o monitoramento do plantio caseiro de forma remota.

A plataforma possui muitas ferramentas e recursos, como: inúmeros modelos de gráficos, botões de acionamento, além de ser open source. A TagoIO disponibiliza estas funções mesmo quando utilizada na versão gratuita, como no caso deste projeto. A plataforma é de fácil acessibilidade, assim pode ser utilizada por usuários

com pouco ou nenhum conhecimento sobre suas funções. Os sensores implementados no projeto foram eficientes e com um baixo tempo de la tência na transmissão das informações, funcionando conforme o esperado. O sistema no geral mostrou-se eficiente, principalmente quando a irrigação foi acionada mediante condição de umidade do solo estabelecida, pois aconteceu de forma automatizada e praticamente instatânea a verificação da condição de umidade do solo pelo microcontrolador.

Referências Bibliográficas

[1] OASIS, "MQTT Version 3.1.1 Plus", http://docs.oasis-open.org/mqtt/mqtt/v3.1.1/mqtt-v3.1.1.html, 2015, acessado em 31/10/2021.

[2] ESPRESSIF, "Datasheet Esp32-wroom-32", https://www.espressif.com/sites/default/files/documentation/esp32-wroom-32_datasheet_en.pdf, 2021, acessado em 31/10/2021.

[3] D.R.KIRAN, *Production Planning and Control*. Butterworth-Heinemann: England, 2019.

[4] BNDES, "Cartilha de Cidades", https://www.bndes.gov.br/wps/wcm/connect/site/db27849e-dd37-4fbd-9046-6fda14b53ad0/produto-13 -cartilha-das-cidades-publicada.pdf ?MOD=AJPERES&CVID=m7tz8bf, 2018, acessado em 04/11/2021.

[5] SANTOS, B. P., SILVA, L. A. M., CELES, C. S. F. S., et al., *Internet das Coisas: da Teoria à Prática*. Sociedade Brasileira de Computação, 2016.

[6] AL-FUQAHA, A., GUIZANI, M., MOHAMMADI, M., et al., "Internet of Things: A Survey on Enabling Technologies, Protocols, and Applications", *IEEE*, v. 17, pp. 2–3, 2015.

[7] KHAN, R. K. S. U. K. R. Z. S., "Future Internet: The Internet of Things Architecture, Possible Applications and Key Challenges", *IEEE*, pp. 2–3, 2013.

[8] BNDES, "Internet das coisas: estimando impactos na economia", https://www.bndes.gov.br/wps/portal/site/home/conhecimento/noticias/noticia/internet-coisas-iot, 2017, acessado em 11/11/2021.

[9] FRACTTAL, "9 aplicações práticas mais importantes da Internet das Coisas", https://www.fracttal.com/pt/blog/aplicacoes-praticas-iot, 2020, acessado em 04/11/2021.

[10] FURTADO, T., "O que é wireless?" https://www.techtudo.com.br/noticias/2011/12/o-que-e-wireless.ghtml, 2011, acessado em 04/11/2021.

[11] WU, R., YANG, X., ZHOU, X., et al., *Enterprise Wireless Local Area Network Architectures and Technologies*. CRC Press, 2021.

[12] ENGST, ADAM C.; FLEISHMAN, G., *Kit do Iniciante em Redes sem Fio*. Editora Pearson, 2005.

[13] MENDES, D. R., *Redes de Computadores*. Novatec Editora, 2015.

[14] GOMES, P. C. T., "O que são protocolos de rede?" https://www.opservices.com.br/protocolos-de-rede/, 2019, acessado em 05/11/2021.

[15] HILLAR, G. C., *MQTT Essentials - A Lightweight IoT Protocol*. Packt Publishing, 2017.

[16] MQTT.ORG, "MQTT: The Standard for IoT Messaging", https://mqtt.org/, 2020, acessado em 07/11/2021.

[17] YUAN, M., "Conhecendo o MQTT", https://developer.ibm.com/br/articles/iot-mqtt-why-good-for-iot/, 2017, acessado em 07/11/2021.

[18] HUNKELER, U., TRUONG, H. L., STANFORD-CLARK, A., "MQTT-S — A publish/subscribe protocol for Wireless Sensor Networks", *IEEE*, pp. 2–7, 2008.

[19] ALERIE LAMPKIN, LEONG, W. T., RAWAT, L. O. S., et al., *Building Smarter Planet Solutions with MQTT and IBM WebSphere MQ Telemetry*. IBM REDBOOKS, 2012.

[20] SINGH, M., RAJAN, M. A., SHIVRAJ, V. L., et al., "Secure MQTT for Internet of Things (IoT)", *IEEE*, pp. 2–6, 2015.

[21] UNION-ITU, I. T., "E.800 : Terms and definitions related to quality of service and network performance including dependability", *Telecommunication stardadization sector of ITU*, pp. 3, 1994.

[22] HWANG, H. C., PARK, J., SHON, J. G., "Design and Implementation of a Reliable Message Transmission System Based on MQTT Protocol in IoT", *Wireless Personal Communications*, pp. 3 – 4, 2016.

[23] KOLBAN, N., *Kolban's Book on ESP32*. Elektor International Media BV, 2018.

[24] IBRAHIM, D., *The Complete ESP32 Projects Guide*. Elektor International Media BV, 2019.

[25] HAT, R., "O que é um IDE?" https://www.redhat.com/pt-br/topics/middleware/what-is-ide, 2019, acessado em 07/11/2021.

[26] MATTEDE, H., "O que são sensores e quais as suas aplicações?" https://www.mundodaeletrica.com.br/o-que-sao-sensores-e-quais-as-suas-aplicacoes/, 2014, acessado em 05/11/2021.

[27] GREENGARD, S., *The internet of things*. Mit Press, 2015.

[28] GARRETT, F., "Carro inteligente: veja como funcionam os modelos sem motorista", https://www.techtudo.com.br/noticias/2019/12/carro-inteligente-veja-como-funcionam-os-modelos-sem-motorista.ghtml, 2019, acessado em 05/11/2021.

[29] KWOK WU, F. B., *Collaborative Internet of Things (C-IoT)*. Wiley, 2015.

[30] ASHARI, T. K. P. S. A., "Internet of Things (IoT) of Smart Home: Privacy and Security", *Article in International Journal of Computer Applications*, pp. 3–5, 2019.

[31] INTELBRAS, "Casa inteligente: a IoT no futuro da automação residencial", https://blog.intelbras.com.br/casa-inteligente-a-iot-no-futuro-da-automacao-residencial/, 2018, acessado em 06/11/2021.

[32] PÉREZ-FREIRE, L., WOLFERT, S., VERDOUW, C., et al., *Digitalizando a indústria*. 2016.

[33] MENDES, L. G., "5 formas de aproveitar a Internet das Coisas na agricultura e tornar sua fazenda mais rentável", 2020, acessado em 10/11/2021.

[34] YUN, T., GUAN, L., "Fiducial Point Tracking for Facial Expression using Multiple Particle Filters with Kernel Correlation Analysis". In: *Proceedings of the International Conference on Image Processing*, pp. 373–376, Sept 2010.

[35] RAPP, V., SENECHAL, T., BAILLY, K., et al., "Multiple Kernel Learning SVM and Statistical Validation for Facial Landmark Detection". In: *Proceedings of the International Conference on Automatic Face Gesture Recognition and Workshops*, pp. 265–271, March 2011.

[36] BEHZAD, A., *Wireless LAN Radios*. Wiley-IEEE Press, 2007.

[37] A, A. M. F., *Wireleraújoss Hacking*. Visual Books, 2013.

[38] TELECO, "Redes Wi-Fi: O Padrão IEEE 802.11n", https://www.teleco.com.br/tutoriais/tutorialwifiiee/pagina_4.asp, 2015, acessado em 17/10/2021.

[39] INTEL, "Diferentes protocolos Wi-Fi e taxas de dados", https://www.intel.com.br/content/www/br/pt/support/articles/000005725/wireless/legacy-intel-wireless-products.html, 2021, acessado em 17/10/2021.

[40] FILIPEFLOP, "Esp-Wroom-32", https://www.filipeflop.com/produto/modulo-wifi-esp32-bluetooth/, 2021, acessado em 31/10/2021.

[41] HIERTZ, G. R., STIBOR, L., ZANG, Y., et al., "The IEEE 802.11 Universe", *IEEE Communications Magazine*, v. 48, pp. 62–70, 2010.

[42] COLOMBO, J. F., DE LUCCA FILHO, J., "INTERNET OF THINGS (IoT) AND 4.0 INDUSTRY: revolutionizing the business world", *Revista interface tecnologica*, pp. 10–12, 2018.

[43] DA INDÚSTRIA, C. N., "Desafios para a indústria 4.0 no Brasil", *CNI*, pp. 11, 2016.

[44] SALEH, M. D., HERMINIAWATI, H. MELLAH, N. D. S., "Comparison of Contrast Enhancement Techniques in License Plate Recognition", *IEEE*, 2008.

[45] V.S. PRAZERES, C., "Representação de camadas da arquitetura IoT." https://www.researchgate.net/figure/Representacao-de-camadas-da-arquitetura-IoT-Adaptado-de-Andrade-et-al-2018_fig1_337031883, 2018, acessado em 06/11/2021.

[46] EMBRAPA, "Internet das coisas pode ajudar a melhorar produtividade agrícola", https://www.embrapa.br/busca-de-noticias/-/noticia/31786119/internet-das-coisas-pode-ajudar-a-melhorar-produtividade-agricola, 2018, acessado em 06/11/2021.