Desenvolvimento de um módulo de Internet das Coisas para a realização de micromedição por sensor de vazão

João Victor Bandeira dos Anjos, João Victor Ferreira Gomes, Luiz Fernando Ferreira Messias Ribeiro, Macgyver Cseh dos Santos, Wellington Gomes da Silva e Marcel Stefan Wagner

Departamento de Engenharia da Computação

Universidade Anhembi Morumbi (UAM)

***Resumo* — Este trabalho tem como objetivo desenvolver um módulo de Internet das Coisas para realizar a micromedição. Este dispositivo utilizará um sensor de vazão para realizar a medição dos dados dispostos nos medidores de água, comumente utilizados pelas empresas nacionais de saneamento, após a leitura do microcontrolador os dados serão enviados para uma aplicação web. Para isso, foram consultados outros trabalhos acadêmicos e fontes relacionadas ao assunto e percebeu-se que a medição utilizando recursos humanos é feita em nível nacional e este trabalho propõe um aumento de eficiência neste processo. Para coleta de dados no medidor, será utilizado um microcontrolador de hardware aberto, bastante difundido no mercado, o ESP32 com 30 pinos e um sensor de vazão de efeito hall YF-S201. Acoplado ao controlador será conectado um hidrogerador de energia, o F-50, e para o gerenciamento e regulação da tensão um módulo será utilizado, comumente empregado para carregamento solar que será adaptado para geração hidrelétrica, o CN-3065 com três entradas (uma serial, uma para o hidrogerador e outra para bateria que servirá como *nobreak*). Com isso, espera-se um aumento na eficiência das leituras, evitando medições menores ou maiores, causadas por erros humanos e, além disso, um aumento no controle de consumo por parte dos clientes das Concessionárias de saneamento, uma vez que terão acesso ao seu uso com maior frequência e, portanto, serão identificados os desperdícios desnecessários motivados por vazamentos internos, que hoje são detectados tardiamente.**

***Palavras-Chave* — micromedição, *IoT*, Internet das Coisas, ESP32**

1. INTRODUÇÃO

A população do Estado de São Paulo, em meados de 2022, estava passando por um momento crítico de escassez hídrica e, segundo a Agência Nacional de Águas e Saneamento (ANA) [1], das unidades da Federação, esta era a que possuía a maior sequência de secas consecutivas e, além disso, continha a condição mais crítica em território nacional.

Por determinação da Agência Reguladora do Estado de São Paulo (Arsesp), a Companhia de Saneamento Básico do Estado de São Paulo (Sabesp) fica obrigada a realizar leituras com periodicidade próxima a 30 dias ou, em casos excepcionais, 180 dias [2]. Somando-se a isso, o serviço de leitura é realizado por um humano, que, previamente, necessita de uma rota logística estipulada pela Companhia.

Neste meio tempo, vazamentos internos nas residências dos clientes podem ocorrer e, o usuário só tem a dimensão do problema após a efetivação da leitura. Gerando frustração e possíveis reclamações posteriores, onerando canais de atendimento e ocasionando problemas para a imagem da organização. Além disso, há um agravamento ambiental e social, haja vista que a água potável, pronta para a utilização humana, é desperdiçada e isso é inaceitável, sobretudo, em tempos de escassez hídrica.

Segundo Matos e Conceição [3], a contenção de perdas é mandatória e, ao mesmo tempo, uma oportunidade, pois há um grande espaço para desenvolvimento tecnológico. Tendo isso em mente, as autoras indicam a utilização de medidores inteligentes, pois suas margens de erro costumam ser inferiores e, além disso, eles podem ser lidos remotamente. Basicamente, o processo de leitura acontece em quatro etapas:

1. Medição;
2. Transferência de dados;
3. Processamento e análise; e
4. *feedback* sobre o consumo.

1.1. PROBLEMA DE PESQUISA

Fazendo uma pesquisa prévia à implementação deste estudo, foram identificados alguns problemas na esfera socioambiental que fomentaram a iniciação da pesquisa e prototipação do projeto. Dentre os principais problemas podemos citar:

- Crise hídrica forçando uma otimização do consumo dentro do Estado de São Paulo;
- Desperdício inerente à vazamentos internos; e

- Má utilização de Recursos Humanos nos serviços operacionais.

1.2. OBJETIVOS

Tendo em vista os problemas destacados no tópico 1.1, os objetivos deste trabalho são norteados pela resolução dos problemas identificados.

A. Objetivos Gerais

A água doce potável é um bem escasso e as secas que o Estado de São Paulo enfrenta agravam a situação. Com isso, reduzir as perdas por vazamentos detectados tardiamente é um dos principais objetivos.

A intelectualidade humana é a energia motriz de uma sociedade, tendo isso em mente, outro objetivo que norteia esta pesquisa é a realocação da mão de obra para tarefas que demandam, realmente, uma intervenção humana, ou seja, que demandam raciocínio e resolutividade de problemas complexos.

Para que exista uma gestão e consumo consciente por parte do usuário, as Companhias de saneamento precisam disponibilizar informação de consumo com maior periodicidade e a criação de um protótipo que auxilie neste problema é uma meta que o grupo pretende alcançar.

B. Objetivos Específicos

Os objetivos específicos são correlacionados ao desenvolvimento de um protótipo de *Internet of Things* que realize as micromedições de forma remota e envie os dados para um servidor na nuvem e, posteriormente, demonstre essas informações de modo gráfico, em um *dashboard*, ao consumidor da Companhia de Saneamento.

Ademais, é parte dos objetivos específicos a evolução individual de cada integrante do grupo de pesquisa. Ambos os graduandos pretendem desenvolver características analíticas e melhorar o *soft skills*. Habilidades comunicativas e investigativas também fazem parte dos objetivos específicos almejados.

1.3. ESTRUTURA DO TRABALHO

O trabalho está dividido em 5 etapas:

1. Introdução: trazendo os motivadores e perspectivas do trabalho;
2. Referencial Teórico: com os dados de embasamento técnico e bibliográfico que fomentou os tópicos posteriores;
3. Metodologia: expondo as ações aplicadas no desenvolvimento da escrita e do protótipo;
4. Resultados: neste item serão expostos alguns resultados encontrados com o decorrer da prototipação;
5. Considerações Finais: coligado ao tópico anterior, neste item foi transcrita uma conclusão após a análise de indicadores feita no levantamento dos resultados.

2. PESQUISA TEÓRICA

Para que haja uma perfeita discussão sobre os resultados e conclusões, se faz necessária a fundamentação teórica transcorrida neste tópico.

2.1. INTERNET DAS COISAS

Segundo Magrini [4], não há consenso sobre a definição de Internet das Coisas. Contudo, de maneira geral, ela pode ser definida como um ecossistema de sensores e microcontroladores embutidos em equipamentos, com o intuito de facilitar as tarefas corriqueiras da sociedade. Ainda segundo Magrini [4], a única convergência entre a grande maioria dos autores que discorrem sobre este assunto está vinculada a hiperconectividade de sensores, computadores e objetos, gerando um grande processamento de dados, envolvendo este ambiente ao conceito de *Big Data*.

A. Experiência da Sabesp com a Leitura Remota do Consumo de Água

A Sabesp [24], empresa responsável pelo saneamento básico no estado de São Paulo, no início de 2019 desenvolveu um projeto de leituras por IoT (Internet das Coisas) que visa otimizar o processo de medição do consumo de água dos clientes.

O projeto consiste em instalar dispositivos inteligentes nos hidrômetros, que enviam os dados de leitura para uma plataforma online, onde podem ser acessados pela Sabesp e pelos próprios clientes.

O projeto traz benefícios como a redução de custos operacionais, a melhoria da qualidade do serviço, a prevenção de fraudes e vazamentos, e a conscientização ambiental dos consumidores. O projeto foi implementado em uma fase piloto em algumas regiões do estado e tem previsão de expansão para outras áreas nos próximos anos.

2.2. MQTT - *MESSAGE QUEUING TELEMETRY TRANSPORT*

Este protocolo foi idealizado e implementado pela *International Business Machines Corporation* (IBM) em meados da década de 90 e tinha como foco sistemas com supervisão e aquisição de dados, todavia, o protocolo ganhou grande notoriedade no ramo de Internet das Coisas [5].

O MQTT se destaca por sua simplicidade sem renunciar à segurança e qualidade de serviço. Suas principais vantagens são a baixa demanda de banda e poucas obrigações no que tange o *hardware* [5].

Ele utiliza a infraestrutura e integração com o TCP/IP, além disso se localiza na mesma camada OSI do HTTP, todavia possui algumas diferenciações, como um *payload* menor e um paradigma que permite a comunicação de 1 para N [5].

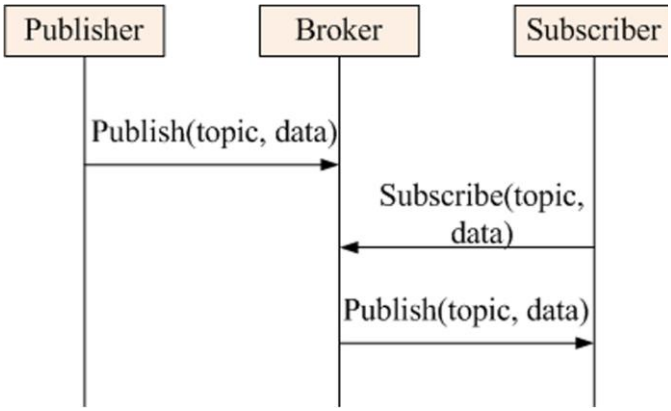

Figura 1 – Paradigma *Publish-Subscribe*.
Fonte: Haripriya, 2019.

A Figura 1 demonstra o funcionamento do paradigma *Publisher-Subscriber*. Diferente do paradigma Request-Reponse, onde o cliente requisita algo e o servidor atenda à solicitação, no *Publisher-Subscriber* o cliente se inscreve em um tópico através de uma requisição ao *Broker* que fará o papel de intermediador entre os publicadores e os inscritos [5]. Como é notado na imagem anterior, uma peculiaridade deste paradigma é a possibilidade de recebimento e publicação por parte do cliente.

2.3. *Eclipse Mosquitto*

O *Eclipse Mosquitto* é um *Broker* de código aberto que implementa versões do protocolo MQTT. Segundo seus desenvolvedores ele é ideal tanto para dispositivos de baixa potência, como a título de exemplo, microcontroladores, como para grandes servidores [20].

2.4. MongoDB

O *MongoDB* é um sistema de gerenciamento de banco de dados não relacional, de código aberto, que utiliza documentos flexíveis em vez de tabelas para processar e armazenar diversos tipos de dados. Ao contrário dos sistemas de gerenciamento de banco de dados relacionais, o *MongoDB* não exige um modelo de dados rígido, proporcionando um ambiente elástico para armazenar e consultar dados variados de maneira eficiente. Isso simplifica o gerenciamento do banco de dados para desenvolvedores e oferece escalabilidade para aplicativos multiplataforma [17].

As unidades fundamentais de dados no *MongoDB* são documentos ou coleções de documentos. Esses documentos, formatados como *Binary* JSON (*JavaScript Object Notation*), podem conter uma variedade de dados e ser distribuídos entre sistemas diversos. Com seu design de esquema dinâmico, o *MongoDB* proporciona flexibilidade excepcional na criação e consulta de registros de dados, permitindo análises eficazes de grandes conjuntos de informações [17].

2.5. *Pymongo*

Pymongo é uma biblioteca Python que facilita a interação com o *MongoDB*, um banco de dados *NoSQL*. Permite conexão, manipulação de bancos de dados, coleções e documentos, além de suportar operações CRUD (*Create, Read, Update, Delete*) e a estrutura de agregação do *MongoDB*. Simplifica a escrita de código Python para executar tarefas comuns no *MongoDB*, como inserção, consulta, contagem e atualização de dados.

2.6. Insegurança Hídrica no Estado de São Paulo

Segundo Castro [8], as regiões metropolitanas, como São Paulo, apresentam baixa segurança hídrica, sobretudo, pela expressiva demanda da população. Somando-se a isso, existe uma requisição volumosa com destino às indústrias do entorno da metrópole.

Agravando a situação de São Paulo, uma parcela da destinação do esgotamento doméstico e dos afluentes industriais acaba sendo, ilegalmente, despejado sem tratamento nos cursos de água. Dessa forma, derivando na poluição dos rios que cortam o Estado, como a título de exemplo podemos citar o Rio Tietê [8].

A Agência Nacional de Águas e Saneamento (ANA) [1] nos expõe que o Estado de São Paulo, em 2022, vinha passando por um momento crítico de escassez hídrica, haja vista que das unidades da Federação ela é a que possui a maior sequência de secas consecutivas. A alta demanda por recursos hídricos e o grande volume de meses de seca colocava São Paulo como o índice mais alarmante na atualidade.

2.7. Perdas Hídricas

Segundo o Sistema Nacional de informações sobre Saneamento (SNIS) [9], as perdas de água podem ser definidas como ineficiências técnicas e são típicas a qualquer sistema de abastecimento de água.

Existem dois tipos de perdas, a primeira é classificada como perda física, aquela que está relacionada ao vazamento, visíveis ou não, no percurso entre a estação de tratamento de água e o local de consumo - imóveis, comércios, indústrias e correlatos. Já a segunda condição de perda pode ser denominada como perda não física, diferentemente da perda física, o consumo ocorre, contudo não é contabilizado pela empresa distribuidora de água. Na perda não física a não contabilização ocorre, especialmente, por conta de fraudes e submedição de hidrômetros [10].

O SNIS [9] menciona que as perdas na região sudeste são de, aproximadamente, 36%, contudo a pior região é a região norte, com uma perda de 56%, aproximadamente.

A Figura 2 mostra que em 2019 nenhum estado conseguiu alcançar a métrica de perdas menores que 20%. Na segunda faixa de 20,1 a 30% dois estados conseguiram, Goiás e Alagoas. Se encontram na faixa de 30,1 a 40%, nove estados (Rio de Janeiro, São Paulo, Espírito Santo, Minas Gerais, Paraná, Santa Catarina, Tocantins, Paraíba e Mato Grosso do Sul) e o Distrito

Federal. Os demais estados estão em situação de desperdício maior que 40%, neste grupo destaca-se o Amapá com 70% de perda [9].

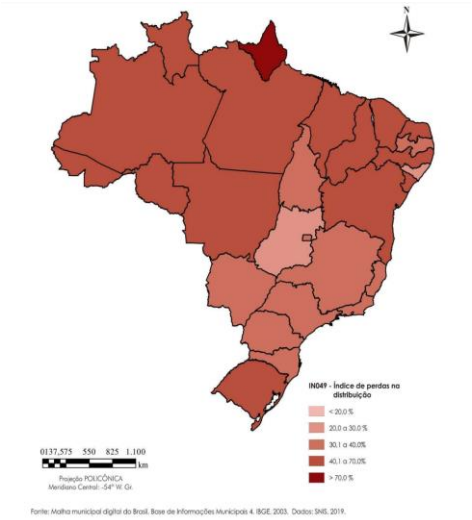

Figura 2 - Mapa do índice de perdas na distribuição (IN049).
Fonte: SNIS, 2019.

2.8. LORAWAN

O ESP32 é um Antes de entendermos o que é o protocolo *LoRaWAN* se faz necessário a introdução do que é a tecnologia *LoRa*. Do acrônimo *Long Range* (Longo Alcance), esta ferramenta é definida por Bertoleti [7] como: "uma tecnologia de radiofrequência que permite comunicações em longas distâncias". Ainda segundo o autor é especificado que a comunicação pode chegar a uma grandeza de quilômetros.

A frequência utilizada para os envios de protocolo é de sub-gigahertz e possui variação conforme as regras regionais de seu país ou continente, a topologia de rede mais utilizada é a estrela e é adequada para a utilização em ambientes rurais e urbanos. Vale ser ressaltado que por possuírem um comprimento de onda reduzido sua taxa de transmissão máxima é diminuída a 37,5kbps, fazendo com que a transferência de dados complexos, como a título de exemplo *streaming*, seja proibitiva [7].

Feita a introdução da tecnologia, podemos explanar sobre o protocolo de rede *LoRaWAN*, ele é aberto permitindo a criação de uma rede completa. De maneira resumida esse padrão implementa os detalhes de qualidade do serviço, funcionamento, segurança e ajuste de potência. Outra característica que difere a *LoRaWAN* da tecnologia *LoRa* é que a primeira demanda por um *gateway* para transmissão dos protocolos para internet, atualmente seus preços não são convidativos [7].

A Figura 3 demonstra como funciona a arquitetura de rede no protocolo *LoRaWAN*.

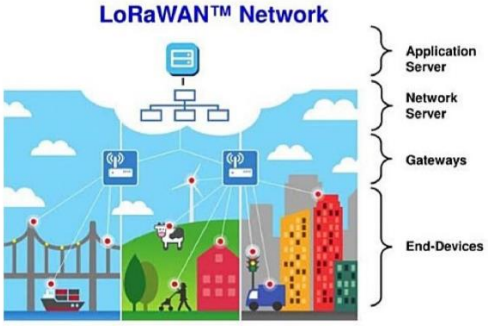

Figura 3 - Arquitetura de rede *LoRaWAN*.
Fonte: Bertoleti, 2019.

2.9. MICROCONTROLADOR ESP32

O ESP32 é um *chip* integrado único que combina as funcionalidades de Wi-Fi e *Bluetooth*, operando na frequência de 2,4 GHz e projetado com a tecnologia de baixa potência. Seu design visa otimizar o desempenho em termos de consumo de energia e rádio frequência, proporcionando robustez, versatilidade e confiabilidade em diversas aplicações e cenários de implementação [12].

Suas principais especificações técnicas são:

Tabela 1 – Especificações Técnicas

COMPONENTES	DESCRIÇÃO
QTDE. DE PINOS:	30
CPU:	Xtensa® Dual-Core 32-bit LX6
ROM:	448 KBytes
RAM:	520 Kbytes
FLASH:	4 MB
CLOCK MÁXIMO:	240MHz
POSSUI:	*Wireless* padrão 802.11 b/g/n
WI-FI:	Conexão Wifi 2.4Ghz (máximo de 150 Mbps)
CONECTOR:	Micro-usb
MODOS DE OPERAÇÃO:	STA/AP/STA+AP
BLUETOOTH:	BLE 4.2
PORTAS GPIO:	25
TENSÃO DE OPERAÇÃO:	4,5 ~ 9V
POSSUI:	Conversor analógico digital (ADC)

Fonte: Própria, 2022.

A Figura 4 demonstra o microcontrolador que será responsável pela gestão dos dados e recursos do projeto.

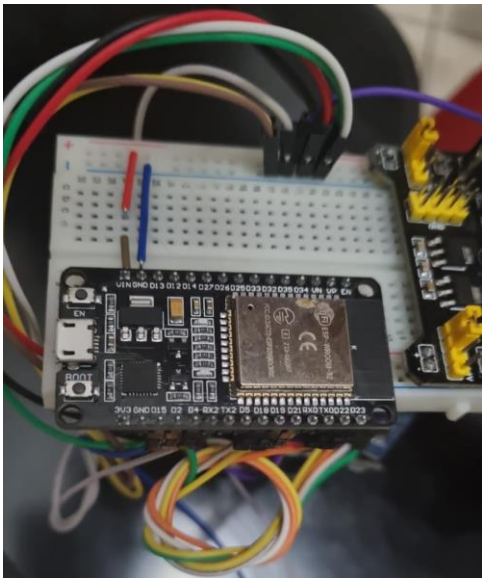

Figura 4 - Microcontrolador ESP32 com 30 pinos.
Fonte: Própria, 2023.

2.10. PLACA *NOBREAK* - *MINI SOLAR LIPO CHARGER* V1.0 - CN3065

Placa acessível, que não demanda programação prévia. O circuito integrado do carregador lida, de maneira autônoma, com o fluxo de energia dos vários componentes. Existe uma gestão entre a porta de entrada de energia solar e a porta de entrada de energia fornecida por baterias [13].

A fabricante, Elecrow [13], alega que a placa regula a tensão de saída de 5V, enquanto a bateria estiver descarregada um LED de luz vermelha é acionado, se a bateria estiver totalmente carregada outro LED é aceso, só que agora com tonalização verde.

Como demonstrado na Figura 5, a placa é composta por três *inputs* duas portas JST 2.0 e outra USB. Como saída temos uma saída USB e outra JST 2.0, ambas de 5V. A carga máxima fornecida é de 1A [13].

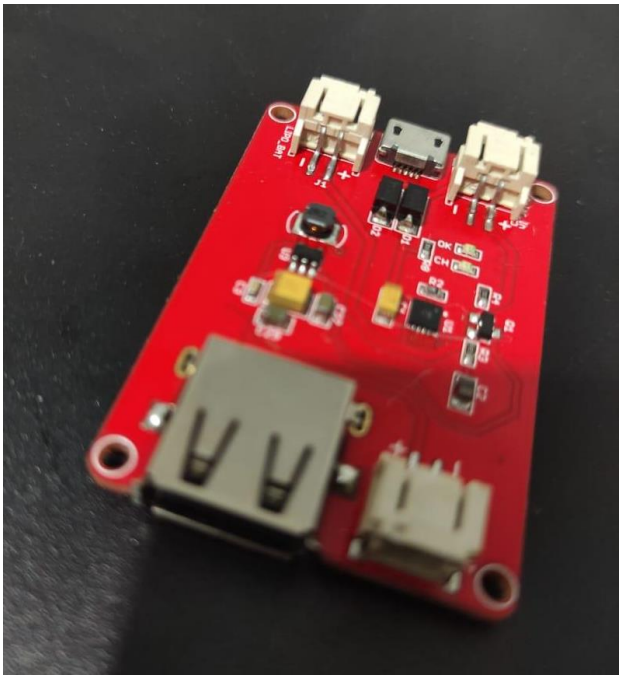

Figura 5 - *Mini solar lipo charger* v1.0 - CN3065.
Fonte: Própria, 2022.

2.11. BATERIA DE LÍTIO ÍON POLÍMERO (LIPO)

Segundo a organização Sistemas e Tecnologia Aplicada [14], o primeiro registro de projeto de baterias de íons de lítio-polímero é datado em 1970 e utilizava um eletrólito seco que possuía características similares a uma película plástica. Este material servia como substituto ao material poroso mergulhado em eletrólitos.

Atualmente, a maioria das baterias com lítio-polímero possuem incorporado um micro separador poroso com alguma umidade, pensando em aumentar a performance em temperatura ambiente. O metal mais comum utilizado como insumo para sua produção é o cobalto [14].

As principais diferenças entre as baterias de íon de lítio normais e as de polímero dizem respeito às baterias de polímeros geralmente são menos espessas, possuem um invólucro flexível e de menor peso (seu formato auxilia no encaixe de aparelhos celulares, por exemplo). Contudo é válido ressaltar que são mais caras para manufaturar, demandam um circuito de controle dedicado e podem inchar se colocadas em alta pressão [14].

Na Figura 6 é demonstrada uma bateria de íon de lítio-polímero.

Figura 6 - Bateria de 3,7V e 2000mAh.
Fonte: Própria, 2022.

2.12. HIDROGERADOR DE ENERGIA - F50

Segunda a fabricante Global Technology [15], a tensão máxima desse hidrogerador de energia é de 12V e sua corrente máxima é de 220mA. Sem o circuito regulador de tensão, sua saída é proporcional à pressão hidráulica. A pressão máxima suportada é de 1,2Mpa.

A imagem abaixo, Figura 7, ilustra a eficiência do hidrogerador conforme a pressão empregada no equipamento. Fica evidente que as variáveis são diretamente proporcionais.

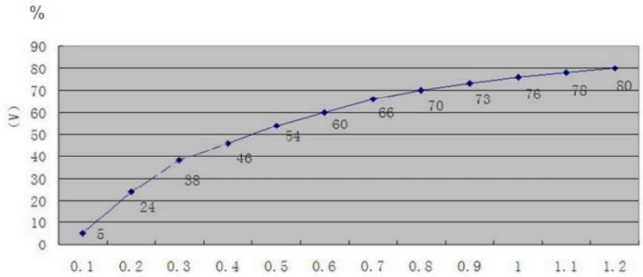

Figura 7 - Desempenho do hidrogerador por pressão (Mpa).
Fonte: Global Technology, 2015.

A Figura 8 demonstra o equipamento gerador de energia elétrica.

Figura 8 - Hidrogerador F-50.
Fonte: Própria, 2022.

2.13. CIRCUITO REGULADOR DE TENSÃO - DFR0379

Este módulo é um *step-down*, isto é, regula sua tensão para baixo e ele foi projetado baseado no regulador de tensão LM2596 de 3A em corrente contínua. A placa suporta de 0 a 40V e contém precisão de 0,05V, variando para cima ou para baixo. Outra característica deste circuito é o seu *display* que serve como voltímetro, lendo a tensão de saída ou entrada. Para alterar a tensão basta girar o potenciômetro de parafuso que é integrado à placa [16].

Segundo o fabricante, DFRobot [16], a taxa de eficiência de conversão é de 88%. A placa possui proteção contra superaquecimento, curto-circuito e inversão de polaridade. Ainda segundo a fabricante, a tensão de entrada deve ser 1,5V maior do que a tensão de saída e o equipamento possui um autocalibrador para fornecer uma saída de alta precisão.

Especificações principais:

- Voltagem de entrada: 4 a 40V;
- Voltagem de saída: 1,25 a 37V;
- Potência de saída: 20W;
- Corrente de saída: 3A;

A Figura 9 ilustra o regulador de tensão que será utilizado no protótipo.

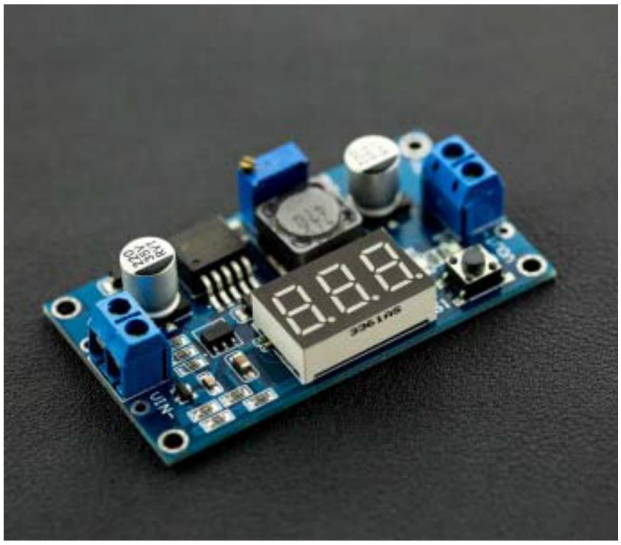

Figura 9 - Regulador de Tensão - DFR0379.
Fonte: DFRobot, 2017.

2.14. CR2032 RECARREGÁVEL

A Figura 10 demonstra a bateria, comumente, utilizada em módulos de *real time clock*. A Tabela 2 explana as especificações técnicas do componente.

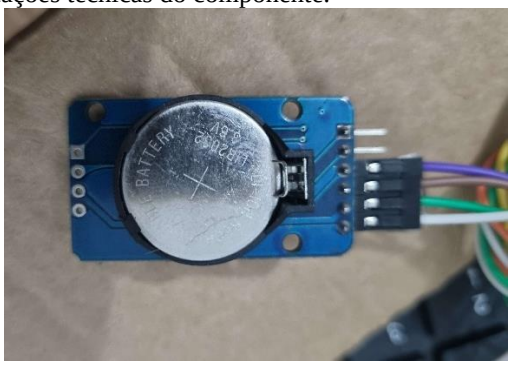

Figura 10 – Bateria Recarregável e Módulo RTC DS3231.
Fonte: Própria, 2023.

Tabela 2 – Especificações Técnicas [22]

COMPONENTES	DESCRIÇÃO
Designação:	ANSI / NEDA-5004LC, IEC-CR2032
Voltagem da Bateria:	3.0 Volts
Peso Médio:	3.3 gramas (0.12 oz.)
Volume:	1.0 centímetros cúbicos (0.06 polegadas cúbicas)
Capacidade Média:	225 mAh a 2.0 volts (Classificado a 10 mil ohms continuamente a 21ºC)
Corrente Máxima de Carga Reversa:	1 microampere
Densidade de Energia:	198 milliwatt hr/g, 653 milliwatt hr/cc

Fonte: Própria, 2023.

2.15. *REAL TIME CLOCK* (RTC) DS3231

O *Real Time Clock* (RTC) DS3231 é reconhecido por sua notável precisão e eficiência energética como um relógio de tempo real. Equipado com um sensor de temperatura integrado e um cristal oscilador, este dispositivo destaca-se pela sua capacidade de manter o tempo com alta precisão, muitas vezes na ordem de poucos segundos por ano [21].

A presença do sensor de temperatura permite não apenas a medição precisa do tempo, mas também a monitorização das condições ambientais. O cristal oscilador de temperatura compensada (TCXO) contribui significativamente para a melhoria da precisão do relógio, especialmente em diversas condições térmicas [21].

Além disso, o DS3231 utiliza o protocolo de comunicação I2C para interagir com outros dispositivos, facilitando a integração em uma variedade de projetos eletrônicos. A inclusão de uma bateria de reserva é fundamental, garantindo que o relógio continue operando com precisão, mesmo durante falhas de energia, proporcionando estabilidade e confiabilidade.

Comumente empregado em aplicações que demandam medição precisa do tempo, como dispositivos de registro de dados e sistemas de controle de acesso, o DS3231 desempenha um papel crucial em assegurar a continuidade da contagem precisa do tempo, mesmo em situações de falta temporária de energia [21].

2.16. DHT22 – SENSOR DE TEMPERATURA E UMIDADE

O módulo DHT22, também conhecido como AM2302, é um sensor digital de temperatura e umidade. Ele é usado para medir a temperatura no intervalo de -40 a 80 graus Celsius e a umidade relativa do ar de 0 a 100%. O sensor possui uma precisão de ±0.5 graus Celsius para temperatura e ±2% para umidade. A resolução é de 0.1 graus Celsius para temperatura e 0.1% para umidade. O módulo opera com uma única leitura a cada 2 segundos, e sua alimentação varia de 3.3V a 6V. A comunicação com o sensor é unidirecional, feita através de um único pino de dados. O DHT22 é comumente utilizado em projetos de IoT, automação residencial e outras aplicações que exigem monitoramento preciso de temperatura e umidade [6]. A Figura 11 ilustra o elemento citado.

Figura 11 – Sensor Medidor Temperatura e Umidade.
Fonte: Própria, 2023.

2.17. SENSOR DE VAZÃO YF-S201

O YF-S201, um sensor de fluxo de água, é composto por um corpo de válvula de plástico, um rotor de água e um sensor de efeito Hall. Quando a água percorre o rotor, ele entra em rotação. A velocidade desse rotor varia de acordo com a taxa de fluxo. O sensor de efeito Hall gera um sinal de pulso correspondente [23].

Principais características:

- Instalação fácil e compacta;
- Sensor de efeito Hall de alta qualidade;

Especificações:

- Tensão de operação: DC 4,5V a 24V (tensão típica: DC 5V a 18V);
- Corrente máxima de operação: 15 mA (DC 5V);
- Faixa de taxa de fluxo: 1 a 30 L/min;
- Temperatura operacional: até 80°C;
- Temperatura do líquido: até 120°C;
- Pressão permitida: até 1,75 MPa.

Conexões elétricas:

- Vermelho: Positivo;
- Preto: Terra (GND);
- Amarelo: Sinal de saída.

Este sensor é amplamente empregado em aquecedores de água, máquinas de venda automática, dispositivos de medição de fluxo e diversas outras aplicações [23].

Abaixo será exposta uma demonstração, na Figura 12.

Figura 12 – Sensor de Efeito Hall.
Fonte: Própria, 2023.

2.18. MÓDULO HW-125 PARA SDCARD

Esse módulo é um dispositivo que permite ler e gravar dados em um cartão SD usando um microcontrolador. Ele pode ser usado para armazenar arquivos de áudio, vídeo, texto ou outros tipos de dados. Ele se conecta ao Arduino por meio dos pinos SPI, que são usados para a comunicação serial entre o microcontrolador e o cartão SD [11].

Algumas características do módulo leitor de cartão SD são:

- Ele facilita a integração do aplicativo SD, tornando-o mais simples;
- Ele se conecta facilmente como um periférico para o módulo de proteção do sensor Arduino;
- Ele suporta entrada de 5V/3.3V, o que é compatível com a maioria dos modelos de Arduino.

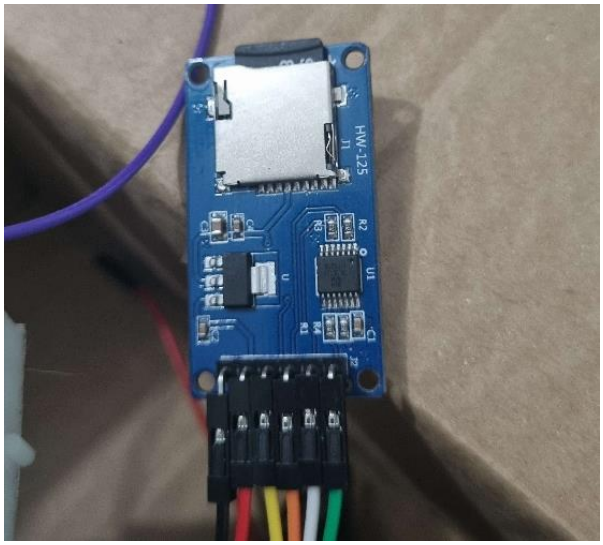

Figura 13 – HW125 Slot Leitor Cartão Sd.
Fonte: Própria, 2023.

3. METODOLOGIA

Neste trabalho algumas metodologias foram utilizadas para o seu desenvolvimento. A primeira a ser destacada é fundamentação teórica, mandatória em trabalhos científicos, com dados referenciais. Além disso, para a realização da documentação e prototipagem, ou seja, criação do mínimo produto viável, será utilizada a metodologia ágil, mais comumente conhecida como *Scrum*.

3.1. FLUXO DA METODOLOGIA

A Figura 14 demonstra o fluxo do processo metodológico implementado nesta pesquisa.

Em um primeiro momento houve uma etapa de reflexão sobre a problemática social que o trabalho se proporia a auxiliar. Neste período foram identificados os problemas descritos na introdução deste artigo.

Em seguida foram levantados alguns dados referenciais, onde são destacadas informações que embasaram o desenvolvimento do protótipo. Nestas referências foram coletados textos em trabalhos acadêmicos, livros, sites e *datasheets* dos principais equipamentos que serão utilizados na etapa de prototipação. Junto deste levantamento foram feitas as compras dos materiais que serão empregados no protótipo. Com os registros dos custos é plausível comparar se a solução encontrada é satisfatória se confrontado com outros *players* do mercado.

Na etapa de prototipação foram utilizados os *hardwares* adquiridos em fases anteriores.

Após o desenvolvimento do equipamento e posterior testagem de eficácia, será concluído os pontos de melhoria e se existe a possibilidade de uma produção em escala.

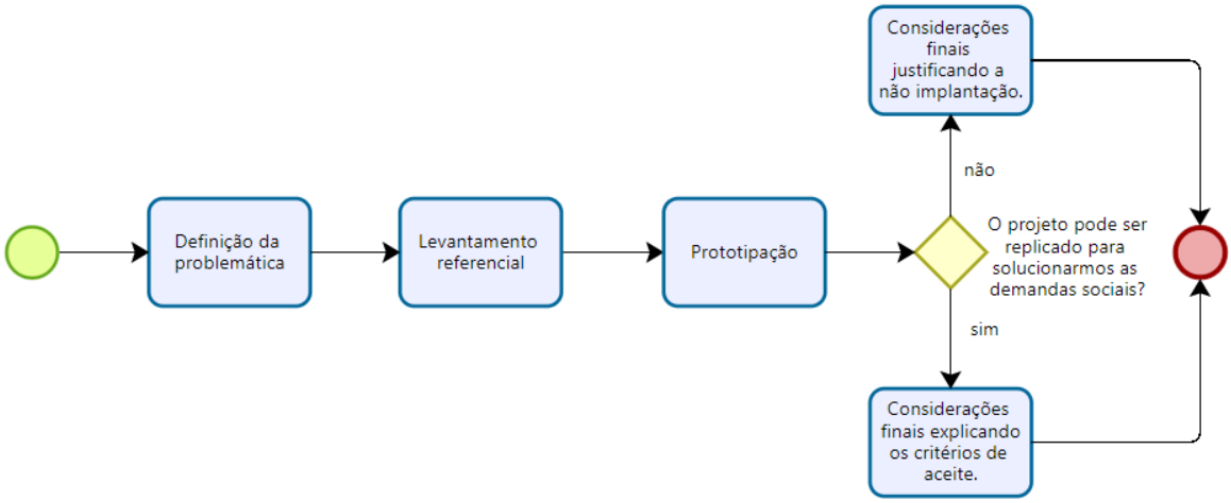

Figura 14 - Fluxograma do processo metodológico.
Fonte: Própria, 2022.

4. DESENVOLVIMENTO

Nesta etapa será demonstrado o desenvolvimento da prototipação do módulo de Internet das Coisas.

4.1. DESCRIÇÃO DA PROTOTIPAÇÃO

Na Tabela 3 é elencado o modelo IPO, do inglês *input process output*, em tradução literal: entrada, processo e saída. Em todas as colunas foram exibidos elementos tangíveis (*hardware*) e elementos intangíveis (informações digitalizadas).

No primeiro momento coletamos as informações oriundas dos sensores, DHT22 e YF-S201, e do *Real Time Clock*,

DS3231. Com a coleta dessa informação o microcontrolador envia uma *string* com todos os dados para a memória secundária. Fazendo que, com isso, haja um *backup* na hipótese de falha de comunicação entre o ESP32 e o servidor de dados do *subscriber*.

O protocolo de comunicação utilizado é o MQTT, e o *payload* é transportado até o servidor através de uma comunicação WI-FI de 2.4 GHz. Os dados são enviados a cada 20 minutos para o *broker*.

Como parâmetros do protocolo de comunicação foi configurado que os dados de *payload* sejam retidos, isto é, assim que o cliente se inscreve no tópico denominado "topicoTodosOsItens" ele recebe as informações enviadas pelo microcontrolador. Após o recebimento são enviadas, com a utilização de uma *API* (*Application Programming Interface*), as informações tratadas para uma interface gráfica com informações de consumo, temperatura e umidade que podem ser visualizadas por um *dashboard*.

Tabela 3 - Modelo *IPO* do projeto.

Input	*Process*	*Output*
• Bateria; • Módulo regulador de tensão; • Módulo de gestão de energia; • Câmera; • Energia elétrica gerada pelo gerador ou armazenada na bateria; • Pulso elétrico ocasionado pela vazão hídrica; • Informação de temperatura e umidade.	• Microcontrolador; • Dados de leitura.	• Circuitos gerenciadores de rede; • Informação de leitura (dado trabalhado).

Fonte: Própria, 2022.

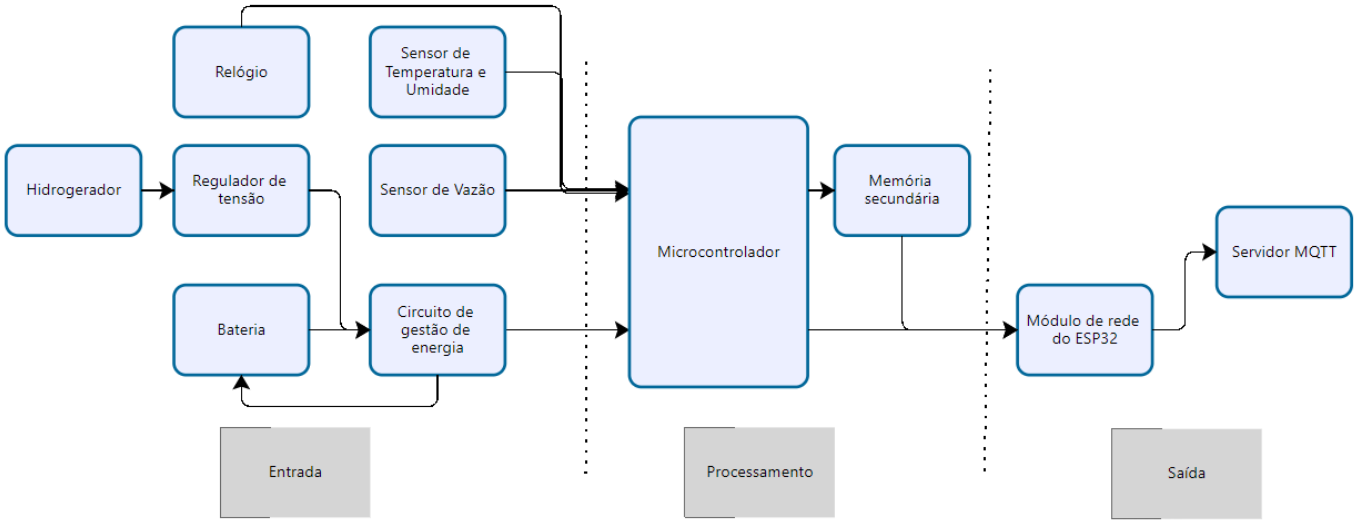

Figura 15 – Esquema físico do protótipo.
Fonte: Própria, 2022.

A arquitetura do hardware do projeto foi ilustrada na Figura 15, com ela fica ainda mais nítido o que foi descrito na Tabela 1, entretanto, o complemento de entendimento se correlaciona apenas no aspecto físico.

O *input* é caracterizado por: entrada de energia produzida pelo hidrogerador e, o excedente, é armazenado na bateria. Para a coleta de informação são utilizados dois sensores o DHT22, para a mensuração de umidade e temperatura, e o YF-S201, um sensor de efeito Hall que é utilizado para a estimativa da vazão de água. Outra informação relevante que é coletada é o dia e a hora em que medição é feita, isso auxilia em uma análise temporal e, além disso, possibilita que se possa fazer correlações entre sazonalidade, temperatura, umidade e consumo hídrico, ajudando em estudos comportamentais para a Companhia de Saneamento.

O *process* é identificado por: Microcontrolador ESP32 e

memória secundária, isto é, cartão TF de 32G.

No *output* temos o módulo de rede internamente acoplado à placa do ESP32 que envia as informações processadas para um servidor de dados.

Na figura 16 é evidenciado o protótipo montado com todos os elementos descritos na imagem anterior, Figura 13.

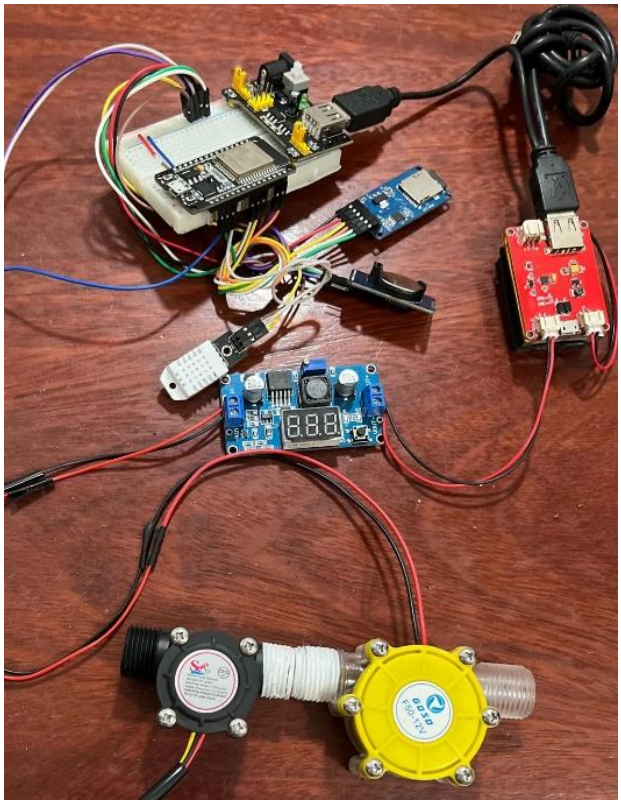

Figura 16 – Protótipo.
Fonte: Própria, 2023.

4.2. CÓDIGO DO MICROCONTROLADOR

Para o desenvolvimento do código utilizado no microcontrolador, por se tratar de um *software* especializado nessa área, foi feito uso da IDE – *Integrated Development Environment* ou em português Ambiente de Desenvolvimento Integrado – Arduino.

A linguagem utilizada foi o C++, haja vista que a IDE que utilizamos tem como base essa linguagem para o desenvolvimento de softwares para sistemas embarcados [25], existe uma vasta possibilidade de pesquisa de documentação de projetistas, facilitando nas modificações e adaptações que foram feitas no projeto deste trabalho.

As bibliotecas utilizadas na aplicação do sistema embarcado foram:

- FS.h: Essa biblioteca permite manipular o sistema de arquivos do ESP32, que é uma área de armazenamento persistente na memória flash. Você pode criar, ler, escrever e apagar arquivos usando essa biblioteca;
- SD.h: Essa biblioteca permite usar um cartão SD com o ESP32, para armazenar dados adicionais. Você pode usar os mesmos métodos da biblioteca FS.h para acessar os arquivos no cartão SD;
- SPI.h: Essa biblioteca permite usar o protocolo SPI (*Serial Peripheral Interface*) para se comunicar com dispositivos externos, como sensores, displays e módulos de memória. O SPI é um protocolo síncrono, que usa quatro pinos: MOSI, MISO, SCK e CS3;
- RTClib.h: Essa biblioteca permite usar um módulo RTC (*Real Time Clock*) para manter a data e a hora atualizadas, mesmo quando o ESP32 está desligado. Você pode usar essa biblioteca para ler e ajustar a data e a hora, bem como para obter informações sobre o dia da semana, o mês e o ano;
- Wire.h: Essa biblioteca permite usar o protocolo I2C (*Inter-Integrated Circuit*) para se comunicar com dispositivos externos, como sensores, displays e módulos de memória. O I2C é um protocolo assíncrono, que usa dois pinos: SDA e SCL;
- Adafruit_Sensor.h: Essa biblioteca é uma interface comum para vários sensores da Adafruit, como acelerômetros, giroscópios, magnetômetros, sensores de luz, temperatura, umidade, pressão e outros. Ela permite obter os dados dos sensores de forma padronizada e consistente;
- DHT.h e DHT_U.h: Essas bibliotecas permitem usar sensores de temperatura e umidade da família DHT, como o DHT11, o DHT22 e o AM2302. Elas permitem ler os valores de temperatura e umidade, bem como calcular o ponto de orvalho e o índice de calor;
- WiFi.h: Essa biblioteca permite usar o módulo WiFi integrado do ESP32, para se conectar a redes sem fio e acessar a Internet. Você pode usar essa biblioteca para configurar o ESP32 como um cliente WiFi, um ponto de acesso WiFi ou ambos;
- PubSubClient.h: Essa biblioteca permite usar o protocolo MQTT (*Message Queuing Telemetry Transport*) para se comunicar com outros dispositivos ou servidores usando o WiFi.

O pino GPIO 15 foi o pino selecionado para capturar os pulsos gerados pelo sensor de efeito Hall, quando ele é acionado é executada uma interrupção para a coleta e acúmulo dos pulsos, armazenados na variável “pulsoAcumulado”. Após vinte minutos é armazenado no SD os dados de data, hora, dia da semana, temperatura, umidade e vazão. Após o armazenamento na memória secundaria é acionada uma reconexão com o servidor MQTT da Eclipse, e é publicado em seus respectivos tópicos os valores coletados.

4.3. DESENVOLVIMENTO *BACK-END* PYTHON

Para o desenvolvimento do *back-end* foi utilizado a linguagem Python, pois tem uma sintaxe simples e fácil de ler. Isso facilita a compreensão do código, o que é benéfico tanto

para iniciantes quanto para desenvolvedores experientes. Segundo a Amazon [26], a linguagem possuí uma rica coleção de bibliotecas e *frameworks* que simplificam tarefas comuns de programação e é amplamente utilizado no desenvolvimento de software empresarial e é uma escolha popular para desenvolvimento rápido de protótipos devido à sua facilidade de uso e velocidade de desenvolvimento. Para efetuarmos o desenvolvimento foram criadas 2 aplicações.

A. Comunicação com o *Broker - Subscriber*

Na primeira aplicação, denominada "*tcc-mqtt-subscriber*", foi empregada a biblioteca *paho-mqtt*, reconhecida por sua eficácia na conexão com brokers. Através dela, possibilitou-se o recebimento de mensagens no evento "*on_message*", a inscrição em tópicos no evento "*on_subscribe*" e a verificação do estado de conexão no evento "*on_connect*", entre outras funcionalidades essenciais. Adicionalmente, integrando o banco de dados não relacional *MongoDB* com a biblioteca *pymongo* em Python, realizou-se o armazenamento das informações provenientes do broker.

Antes do registro no banco de dados, implementou-se uma etapa de tratamento abrangente. As informações são recebidas em uma única *string*, delimitadas por ponto e vírgula, sendo essencial a aplicação da função "*split*" para subdividir esses elementos. Em seguida, utilizaram-se esses fragmentos para reconstruir um dicionário, realizar os tratamentos necessários e, por fim, proceder ao armazenamento no banco de dados. Destaca-se a criação da função "consumo_diario" dentro do evento "*on_message*", responsável por analisar e calcular a diferença entre o consumo no dia atual e no dia anterior, este cálculo é feito através de um pipeline de agregação utilizando os seguintes componentes:

- $*match*: Este estágio filtra documentos com base em determinados critérios. Neste caso, os documentos são filtrados com base na condição em que o campo 'data' deve ser maior ou igual a “dia_anterior” e menor que “dia_atual”.
- $*sort*: Este estágio ordena os documentos com base no campo 'data' de forma descendente (-1), ou seja, do mais recente para o mais antigo.
- $*group*: Este estágio agrupa os documentos com base em um critério específico. No exemplo, os documentos são agrupados pelo dia da data (formatada como '%d-%m-%Y') e, para cada grupo, é mantido apenas o último valor de “vazao_litro_acumulada”.
- $*project*: Este estágio projeta os campos desejados no resultado final. Neste caso, o resultado final contém apenas os campos 'dia' (extraído do campo “_id.dia”) e “vazao_litro_acumulada”, enquanto o campo “_id” é removido.

Essa métrica de consumo diário é armazenada no banco para posterior recuperação pela aplicação "*tcc-mqtt-api*".

B. *API REST*

Na segunda aplicação, denominada "*tcc-mqtt-api*", também implementada em Python, faz-se uso da biblioteca *FastAPI*, reconhecida por sua eficiência na construção de *APIs REST*. Com ela, tornou-se possível listar as informações previamente armazenadas no banco de dados. As informações são apresentadas no arquivo "*project.py*", onde é possível realizar edições e ajustar os dados na exibição.

No serviço ("*service*"), incorporou-se uma breve validação de data para permitir consultas por datas específicas, proporcionando flexibilidade na busca. Por meio da inclusão do parâmetro "*match*" na consulta *pymongo*, conseguiu-se filtrar as informações conforme as datas desejadas. No repositório ("*repository*"), realizou-se a configuração completa de paginação, estabelecendo limites por página e ordenando os resultados. Utilizou-se a função " *aggregate* " para recuperar as informações conforme os filtros aplicados anteriormente. Isso permite a visualização de todos os registros, independentemente de estarem filtrados ou não.

É importante ressaltar que é utilizado na api os conceitos do padrão chamado DDD (*Domain Driven Design*), onde:

- *Controller*: Em uma perspectiva DDD, os *controllers* geralmente não fazem parte da linguagem do domínio, pois são mais voltados para a interação com o usuário. No entanto, eles podem servir como ponte entre a interface do usuário e a lógica de domínio.
- *Service*: O diretório *service* sugere a presença de lógica de aplicação ou de domínio, o que é consistente com a abordagem DDD. Serviços geralmente encapsulam operações relacionadas ao domínio que não pertencem naturalmente a uma entidade específica.
- DTO (*Data Transfer Object*): Em DDD, DTOs podem ser usados para transferir dados entre camadas, mas geralmente estão mais associados à camada de aplicação do que à camada de domínio, no caso desta aplicação, utilizamos para validação dos dados de entrada.
- DAO (*Data Access Object*): Em DDD, o acesso aos dados é geralmente tratado no contexto de repositórios, mas a presença de um diretório dao pode indicar uma abordagem mais pragmática para o acesso a dados, nessa estrutura são utilizados para conexão com o banco de dados e tratamento dos dados de saída.
- *Repository*: O diretório *repository* está alinhado com o conceito de repositórios em DDD, que são responsáveis por abstrair o acesso a dados e fornece métodos para recuperar e persistir entidades do domínio.

4.4. FRONT-END REACT

Para aprimorar o controle e a rastreabilidade dos dados coletados pelo ESP32, o *framework React* na versão 18.2.0 foi selecionado, em conjunto com *Typescript* e o framework SASS para estilização, para desenvolver o dashboard.

- Utilização de *Typescript*: A escolha do *Typescript* se deve à sua natureza "tipada", proporcionando maior qualidade ao código e reduzindo erros no tratamento de dados. A longevidade e facilidade de manutenção na linguagem web foram fatores determinantes na decisão, diferenciando-a do Javascript.
- *Framework React*: A adoção do framework *React* se justifica por sua posição proeminente no mercado, amplamente utilizado por grandes players. Isso proporciona facilidade no desenvolvimento, com amplo suporte e disponibilidade de bibliotecas, agilizando a criação de novos componentes.
- Biblioteca "*react-apexcharts*" para *Dashboards*: Para o desenvolvimento dos *dashboards*, implementamos a biblioteca "*react-apexcharts*". Essa escolha se fundamenta em sua simplicidade, boa manutenibilidade e suporte consistente por parte de seus criadores, garantindo a escalabilidade das telas.
- Utilização de *SASS* em vez de *CSS*: Optamos pelo uso do *SASS* devido à sua abordagem mais amigável. *SASS* transforma o CSS em algo mais próximo de uma linguagem de programação, acelerando o desenvolvimento por meio da criação de variáveis, classes e escopo para evitar interferências indesejadas.
- Material *UI* e *Heroicons* para *Front-End*: No *Front-End*, priorizamos um *layout* eficiente para proporcionar uma experiência de usuário superior. Para isso, integramos a biblioteca de componentes do Material *UI*, uma das maiores e constantemente atualizadas. Complementarmente, utilizamos os ícones da biblioteca *Heroicons*.
- Consumo de Dados da *API* com *Axios*: Para o consumo de dados da *API*, adotamos o *Axios*, responsável por realizar chamadas de métodos *HTTPs*, estabelecendo a conexão eficiente entre a API e o *Front-End*.
- Apresentação da Tela Principal: A tela principal exibe o dashboard, apresentando informações de consumo dos clientes de forma mensal ou semanal. Adicionalmente, são destacadas informações relevantes, como o consumo total do hidrômetro e o valor mensal em R$. Essa interface representa a centralização das principais métricas para uma visualização abrangente e intuitiva.

Na Figura 17 será demonstrado todo o processo, ponta a ponta do projeto, desde o microcontrolador até chegar ao cliente final:

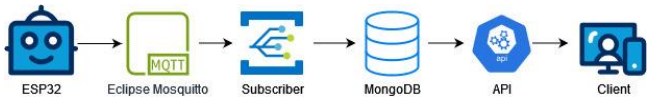

Figura 17 – Macro esquema do projeto.
Fonte: Própria, 2023.

5. RESULTADOS

Neste item será quantificado e descrito os valores encontrados posteriormente à prototipação e análises realizadas no processo metodológico do artigo.

5.1. LOGO DO PROJETO

O logo do projeto será exposto na Figura 18. Ele trás a gravura do texto *Aqua Monitor* com um design que mistura tons com as colorações azul e branco, cores escolhidas por possuírem uma assimilação lúdica com a água. Sobre as letras "ua" existe um desenho de uma onda, que tem como significado as ondas desenhadas na superfície aquática e ao mesmo tempo as ondas de rádio que são utilizadas nas tecnologias de telecomunicações.

Figura 18 – Logo da marca fictícia.
Fonte: Própria, 2023.

5.2. MEDIÇÃO DE DIFERENÇA DE POTENCIAL DO HIDROGERADOR

Houve um teste para a mensuração da capacidade do hidrogerador, para garantir que medição se aproxime da realidade o F-50 foi acoplado a uma torneira diretamente ligada ao cavalete da companhia de saneamento.

Utilizando o voltímetro e o painel do circuito regulador de tensão, podemos notar que a tensão de entrada é equivalente a 5,2V. A Figura 18 evidência este valor, lembrando que o painel está invertido.

Figura 19 – Tensão de entrada (painel invertido).
Fonte: Própria, 2023.

A diferença de potencial (DDP) alcançada na saída foi de 4,5V. Ilustrada na Figura 19.

Figura 20 – Tensão de saída (painel invertido).
Fonte: Própria, 2023.

Na Figura 20 podemos notar como foi realizado o teste de DDP como um todo. Ela demonstra que foi acoplada uma unidade extensora entre a torneira e o hidrogerador e evidência que a tesão mensura é derivada da energia mecânica emprega no F-50. É de grande valia evidenciarmos que a iluminação do diodo emissor de luz da placa *Mini solar lipo charger v1.0* está com a tonalidade vermelha, conforme indicado pelo fabricante, demonstrando que a bateria de 2mAh está sendo carregada naquele momento.

Figura 21 – Evidência do esquema da DDP (painel invertido).
Fonte: Própria, 2023.

5.3. MEDIÇÃO DE CORRENTE DEMANDADA PELO PROTÓTIPO

Nesse tópico será exposta a evidência da leitura da corrente consumida no protótipo. Isso terá grande utilidade no dimensionamento da capacidade de armazenamento de energia da bateria de polímero de lítio que deverá ser utilizada no equipamento final, isto é, no equipamento que poderá ir para a manufatura.

A Figura 22 ilustra a medição de corrente feita com um multímetro. O aparelho foi configurado para leitura com capacidade máxima de 200mA e foi notado que o consumo do protótipo é de, aproximadamente, 74 mA

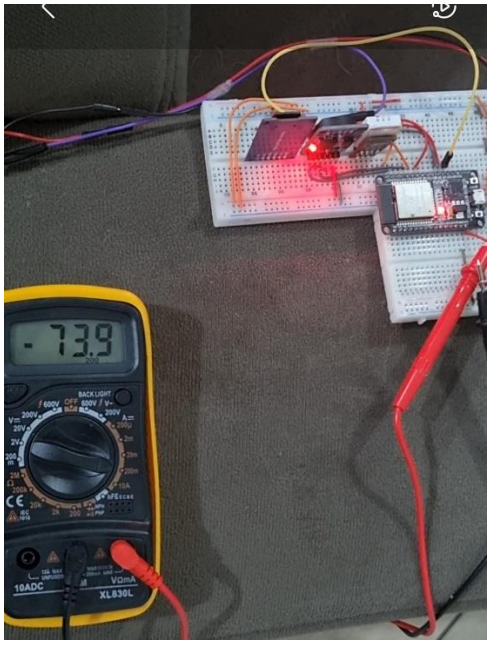

Figura 22 – Corrente consumida no protótipo.
Fonte: Própria, 2023.

É notório que para a duração de um dia sem alimentação de energia, se faz imperativa a utilização de uma bateria de aproximadamente 1.776mAh (74mA*24h).

5.4. TESTE EM CAMPO

Neste item serão descritas as capturas e demonstrações dos dados realizados no teste de hardware e software.

A. Medição pelo ESP32

Para o teste em campo foram efetuadas algumas leituras oriundas, sobretudo, dos dois sensores, o DHT22 e do YF-S201.

Os dados de último pulso acumulado, variável importantes para o cálculo da quantia de vazão em litros acumulada, é sempre buscada na memória secundária após a reinicialização do equipamento. De forma similar, os dados de data e hora são acrescidos graças ao DS3231, que resgatam valores salvos no módulo.

Para o teste que evidência o funcionamento colocamos leituras de 10 em 10 segundos, todavia para o funcionamento cotidiano, o código realizará as leituras com periodicidade contendo o espaçamento de 20 minutos, isto é, três envios de informação por hora.

A Figura 21 ilustra o monitor serial fazendo a leitura das informações que os sensores enviam, essas informações, posteriormente serão entregues a um servidor MQTT disponibilizado pela *Eclipse Foundation*, e, na sequência, uma API salvará essas informações em banco de dados *NoSQL*, conforme o item B do tópico 5.2.

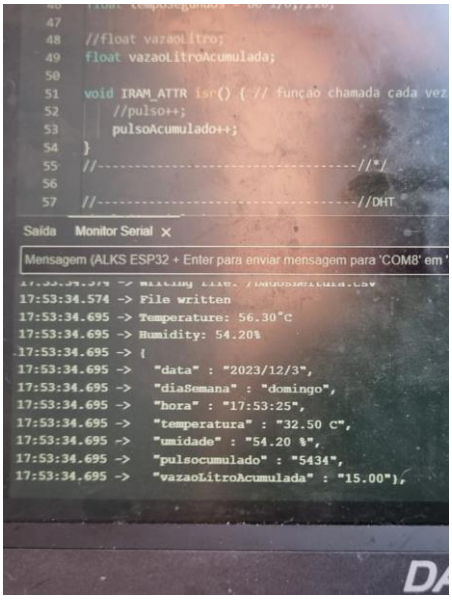

Figura 23 – Monitor Serial.
Fonte: Própria, 2023.

O esquema físico que está fazendo a medição será demonstrado na Figura 22.

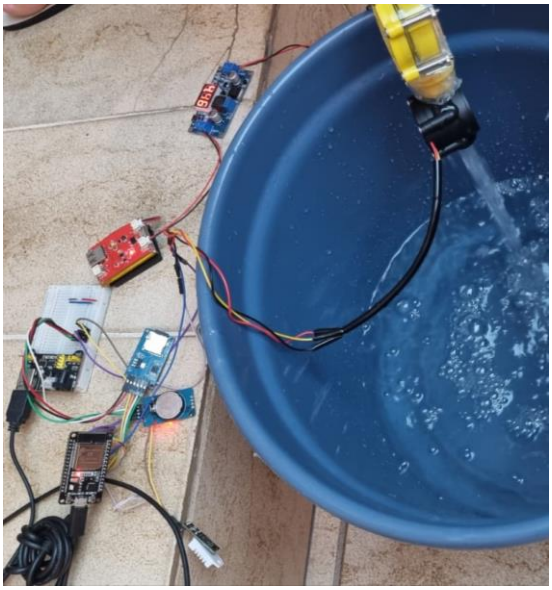

Figura 24 – Esquema Físico da Medição.
Fonte: Própria, 2023.

B. Medição pela API

Na figura 23 está exemplificando a medição recebida pela aplicação e enviada para o banco de dados.

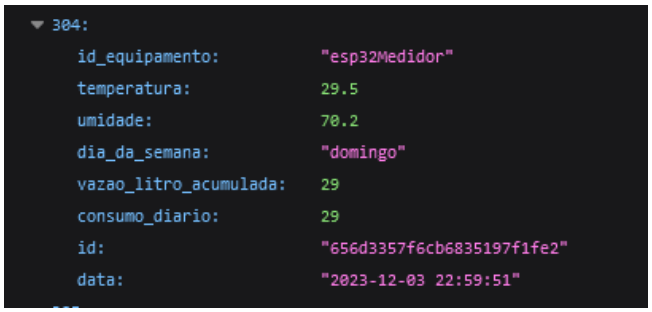

Figura 25 – Evidência do JSON recebido pela aplicação
Fonte: Própria, 2023.

Este resultado é salvo no banco de dados e enviado para que o *client* possa consumi-lo e apresentá-lo em tela.

C. Dashboard

Na figura 24 está apresentando o *dashboard* que mostra um gráfico de consumo total x dia comparando com uma média de todos os clientes cadastrados com a vazão total de determinado cliente. Ao lado é mostrado dois gráficos menores que demonstram temperatura e umidade no ambiente em que o hardware está instalado. Acima do *dashboard* foram utilizados blocos para demonstrar dados importantes, como o consumo total em litros, consumo total em metros cúbicos, consumo mensal em litros e valos mensal em reais.

Figura 26 – *Dashboard*
Fonte: Própria, 2023.

A Figura 25 apresenta a segunda parte da tela principal, exibindo uma tabela com cada leitura realizada pela aplicação. Cada entrada na tabela representa a temperatura, o horário e o nome do equipamento onde os dados foram analisados.

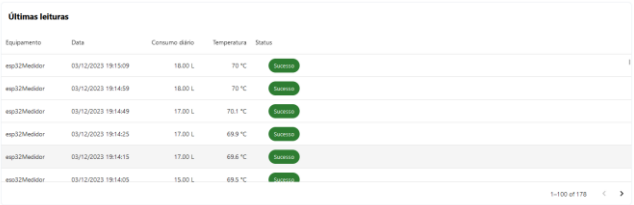

Figura 27 – Tabela de dados
Fonte: Própria, 2023.

5.5. CUSTOS E COMPARAÇÕES

Foram elencados os custos de logística e de aquisição do *hardware* para o desenvolvimento do protótipo. Essas informações foram dissertadas na Tabela 4.

O levantamento foi feito de forma simplória, com o intuito de dimensionar, superficialmente, o valor que seria gasto com a industrialização do projeto. Como ponto de atenção é mandatório explicitar que existe a possibilidade de redução deste valor, tendo em vista a economia de escala que pode ser empregada na fabricação. Além disso, os materiais adquiridos podem ser redimensionados para que se encontre um ponto de equilíbrio que intersecciona a curva de custo e benefício de forma mais eficiente.

Tabela 4 - Custo de implantação do protótipo - segundo semestre de 2022.

Equipamento	Valor (R$)
Microcontrolador Esp32	34,20
Sensor De Fluxo/vazão Água 1/2 Yf-s201 Arduino Efeito Hall	38,00
Modulo Leitor Cartão Micro Sd Card (leitura/escrita) ArduinoTime	15,00
Bateria Lir2032 Cr2032 Recarregavel Li-ion 3,6v Oferta	35,00
Modulo Ds3231 At24c32 I2c Real Time Clock Rtc Ultra Preciso	35,99
Módulo Sensor De Umidade E Temperatura Am2302 Dht22 Arduino	35,15
Placa nobreak - Mini solar lipo charger v1.0 - CN3065	84,50
Bateria de Lítio Íon Polímero (LiPo) - 2.000 mAh	64,34
Hidrogerador de Energia - F50	40,53
Circuito Regulador de Tensão - DFR0379	27,00
Total	409,71

Fonte: Própria, 2022.

Já na Tabela 5 foi evidenciado os valores cobrados nos principais hidrômetros que podem ser classificados como produtos substitutos ao que será desenvolvido neste artigo.

É notório que o valor do protótipo é competitivo frente aos principais *players* localizados. Sobretudo que o projeto pode ter um caráter modular, ou seja, para alguns clientes, por exemplo, o sensor de temperatura pode ser retirado caso essa informação não tenha valor agregado ao cliente final.

Tabela 5 - Análise dos principais produtos do mercado - segundo semestre de 2022.

Equipamento	Valor (R$)
Hidrômetro Multijato Saneago 7m³/h - 1 polegada x260mm equipado para telemetria	494,99
Hidrômetro Multijato 10m³/h 1 polegada 260mm - Equipado para telemetria	502,99
Hidrômetro Multijato 20m³/h - Polegada 1 ½ - Equipado - Sob encomenda	732,99
Hidrômetro Multijato 3/4" Qmax 5m³/h com Saída Pulsada	338,00
Hidrômetro Multijato 7m³/h 1pol x260mm - Equipado para telemetria Sob Encomenda	489,99

Fonte: Própria, 2022.

6. CONCLUSÃO

Imaginando que um novo dimensionamento de insumos e que a produção em larga escala resulta, em algum grau, na otimização dos custos de fabricação. Comparando os principais *players* do mercado é possível afirmar que mesmo sem a otimização dos custos o módulo de Internet das Coisas proposto

neste artigo se mantém competitivo.

Com o uso da automatização da micromedição a mão de obra atual pode ser realocada para funções que demandam necessidade analítica do processo de faturamento da organização de *utilities*. Além disso, o cliente poderá ter acesso às informações que, atualmente, possuem uma periodicidade mais espaçada. Esta demora pode resultar em tomada de ação tardia na solução do vazamento e, com isso, em um desperdício de recurso natural essencial para a sobrevivência humana.

Contudo, existe a necessidade de ressaltarmos que é exigida uma análise prévia dos processos internos da empresa distribuidora de água e, treinamentos para capacitar a realocação da mão de obra precisam ser planejados. Além disso, ressalvas coligadas à possibilidade de furtos dos *hardwares* precisam ser apreciadas antes de sua implantação em larga escala.

Mesmo com as advertências, o grupo concluiu que o registrador eletrônico contém mais pontos positivos do que pontos negativos. Com isso, a conclusão é favorável à produção do módulo em maior escala.

6.1. Acertos e Desafios

Este projeto foi de suma importância para o desenvolvimento acadêmico do grupo, nele foi possibilitado o progresso na área de pesquisa e, com isso, algumas habilidades analíticas puderam ser exercitadas. Ademais, para que os objetivos propostos fossem solucionados, incontáveis matérias dadas, principalmente, no ciclo específico do Curso de Engenharia da Computação, foram utilizadas como sustentáculo do desdobramento do trabalho.

É possível afirmar que o presente artigo atendeu aos requisitos necessários. Foram feitas análises de problemáticas sociais, consultas em referenciais teóricos e uma análise dos resultados.

A principal dificuldade enfrentada pelo grupo foi a necessidade de mudança de escopo do projeto. A princípio, o grupo buscou realizar a leitura por imagem usando rede neural convolucional, mas as leituras ficavam imprecisas com pequenas variações de posição do equipamento, o que inviabilizou os testes. Diante disso, o grupo optou por usar o sensor de vazão que se baseia no princípio conhecido pelo nome de seu descobridor, o Efeito Hall.

6.2. Trabalhos Futuros e Melhorias

Para os pesquisadores que desejam incrementar este artigo, deixamos a sugestão de criação de um módulo de análise de possíveis irregularidades, onde próximo da relojoaria, um sensor que detecta campos magnéticos mensura se houve uma tentativa de fraude.

O desenho da placa de circuito impresso, junto da criação da carcaça de proteção é uma evolução válida, sobretudo para o bom funcionamento e proteção dos materiais utilizados. Na eventualidade de uma conversa com empresas que são fornecedoras industriais a atualização dos custos de produto, visando uma comparação mais fidedigna com os concorrentes, será de grande valia em um trabalho futuro.

O desenvolvimento de um módulo que elimina, ou reduz, a incidência das leituras de ar na tubulação, pode ser um diferencial mercadológico.

Por fim, a última melhoria proposta pelo grupo se interliga ao aspecto de conexão. O uso de uma tecnologia de rádio de longa distância auxiliará no processo de automatização de leitura. A tecnologia que o grupo propõe é a *LoRaWAN*.

7. Referências

[1] Seca fica mais branda nas regiões Sul e Centro-Oeste em agosto. Fenômeno fica mais severo no Sudeste, Nordeste e Tocantins. Agência Nacional de Águas e Saneamento Básico (ANA), 2022. Disponível em: <https://www.gov.br/ana/pt-br/assuntos/noticias-e-eventos/noticias/seca-fica-mais-branda-nas-regioes-sul-e-centro-oeste-em-agosto-fenomeno-fica-mais-severo-no-sudeste-nordeste-e-tocantins>. Acesso em: 25 Oct. 2022.

[2] DE SANEAMENTO BÁSICO, SABESP SA-Companhia. do Estado de São Paulo. Disponível em: <https://site.sabesp.com.br/uploads/file/clientes_servicos/deliberacao_arsesp106_13112009.pdf>. Acesso em: 25 Oct. 2022.

[3] PEREIRA DE MATTOS, D.; LYRA DA CONCEIÇÃO, N. Desenvolvimento do protótipo de um hidrômetro inteligente para consumo consciente do recurso águaX Simpósio de Engenharia de Produção - SIMEP 2022. Anais...Rio de Janeiro, Rio de Janeiro: Even3, 2022Disponível em: <http://dx.doi.org/10.29327/xsimep.472313>. Acesso em: 27 oct. 2022

[4] MAGRANI, Eduardo. A Internet das Coisas. [s.l.]: Editora FGV, 2018.

[5] CODROPS, for. Protocolo MQTT. Disponível em: <https://www.gta.ufrj.br/ensino/eel878/redes1-2018-1/trabalhos-vf/mqtt/#Prot>. Acesso em: 22 May 2023.

[6] ETC2. (s.d.). DHT22 Datasheet. Disponível em: https://pdf1.alldatasheet.com/datasheet-pdf/view/1132459/ETC2/DHT22.html. Acesso em: 22 Oct. 2023.

[7] BERTOLETI, P. Projetos com ESP32 e LoRa. [s.l.] Editora NCB, 2019.

[8] CASTRO, César Nunes de. Água, problemas complexos e o Plano Nacional de Segurança Hídrica. 2022.

[9] SISTEMA NACIONAL DE INFORMAÇÕES SOBRE SANEAMENTO - SNIS. Diagnóstico SNIS AE_2019 - REPUBLICAÇÃO _31-03-2021. 2019.

[10] Sabesp » Água » Controle de perdas. Disponível em: <https://site.sabesp.com.br/site/interna/Default.aspx?secaoId=37>. Acesso em: 25 Oct. 2022.

[11] Components101» Micro SD Card Adapter Module. Disponível em: <https://components101.com/modules/micro-sd-card-module-pinout-features-datasheet-alternatives>. Acesso em 25 Nov. 2023.

[12] Espressif Systems (Sem data). ESP32 Datasheet [online]. Disponível em: https://www.espressif.com/sites/default/files/documentation/esp32_datasheet_en.pdf . Acesso em: [18 Nov 2023].

[13] Mini solar Lipo Charger v1.0. Disponível em: <https://www.elecrow.com/wiki/index.php?title=Mini_solar_Lipo_Charger_v1.0>. Acesso em: 29 out. 2022.

[14] SISTEMAS E TECNOLOGIA APLICADA IND. COM. LTDA. Baterias e Lítio. 2020.

[15] SHENZHEN GLOBAL TECHNOLOGY CO. LTDA. Datasheet - F50-12V_SGT.xls. 2015.

[16] DFROBOT. Datasheet do DFR0379. 5 fev. 2017.

[17] IBM. MongoDB: Banco de dados NoSQL para aplicativos empresariais. Disponível em: https://www.ibm.com/br-pt/topics/mongodb. Acesso em: 20 Set 2023.

[18] AP, Haripriya et al. Secure-MQTT: an efficient fuzzy logic-based approach to detect DoS attack in MQTT protocol for internet of things. EURASIP Journal on Wireless Communications and Networking, v. 2019, n. 1, p. 1-15, 2019.

[19] Welcome. AI on the Edge Device. Disponível em: <https://jomjol.github.io/AI-on-the-edge-device-docs/#idea>. Acesso em: 22 May 2023.

[20] Eclipse Mosquitto. Eclipse Mosquitto. Disponível em: <https://mosquitto.org/>. Acesso em: 25 May 2023.

[21] **AllDatasheet. DALLAS DS3231 Datasheet [Online]. Disponível em:** https://pdf1.alldatasheet.com/datasheet-pdf/view/112132/DALLAS/DS3231.html**. Acesso em: 13 Nov 2023.**

[22] AllDatasheet. ENERGIZER CR2032 Datasheet [Online]. Disponível em: https://pdf1.alldatasheet.com/datasheet-pdf/view/160971/ENERGIZER/CR2032.html. Acesso em: 20 Set 2023.

[23] Vida de Silício. (s.d.). Sensor de Vazão de Fluxo de Água YF-S201. Disponível em: https://www.vidadesilicio.com.br/produto/sensor-vazao-fluxo-agua-yf-s201/.

[24] SABESP. Sabesp inicia projeto piloto de leitura remota do consumo de água. Disponível em: https://site.sabesp.com.br/site/interna/Default.aspx?secaoId=732. Acesso em: 17 dez. 2023.

[25] CHAVIER, Luís Fernando. Programação para Arduino: Primeiros Passos. [S.l.]: [s.n.], 2017. Disponível em: https://professor.luzerna.ifc.edu.br/marcelo-cendron/wp-content/uploads/sites/40/2017/03/Programa%C3%A7%C3%A3o-para-Arduino-Primeiros-Passos-Conceitos-iniciais-de-programa%C3%A7%C3%A3o-para-Arduino-Projeto-de-eletr%C3%B4nica-modular-com-Arduino-Circuitar.pdf. Acesso em: 18 dez. 2023.

[26] AMAZON WEB SERVICES. O que é Python?. [s.d.]. Disponível em: https://aws.amazon.com/pt/what-is/python/#:~:text=O%20Python%20%C3%A9%20uma%20linguagem,executada%20em%20muitas%20plataformas%20diferentes.. Acesso em: 16 dez. 2023.

8. ANEXOS

https://github.com/joaovfg96/sensor-de-vazao-tcc