UNIVERSIDADE FEDERAL DE PERNAMBUCO
CENTRO DE TECNOLOGIA E GEOCIÊNCIAS
DEPARTAMENTO DE ENGENHARIA ELÉTRICA
CURSO DE GRADUAÇÃO EM ENGENHARIA ELÉTRICA

PEDRO IGOR DA SILVA AGUIAR

SISTEMA DE MONITORAMENTO INTELIGENTE DE ENERGIA APLICADO A MICRORREDES: PROJETO DE MEDIDOR BIDIRECIONAL E APLICAÇÃO WEB.

Recife
2024

PEDRO IGOR DA SILVA AGUIAR

SISTEMA DE MONITORAMENTO INTELIGENTE DE ENERGIA APLICADO A MICRORREDES: PROJETO DE MEDIDOR BIDIRECIONAL E APLICAÇÃO WEB.

Trabalho de Conclusão de Curso apresentado ao Departamento de Engenharia Elétrica da Universidade Federal de Pernambuco, como requisito parcial para obtenção do grau de Engenheiro Eletricista.

Orientador(a): Prof. Dr. José Filho da Costa Castro

Coorientador: Prof. Dr. Davidson da Costa Marques

Recife
2024

Ficha de identificação da obra elaborada pelo autor,
através do programa de geração automática do SIB/UFPE

Aguiar, Pedro Igor da Silva.
Sistema de monitoramento inteligente de energia aplicado a microrredes: projeto de medidor bidirecional e aplicação web. / Pedro Igor da Silva Aguiar. - Recife, 2024.
95 p. : il., tab.

Orientador(a): José Filho da Costa Castro
Cooorientador(a): Davidson da Costa Marques
(Graduação) - Universidade Federal de Pernambuco, Centro de Tecnologia e Geociências, , 2024.
Inclui referências, apêndices.

1. Microrredes. 2. Medidor Inteligente. 3. Aplicação Web. 4. Monitoramento Remoto. 5. Gestão da Energia. I. Castro, José Filho da Costa. (Orientação). II. Marques, Davidson da Costa. (Coorientação). IV. Título.

620 CDD (22.ed.)

PEDRO IGOR AGUIAR DA SILVA

SISTEMA DE MONITORAMENTO INTELIGENTE DE ENERGIA APLICADO A MICRORREDES: PROJETO DE MEDIDOR BIDIRECIONAL E APLICAÇÃO WEB.

Trabalho de Conclusão de Curso apresentado ao Departamento de Engenharia Elétrica da Universidade Federal de Pernambuco, como requisito parcial para obtenção do grau de Engenheiro Eletricista.

Aprovado em: 21/03/2024.

BANCA EXAMINADORA

_____
Prof. Dr. José Filho da Costa Castro (Orientador)
Universidade Federal de Pernambuco

_____
Prof. Dr. Douglas Contente Pimentel Barbosa (Examinador Interno)
Universidade Federal de Pernambuco

_____
M.Sc Néstor Iván Medina Galdino (Examinador Interno)
Universidade Federal de Pernambuco

AGRADECIMENTOS

Agradeço a minha família, em especial, aos meus pais, Maria da Guia e Irenildo, a minha irmã, Gabriela, e a minha tia, Maria José, por serem a base e fornecerem o suporte necessário para o meu desenvolvimento profissional, investindo na minha educação, absorvendo os pesos da jornada e acreditando que sou capaz de conquistar todos os meus objetivos.

À equipe Mandacaru Aerodesign, por ser a minha segunda casa, por me fazer resiliente, por me apresentar amigos que levarei para o resto da vida e por todos os momentos de felicidade, decepção, triste e redenção vividos durante o curso. Enfim, por me dar a honra de ver meu próprio avião voar.

Aos amigos da faculdade, que ajudaram a deixar toda a graduação mais leve. Em particular, ao grupo Insolide, serei eternamente grato pelas experiências, que me mantiveram firme e me permitiram pular muros na etapa final.

A Salvus, por fornecer um ambiente incrível para o meu desenvolvimento profissional, e aos amigos que fazem ou fizeram parte dela. Obrigado pela parceria, pelos momentos e pelos ensinamentos que sempre me deixarão saudades.

Aos professores do Departamento de Engenharia Elétrica da UFPE, em especial, ao professor Dr. Davidson Marques e ao professor Dr. José Castro pelas orientações, disponibilidade de tirar dúvidas e me auxiliar em todo trabalho. Obrigado por acreditarem na minha capacidade.

Por fim, a todos que, direta ou indiretamente, fizeram parte dessa importante trajetória. Muito obrigado.

RESUMO

Uma rede elétrica inteligente é um novo tipo de rede em desenvolvimento, que permite fluxo de energia não convencional e fluxo de informação bidirecional. Ou seja, ela é fundamental para a implementação de uma rede de entrega de energia automática e distribuída. Uma microrrede é um sistema controlável composto por um agrupamento de cargas e recursos energéticos distribuídos, os quais incluem tecnologias de geração e armazenamento. Esse sistema opera fornecendo energia de maneira integrada dentro dos limites da área de uma concessionária de distribuição. Nesse contexto, faz-se necessário o monitoramento e gerenciamento de forma inteligente desses recursos a fim de promover eficiência e qualidade no fornecimento da energia. Este trabalho consiste no projeto de um medidor inteligente bidirecional de energia e uma plataforma web de monitoramento e gerenciamento aplicados a microrredes. Assim, foi produzido um dispositivo que realiza medições dos parâmetros elétricos, tais como tensão, corrente, consumo, geração, fator de potência, entre outros, e que envia os dados em tempo real via internet para armazenamento na nuvem. Além disso, desenvolveu-se uma aplicação web para exibir as informações coletadas e, assim, permitir o gerenciamento da microrrede. Dessa forma, é possível monitorar o consumo e a geração, bem como, obter o histórico dos parâmetros do sistema para análises mais complexas.

Palavras-chaves: Microrredes. Medidor Inteligente. Aplicação Web. Monitoramento Remoto. Gestão da Energia.

ABSTRACT

A smart grid is a new type of network under development, which allows unconventional power flow and bidirectional information flow. In other words, it is essential for implementing an automatic and distributed energy delivery network. A microgrid is a controllable system composed of a grouping of loads and distributed energy resources, including generation and storage technologies. This system operates by providing energy in an integrated manner within the limits of a distribution utility area. In this context, it is necessary to intelligently monitor and manage these resources to promote efficiency and quality in energy supply. This work consists of the design of a bidirectional smart energy meter and a web monitoring and management platform applied to microgrids. Therefore, a device was produced that measures electrical parameters such as voltage, current, consumption, production, power factor, among others, and sends real-time data via the internet for cloud storage. Additionally, a web application was developed to display the collected information and thus allow microgrid management. In this way, it is possible to monitor consumption and generation, as well as obtain the system parameter history for more complex analyses.

Keywords: Microgrids. Smartmeter. Web Application. Remote Monitoring. Energy Management.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

API	*Application Programming Interface*
AWS	*Amazon Web Services*
CPU	*Central Processing Unit*
DER	Diagrama Entidade-Relacionamento
DNS	*Domain Name System*
EC2	*Elastic Compute Cloud*
ER	Entidade-Relacionamento
EV	*Electric Vehicle*
GD	Geração Distribuída
GPIO	*General Purpose Input/Output*
HEV	*Hybrid Electric Vehicle*
HTTP	*HyperText Transfer Protocol*
HTTPS	*Hyper Text Transfer Protocol Secure*
I2C	*Inter-Integrated Circuit*
I2S	*Inter-IC Sound Bus*
IoT	*Internet of Things*
IP	*Internet Protocol*
JSON	*JavaScript Object Notation*
LAM	Laboratório de Mobilidade
LED	*Light-Emiting Diode*
LoRaWAN	*Long Range Wide Area Network*
M2M	*Machine to Machine*
MQTT	*Message Queuing Telemetry Transport*
MMGD	Micro e Minigeração Distribuída

PCB	*Printed Circuit Board*
RDB	*Relational Database*
REDs	Recursos Energéticos Distribuídos
REI	Rede Elétrica Inteligente
REST	*Representational State Transfer*
RF	Radiofrequência
RDS	*Relational Database Service*
RTC	*Real Time Clock*
SD	*Standart Capacity*
SG	*Smart Grid*
SGBD	Sistema Gerenciador de Banco de Dados
SMES	*Superconducting Magnetic Energy Storage*
SPI	*Serial Peripheral Interface*
TCP	*Transmission Control Protocol*
TC	Transformador de Corrente
TI	Tecnologia da Informação
TP	Transformador de Potencial
TVS	*Transient Voltage Suppression*
UART	*Universal Asynchronous Receiver-Transmitter*
URL	*Uniform Resource Locator*
USB	*Porta Serial Universal*
WWW	*World Wide Web*

LISTA DE SÍMBOLOS

V	Tensão elétrica
V_1	Tensão no primário do transformador
V_2	Tensão no secundário do transformador
V_{rms}	Tensão Eficaz
V_{out}	Tensão de saída
V_{adc}	Tensão de referência do ADC
V_{offset}	Tensão de *offset*
I	Corrente elétrica
I_{rms}	Corrente Eficaz
I_1	Corrente no primário do transformador
I_2	Corrente no secundário do transformador
$I_{1,pico}$	Corrente de pico no primário do transformador
$I_{2,pico}$	Corrente de pico no secundário do transformador
N	Número de amostras
N_1	Número de espiras no primário do transformador
N_2	Número de espiras no secundário do transformador
R	Resistência
C	Capacitância
P	Potência Ativa
Q	Potência Reativa
S	Potência Aparente
E	Energia
FP	Fator de potência
f_c	Frequência de corte

SUMÁRIO

1 INTRODUÇÃO

No Brasil, a implementação das redes elétricas inteligentes apresenta particularidades que a diferenciam dos países desenvolvidos, conforme discutido em (RIVERA; ESPOSITO; TEIXEIRA, 2013). Nesse contexto, é importante destacar que a matriz elétrica, que possui dimensão continental, é predominantemente renovável e está interligada por um sistema integrado de geração e transmissão. Além disso, o país possui um alto potencial de recursos renováveis e não renováveis ainda não explorados, enquanto o de energia per capita é consideravelmente inferior em comparação com outras nações. No entanto, as tarifas de energia no Brasil estão entre as mais elevadas do mundo.

Nessa perspectiva, devido à procura dos consumidores por alternativas para reduzir a conta de energia elétrica, a modalidade de Micro e Minigeração Distribuída (MMGD) cresce cada vez mais a cada ano. Conforme descrito em (EPE, 2021), no ano de 2019, foram adicionados 1,5 GW de capacidade em sistemas de MMGD no sistema elétrico brasileiro. No ano subsequente, a tecnologia fotovoltaica distribuída se destacou como a segunda fonte com maior incremento de capacidade instalada na matriz elétrica do Brasil, ultrapassando as fontes eólica e fotovoltaica centralizada, ficando somente atrás da hidrelétrica.

Dessa forma, devido à qualidade dos recursos energéticos nacionais, às tarifas finais de eletricidade elevadas e a um modelo de compensação de créditos vantajoso, o investimento em geração própria tornou-se rentável no Brasil. Esse cenário atraiu não apenas consumidores residenciais, mas também grandes redes varejistas, instituições financeiras e indústrias a investirem em sistemas de MMGD (EPE, 2021).

Outrossim, o armazenamento de eletricidade pode ser empregado em diversos pontos do setor elétrico. No Brasil, há perspectivas de uso de baterias em unidades consumidoras para o aumento do autoconsumo da microgeração distribuída (EPE, 2021). Isso porque os sistemas de armazenamento são uma alternativa para complementar a geração distribuída intermitente, como sistemas fotovoltaicos, atuando como um reserva de energia.

Nesse cenário, a associação de cargas e recursos energéticos formam as microrredes. Assim, este trabalho propõe um sistema de monitoramento e gestão eficiente das microrredes para potencializar a utilização dos seus recursos e sua integração com a rede elétrica a fim de proporcionar um fornecimento de energia de qualidade e com menores tarifas.

1.1 OBJETIVOS

1.1.1 Objetivo Geral

Promover a gestão energética de microrredes e a integração dos recursos energéticos distribuídos através de um sistema de monitoramento inteligente a fim de garantir maior confiabilidade, eficiência e qualidade da energia elétrica.

1.1.2 Objetivos Específicos

1. Projetar e implementar um medidor bidirecional monofásico não invasivo de baixa tensão:
    - Identificar e especificar as grandezas essenciais para medição de consumo e geração de energia;
    - Desenvolver, prototipar e validar o circuito eletrônico de medição;
    - Projetar e fabricar a PCB do circuito desenvolvido;
2. Criar um sistema Web para armazenamento, análise e exibição dos dados:
    - Implementar a interface de comunicação para integração os dispositivos de medição e o sistema Web;
    - Modelar um banco de dados para armazenamento das informações na nuvem;
    - Projetar uma API para consulta dos dados armazenados;
    - Desenvolver a interface de usuário para a aplicação Web;

1.2 JUSTIFICATIVA

Com a constante evolução tecnológica e redução dos custos de investimento, há um aumento da presença dos recursos energéticos distribuídos no sistema elétrico. Estima-se que os recursos energéticos distribuídos correspondam a 19% do consumo de energia elétrica até 2030, segundo (EPE, 2021).

Nessa perspectiva, o monitoramento inteligente desses recursos é de extrema relevância para um gerenciamento do fluxo de energia, consumo, detecção de falhas e etc. Dessa forma, o medidor inteligente é uma peça fundamental para viabilizar a implementação de microrredes, além disso, a plataforma Web permite a integração dos dados de diversos subsistemas como sistemas fotovoltaicos, sistemas eólicos, sistemas de armazenamento e estações de carregamento de veículos elétricos em um único lugar.

1.3 ESTRUTURA DO TRABALHO

Este trabalho de conclusão de curso está organizado em 6 capítulos. No Capítulo 1, foram apresentados os objetivos e a motivação no contexto da crescente disseminação da geração distribuída no cenário nacional.

No Capítulo 2, são introduzidos os conceitos para entender o escopo de aplicação do medidor, tais como redes elétricas inteligentes, recursos energéticos distribuídos e microrredes. Além disso, abordam-se as tecnologias de *hardware* e *software* utilizadas para desenvolver o sistema de monitoramento.

O Capítulo 3 apresenta a arquitetura, os requisitos, os componentes e as tecnologias utilizadas, bem como as etapas de projeto do sistema desenvolvido. Enquanto no Capítulo 4, são evidenciados os resultados de cada fase do desenvolvimento.

Por fim, no Capítulo 5, observando os objetivos propostos na Seção 1.1, é realizada uma conclusão geral do que foi desenvolvido e discutida sugestões de trabalhos futuros.

2 FUNDAMENTAÇÃO TEÓRICA

2.1 REDES ELÉTRICAS INTELIGENTES

Uma Rede Elétrica Inteligente (REI) ou *Smart Grid* (SG) é capaz de integrar de maneira inteligente os componentes conectados a ela para otimizar a produção, distribuição e consumo de energia. Dessa forma, ela viabiliza a entrada na rede de novos fornecedores – como fontes geradoras descentralizadas, renováveis e intermitentes – e consumidores – como veículos elétricos –, permitindo melhorias em monitoramento, gestão, eficiência e qualidade da energia (VIJAYAPRIYA; KOTHARI, 2011).

Assim, as SGs são caracterizadas pelo uso de tecnologias de informação e comunicação por meio de dispositivos de medição inteligentes, automação e sistemas de gerenciamento. Segundo (RIVERA; ESPOSITO; TEIXEIRA, 2013), a implantação das REIs pode ser estruturada em três perspectivas:

1. Intervenções para incorporar inteligência ao sistema de fornecimento de energia elétrica, promovendo robustez, segurança e agilidade na rede;
2. Substituição dos medidores eletromecânicos por eletrônicos inteligentes, que passam a oferecer funcionalidades como o corte e religamento remoto, indicativos de qualidade de energia, entre outros;
3. Uso de sistemas inteligentes nos centros consumidores, permitindo uma melhor gestão do consumo energético, da geração distribuída e armazenamento de energia.

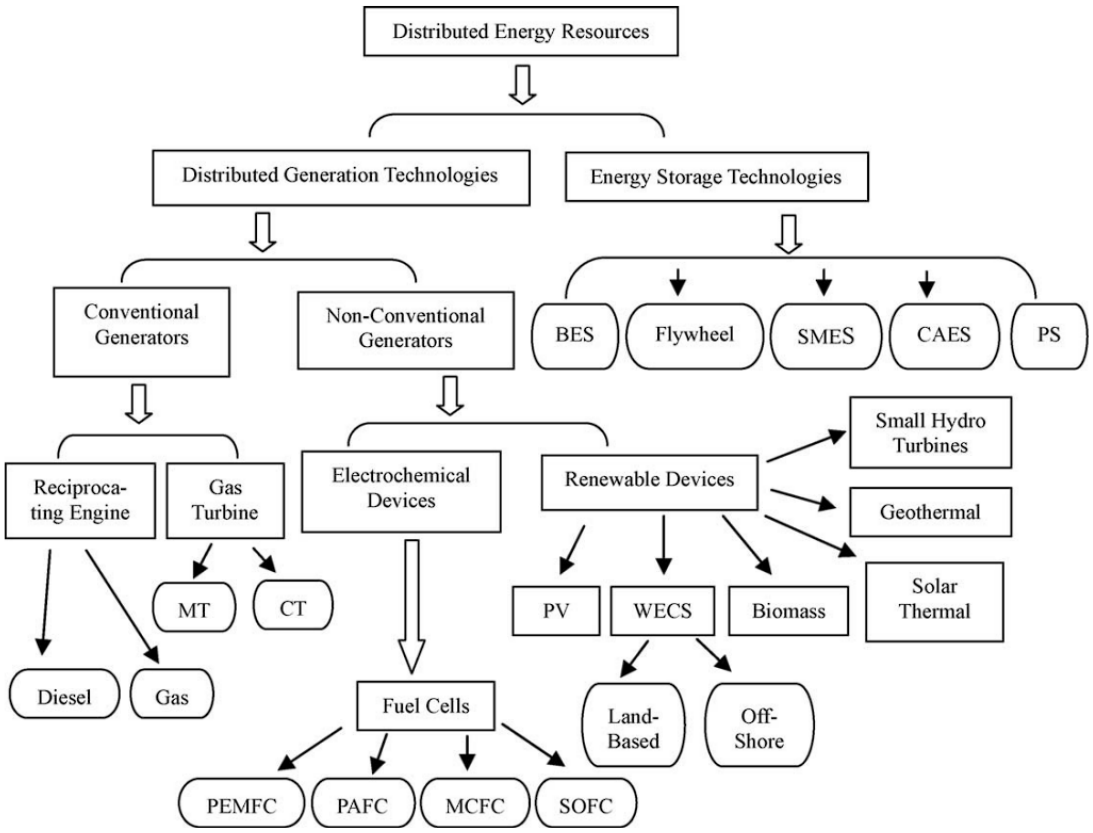
O objetivo deste trabalho é desenvolver um sistema de monitoramento inteligente focado na terceira perspectiva, a fim de proporcionar um melhor gerenciamento e integração entre microrredes (Seção 2.3) e seus recursos energéticos distribuídos (Seção 2.2).

2.2 RECURSO ENERGÉTICOS DISTRIBUÍDOS

Como apresentado em (AKOREDE; HIZAM; POURESMAEIL, 2010), os Recursos Energéticos Distribuídos (REDs) se referem aos meios de geração de energia elétrica que estão diretamente conectados aos sistemas de distribuição de média e baixa tensão em vez de sistemas de transmissão de energia em grande escala, isto é, então ligados próximos a unidades consumidoras, atrás do medidor (*behind-the-meter*).

Os REDs podem ser divididos em duas categorias: tecnologias de geração, tais como células de combustível, microturbinas e sistemas fotovoltaicos, e tecnologias de armazenamento de energia, que incluem baterias, volantes de inércia, veículos elétricos, entre outros (AKOREDE; HIZAM; POURESMAEIL, 2010). A Figura 1 ilustra as subdivisões dessa classificação e as principais tecnologias associadas a elas.

Figura 1 – Tecnologias dos REDs.



Fonte: (AKOREDE; HIZAM; POURESMAEIL, 2010).

2.2.1 Geração Distribuída

A Geração Distribuída (GD) é definida como qualquer fonte de energia elétrica de capacidade limitada, diretamente conectada à rede de distribuição de energia elétrica, que é consumida pelos usuários finais (AKOREDE; HIZAM; POURESMAEIL, 2010). Desse modo, a GD pode incluir tecnologias como sistemas fotovoltaicos, turbinas eólicas, usinas hidrotermais, motores a combustão, entre outros. Este trabalho utilizou um sistema fotovoltaico como ambiente de testes e validações do projeto desenvolvido.

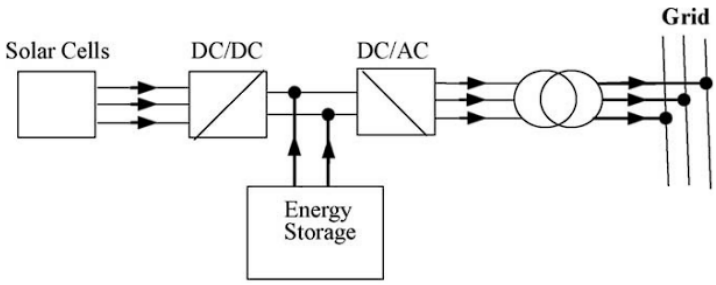
2.2.1.1 Sistemas Fotovoltaicos

A conversão da energia térmica solar em energia elética é feita utilizando células fotovoltaicas, através do efeito fotovoltaico. Nesse contexto, define-se como módulo fotovoltaico o conjunto de células conectadas em arranjos, série ou paralelo, a fim de produzir tensão e corrente suficientes para a utilização prática da energia (PINHO, 2014).

Como detalhado em (PINHO, 2014), um sistema fotovoltaico é composto por um bloco gerador, um bloco de condicionamento de potência e, opcionalmente, um bloco de armazenamento. O bloco gerador contém os módulos fotovoltaicos em diferentes associações, o cabeamento que os interliga e a estrutura de suporte. O bloco de condicionamento de potência

é constituído de conversores CC-CC, inversores, controladores de carga e outros dispositivos de proteção e controle. Por fim, o bloco de armazenamento é normalmente constituído de baterias. A Figura 2 apresenta um exemplo de topologia de um sistema fotovoltaico.

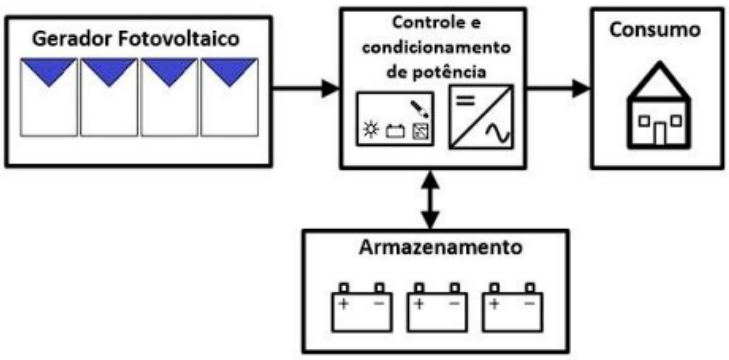
Figura 2 – Esquemático de um sistema fotovoltaico.



Fonte: (AKOREDE; HIZAM; POURESMAEIL, 2010).

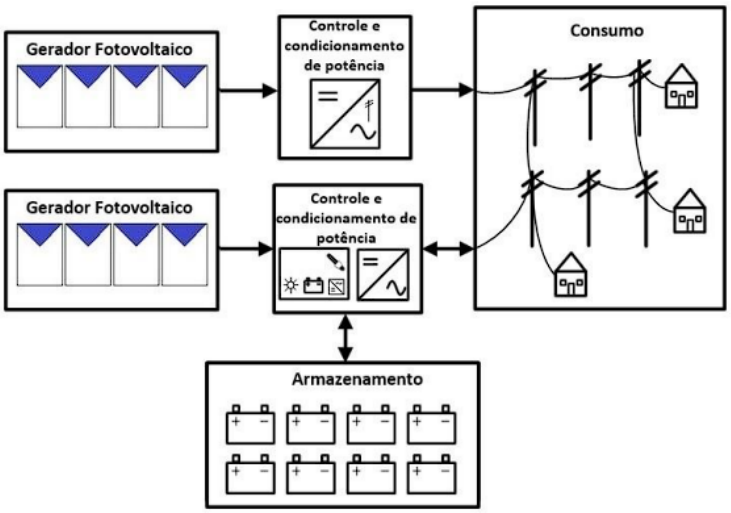
Além disso, esses sistemas são caracterizados por estarem ou não interligados ao sistema público de fornecimento de energia elétrica (AKOREDE; HIZAM; POURESMAEIL, 2010). A principal diferença entre essas configurações é o destino da energia excedente. No caso de o sistema não estar conectado à rede elétrica (*off-grid*), essa energia normalmente é direcionada aos sistemas de armazenamento para uso posterior (Figura 3). Entretanto, quando conectado (*on-grid*), a energia excedente pode ser enviada para a rede. Não obstante, um sistema *on-grid* também pode incorporar tecnologias de armazenamento de energia, como mostrado na Figura 4.

Figura 3 – Diagrama de blocos um sistema *off-grid* com sistema de armazenamento.



Fonte: (PINHO, 2014).

Figura 4 – Diagrama de blocos um sistema *on-grid* com e sem sistema de armazenamento.



Fonte: (PINHO, 2014).

2.2.2 Sistemas de Armazenamento

O sistema elétrico deve ter alguma capacidade de armazenamento para superar as flutuações no fornecimento de energia. Dessa forma, os sistemas de armazenamento de energia fornecem não só uma alternativa para reduzir o desperdício quando a geração é insuficiente, mas também um meio de aproveitar o excesso de produção (AKOREDE; HIZAM; POURESMAEIL, 2010). Eles podem incluir: baterias, volantes de inércia, supercondutores magnéticos (SMES), veículos elétricos, etc.

De acordo com (LASSETER, 2002), os sistemas de armazenamento, além de mitigar a intermitência de fontes renováveis, proporcionam estabilidade, qualidade e confiabilidade ao fornecimento de energia, melhorando o desempenho total da rede. Eles podem atuar como uma reserva de energia que equilibra a demanda do consumidor nos períodos nos quais a geração na GD é nula ou deficiente. Portanto, esses sistemas podem compensar as quedas da tensão local, servir como reserva durante falhas no fornecimento, apoiar a rede em situações de sobrecarga, entre outros.

2.2.2.1 Baterias

As baterias recarregáveis são utilizadas para armazenar eletricidade na forma de energia química. Para atender aos requisitos de armazenamento de energia, a bateria deve ter alta densidade de energia, alta potência, alta eficiência de carga, boa capacidade de ciclagem, longa vida útil e baixo custo inicial (AKOREDE; HIZAM; POURESMAEIL, 2010). A maioria dos sistemas de armazenamento de bateria é composto por grandes números de células de chumbo-ácido, utilizando tecnologia semelhante à encontrada em baterias de automóveis.

Ademais, conforme explicado em (AKOREDE; HIZAM; POURESMAEIL, 2010), outras aplicações em que as baterias são consideradas para sistemas de energia elétrica incluem nivelamento

de carga e controle de tensão, reativos e frequência. Adicionalmente, elas proporcionam um tempo de resposta rápido, são silenciosas e não poluentes, sendo ideais para instalação em áreas próximas aos centros de carga.

2.2.2.2 Veículos Elétricos

Atualmente, existem três tipos relevantes de veículos elétricos: os totalmente elétricos (EVs), os elétricos com célula de combustível e os híbridos elétricos (HEVs). Os EVs a bateria e com célula de combustível são impulsionados apenas por energia elétrica, enquanto os HEVs disponíveis atualmente também possuem um motor de combustão interna (LOPES FILIPE JOEL SOARES, 2010).

Como esses veículos exigem o uso de baterias com alta capacidade de armazenamento de energia e grandes requisitos de carga elétrica, uma ampla implantação desse conceito pode impactar na operação do sistema elétrico, assim como possibilitar e beneficiar o uso de recursos energéticos não poluentes. Isso porque os EVs, quando estacionados e conectados à rede elétrica, não apenas absorvem e armazenam energia, mas também têm a capacidade de fornecer eletricidade de volta à rede. Essa capacidade permite a prestação de serviços auxiliares, ou seja, os EVs podem ser capazes de fornecer energia extra para picos de demanda, realizar deslocamento de carga no nível de distribuição, fornecer reservas girantes ou regulação para o sistema (LOPES FILIPE JOEL SOARES, 2010).

Em síntese, com a falta de controle das fontes de energia renováveis intermitentes, a capacidade dos EVs de armazenar e injetar energia posteriormente no sistema evitará o desperdício de energia limpa, resultando na redução do uso de unidades convencionais de combustíveis fósseis e geradores caros durante as horas de pico. Isso permitirá a redução das emissões de poluentes e dos custos de geração de energia, provocando também impactos significativos no comportamento dos mercados de eletricidade (AKOREDE; HIZAM; POURESMAEIL, 2010).

2.3 MICRORREDES

Em (LASSETER, 2002), a microrrede é conceituada como um agrupamento de cargas e REDs operando como um único sistema controlável que fornece energia. Além disso, elas têm capacidade de operar de forma autônoma ou conectada à rede (BARNES et al., 2007).

Nesse contexto, como detalhado em (BELLIDO, 2018), as microrredes apresentam diversas condições fundamentais para sua funcionalidade e operacionalidade. Isso inclui a capacidade de gerar energia para atender à demanda do consumidor; o gerenciamento eficaz para manter os requisitos mínimos de operação; a facilidade na integração de novos sistemas, na conexão e no isolamento da rede; e a continuidade operacional mesmo em caso de perda de qualquer equipamento.

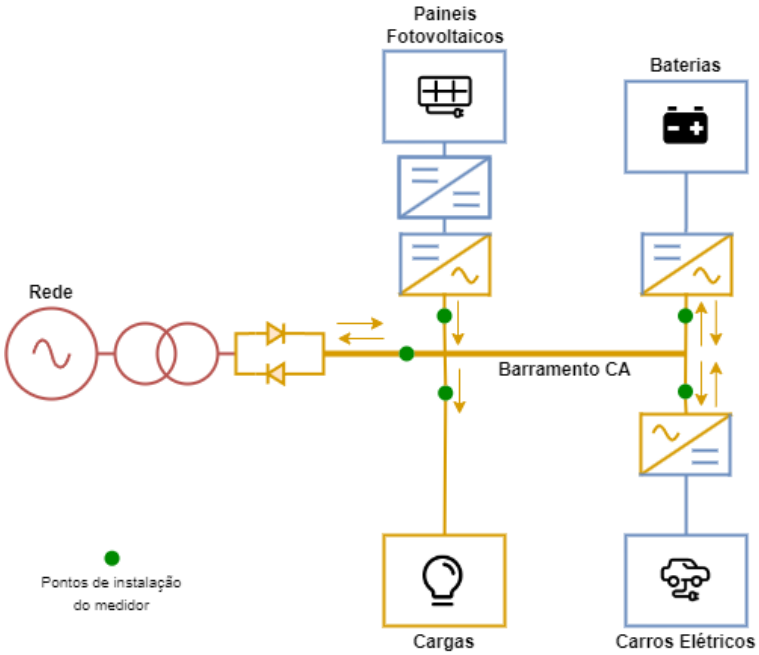
Com objetivo de viabilizar essas condições, microrredes possuem componentes que garantem seu funcionamento, integração e proteção, tais como interfaces de conexão, equipamentos,

controles e dispositivos de proteção. Segundo (BELLIDO, 2018), esses componentes são agrupados em três grupos:

- **Geração, armazenamento e cargas:** os REDs (Seção 2.2) são as tecnologias geralmente utilizadas para geração e armazenamento nas microrredes. As cargas são os componentes que consumem energia;
- **Rede física para distribuição:** são as responsáveis por conectar e distribuir a energia entre os diversos componentes da microrrede. Elas podem ser aéreas ou subterrâneas, e operam de maneira similar às redes de distribuição, porém, em menor escala, devendo também possuir dispositivos de proteção e segurança;
- **Controles avançados:** são utilizados para proteger, operar e controlar a microrrede para atender as demandas dos consumidores de forma segura e confiável.

A Figura 5 exemplifica o diagrama de uma microrrede e seus componentes. Neste trabalho, projetou-se um medidor monofásico inteligente bidirecional que pode ser instalado em diferentes pontos do barramento de corrente alternada a fim de medir os parâmetros de consumo e geração do recurso que está sendo monitorado.

Figura 5 – Diagrama de uma microrrede.



Fonte: O Autor (2024).

2.4 MEDIDORES INTELIGENTES DE ENERGIA

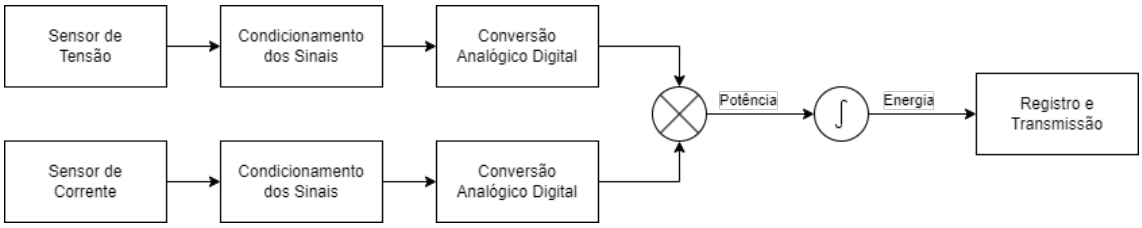
Conforme definido em (ZHENG; GAO; LIN, 2013), o medidor inteligente ou *smartmeter* é um dispositivo de medição de energia que adquire informações de carga dos usuários, mede o consumo de energia dos consumidores e fornece informações adicionais a fim de melhor monitoramento e a gestão da energia.

Com o *smartmeter*, a partir da medição das variáveis elétricas, como tensão e corrente são calculadas as informações de consumo, geração, potência, fator de potência, frequência de energia, entre outros, e registradas em tempo real. Além disso, ele pode ser capaz de desconectar e reconectar determinadas cargas remotamente, ou seja, ele suporta comunicações bidirecionais entre o dispositivo e o sistema (ZHENG; GAO; LIN, 2013).

Para realizar essas funcionalidades, são utilizados vários sensores aliados a uma infraestrutura de comunicação dedicada. Neste trabalho, projetou-se um medidor inteligente que utiliza a comunicação via radiofrequência (RF) para trocas bidirecionais de informações e adquire os dados por meio de sensores de tensão e corrente.

A Figura 6 apresenta o diagrama de blocos do funcionamento de um medidor. A medição é feita a partir de sensores de tensão e corrente, que recebem os parâmetros de entrada. Os sinais dos sensores são condicionados para serem digitalizados pelo ADC e processados no microcontrolador. Em seguida, os valores obtidos são multiplicados, obtendo-se a potência instantânea e, posteriormente, integrados para calcular a energia. Por fim, as informações são registradas e transmitidas (NICOLAU; SOUZA; FROTA, 2013).

Figura 6 – Diagrama de funcionamento do medidor inteligente.



Fonte: O Autor (2024).

2.5 MEDIÇÃO DE CORRENTE

Os sensores de corrente podem ser classificados com base nos conceitos físicos que regem seu funcionamento (ZIEGLER et al., 2009). A Tabela 1 apresenta exemplos de formas de sensoriamento e seus respectivos princípios físicos.

Os sensores de corrente baseados na lei da indução de Faraday proporcionam o isolamento elétrico inerente entre a corrente medida e o sinal de saída. O isolamento elétrico permite a medição com maior segurança de correntes em um potencial de tensão mais elevado (ZIEGLER et al., 2009). Por esse motivo, foram utilizados transformadores de corrente para realizar o sensoriamento neste projeto.

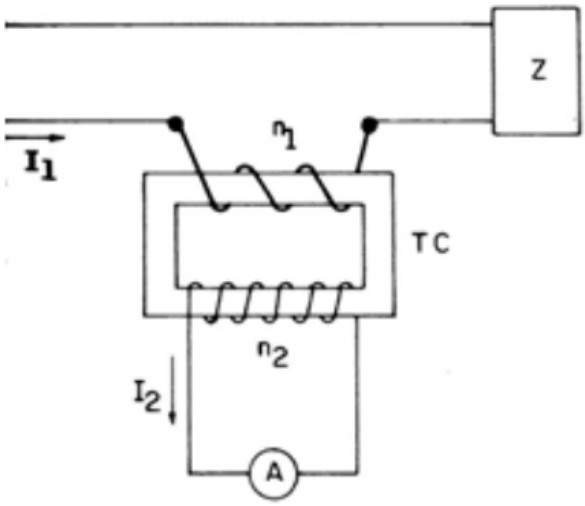
Tabela 1 – Classificação dos sensores de corrente.

Princípio de Físico	**Sensores**
Lei de Ohm	Resistor Shunt
Lei de Faraday	Bobina de Rogowski e Transformador de Corrente
Campo Magnético	Sensor de Efeito Hall
Efeito Faraday	Método de Detecção com Polarímetro e Método de Detecção com Interferômetro

2.5.1 Transformador de Corrente

O transformador de corrente (TC) é um dispositivo de instrumentação não invasivo que adquire uma amostra da corrente que flui em uma linha e a reduz a um nível seguro e mensurável (CHAPMAN, 2013). Sua construção é composta por um núcleo ferromagnético envolvido por espiras feitas de condutor de cobre com grande seção. O enrolamento primário é constituído de poucas espiras, enquanto o secundário possui várias espiras (Figura 7). Além disso, há TCs em que o próprio condutor do circuito principal serve como primário, neste caso, é considerado que o enrolamento primário contém apenas uma espira (FILHO, 1980).

Figura 7 – Esquemático de um transformador de corrente.



Fonte: Adaptado de (FILHO, 1980).

O TC possui os enrolamentos fracamente acoplados, ou seja, o fluxo mútuo é menor que o fluxo de dispersão. Dessa forma, o núcleo retém e concentra uma pequena parte do fluxo oriundo da linha primária e esse fluxo induz uma tensão e uma corrente no enrolamento secundário. Assim, o sinal de saída é proporcional à corrente no primário (CHAPMAN, 2013). Em um transformador ideal de corrente construído com uma relação de espiras N_1:N_2, a relação de transformação entre o módulo da corrente no secundário e o módulo da corrente de entrada é dado pela Equação 2.1, e a defasagem é nula.

$$\frac{I_2}{I_1} = \frac{N_1}{N_2} \tag{2.1}$$

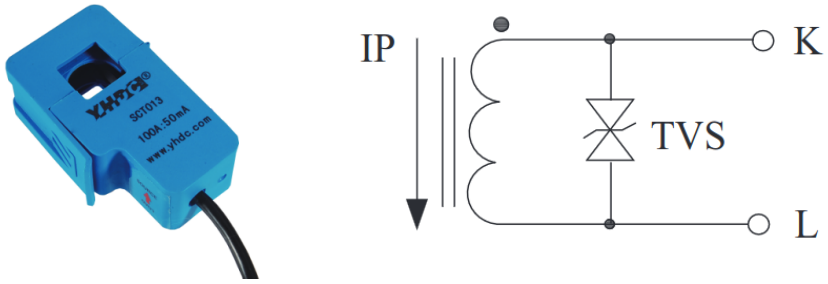
Ademais, tensões elevadas podem surgir se os terminais do enrolamento secundário estiverem abertos. Portanto, é importante que o secundário do TC esteja em curto-circuito ou conectado através de um resistor de carga, ou de sensibilidade (ZIEGLER et al., 2009).

2.5.1.1 SCT-013

O TC utilizado no projeto foi o SCT-013-000 (Figura 8), que possui uma relação de espiras de 1:2000, um diodo TVS[1] para proteção e uma capacidade de medição corrente eficaz de até 100 A (ELECROW, 2011). Ou seja, pela Equação 2.1, o sinal de saída é uma corrente senoidal de aproximadamente 71 mA de amplitude. Dessa forma, é necessário um circuito condicionador de sinais para adequar esse sinal ao dispositivo de medição.

Figura 8 – Transformador de corrente SCT-013.

(a) Representação real. (b) Esquema elétrico do SCT-013.

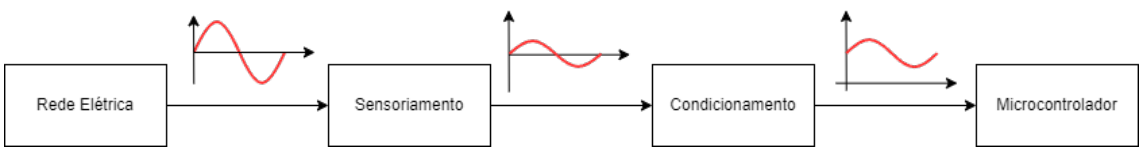


Fonte: (ELECROW, 2011).

2.5.2 Circuito Condicionador de Sinais

Normalmente, os conversores analógico-digitais dos microcontroladores são capazes de converter apenas sinais de tensão positivos. Como a saída do TC é uma corrente alternada proporcional a de entrada, faz-se necessário um circuito que transforme o sinal de corrente alternada em um sinal de tensão positiva para uma correta digitalização (Figura 9).

Figura 9 – Diagrama de condicionamento do sinal.



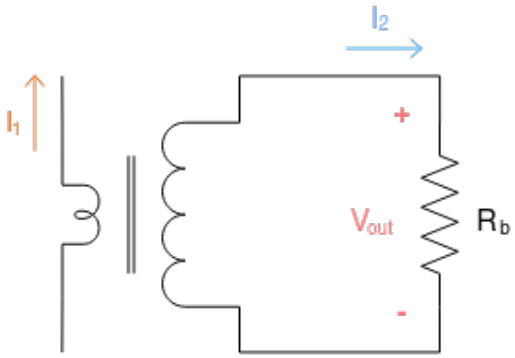
Fonte: O Autor.

Inicialmente, é preciso converter um sinal de corrente em um sinal de tensão. Para isso, utiliza-se um resistor de carga (*burden resistor*) conectado nos terminais do secundário do TC.

[1] Um diodo de supressão de tensão transitória (TVS) é um componente eletrônico usado para proteger circuitos eletrônicos de picos de tensão induzidos (SEMICONDUCTOR, 2005).

Assim, devido à lei de Ohm, a corrente alternada da saída do TC (I_2) irá gerar uma queda de tensão alternada (V_{out}) (Figura 10). O resistor R_b deve ser escolhido de forma que a tensão de pico a pico gerada seja menor ou igual a tensão de referência do ADC (V_{adc}), ou seja, a tensão de pico deve ser metade de V_{adc} (Figura 11).

Figura 10 – Resistor de carga do TC.

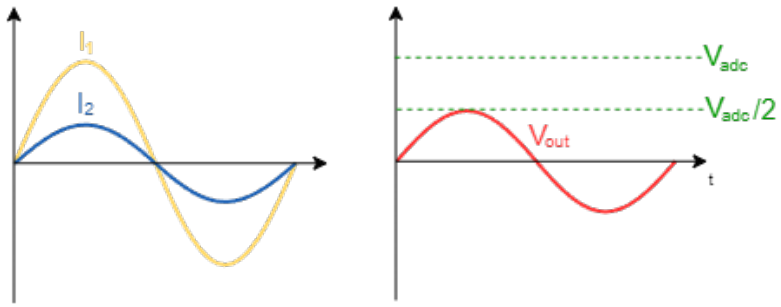


Fonte: O Autor (2024).

Figura 11 – Gráficos de corrente e tensão no TC.

(a) Corrente no primário e no secundário do TC.

(b) Tensão gerada no resistor de carga.



Fonte: O Autor (2024).

Desse modo, para calcular o valor do resistor deve-se:

1. Calcular a tensão de pico no primário do TC:

$$I_{1,pico} = \sqrt{2} I_{1,rms} \tag{2.2}$$

2. Calcular a corrente de pico na saída do TC através da relação de transformação (Equação 2.1):

$$I_{2,pico} = \frac{N_1 I_{1,pico}}{N_2} \tag{2.3}$$

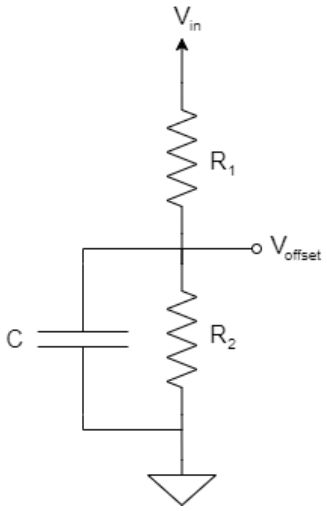
3. Através da lei de Ohm, encontrar o valor do resistor que crie uma queda de tensão igual a metade da tensão de referência do ADC quando a corrente de pico do secundário passa por ele:

$$R_b = \frac{V_{2,pico}}{I_{2,pico}} = \frac{V_{adc}}{2 I_{2,pico}} \tag{2.4}$$

Em seguida, utiliza-se um circuito somador para adicionar ao sinal uma tensão constante, conhecida como tensão de *offset*. Dessa forma, é realizado um deslocamento de nível do sinal, tornando-o estritamente positivo. A tensão de *offset* foi gerada a partir de um circuito divisor de tensão adicionando um capacitor de filtro (Figura 12) e pode ser calculada pela Equação 2.5.

$$V_{offset} = V_{in}\frac{R_2}{R_1 + R_2} \tag{2.5}$$

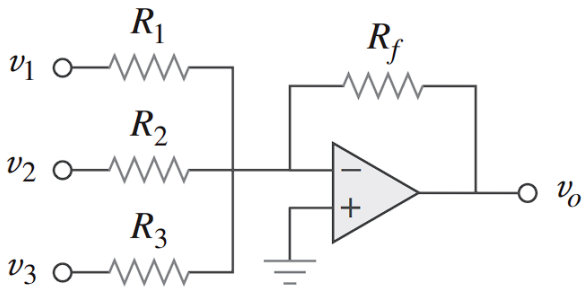
Figura 12 – Divisor de tensão.



Fonte: O Autor (2024).

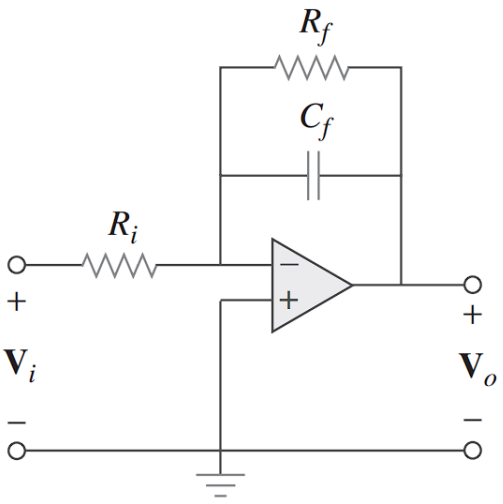
Além disso, aplicou-se um filtro passa-baixa para eliminar possíveis ruídos inerentes ao sinal antes de serem captados pelo ADC. Então, foi desenvolvido um circuito baseado no amplificador somador (Figura 13) em conjunto com um filtro ativo de primeira ordem (Figura 14).

Figura 13 – Amplificador somador.



Fonte: (ALEXANDER, 2014).

Figura 14 – Filtro passa-baixa ativo de primeira ordem.



Fonte: (ALEXANDER, 2014).

Como mostrado em (ALEXANDER, 2014), a tensão de saída do amplificador somador e a frequência de corte (f_c) do filtro passa baixa são determinados pelas Equações 2.6 e 2.7 respectivamente.

$$V_o = -R_f\left(\frac{V_1}{R_1} + \frac{V_2}{R_2} + \frac{V_3}{R_3}\right) \tag{2.6}$$

$$f_c = \frac{1}{2\pi R_f C_f} \tag{2.7}$$

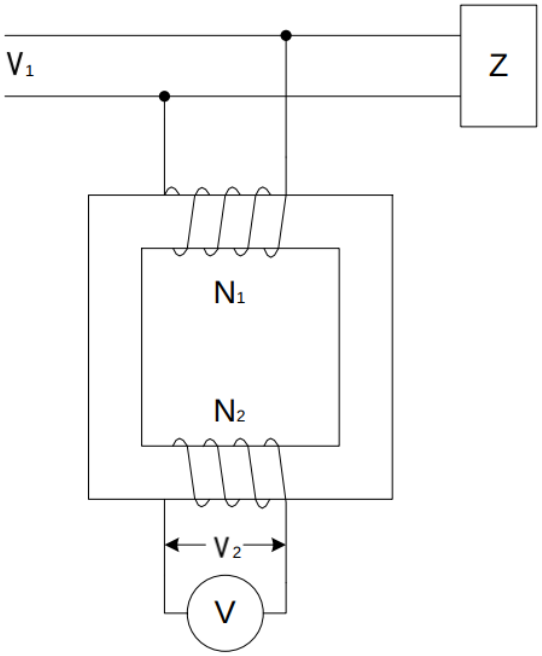
2.6 MEDIÇÃO DE TENSÃO

A estratégia para medição de tensão é reduzir o potencial de entrada da rede para um nível que possa ser suportado pelo medidor. Neste trabalho, o sensoriamento foi realizado utilizando um transformador de potencial, na fase de prova de conceito, e uma malha de resistência com amplificador diferencial, no projeto da PCB.

2.6.1 Transformador de Potencial

O transformador de potencial (TP) é um instrumento desenvolvido para controle e medição de potencial elétrico, utilizado com intuito de reduzir a tensão primária do sistema a valores aceitáveis para o dispositivo de medição. Dessa forma, como descrito em (FILHO, 1980), o TP é um transformador especialmente enrolado com um número de espiras no primário superior ao secundário (Figura 15). Assim como os TCs, eles promovem isolamento elétrico entre a tensão medida e o sinal de saída.

Figura 15 – Esquemático de um transformador de potencial.



Fonte: Adaptado de (FILHO, 1980).

Como sua única finalidade é fornecer amostras de tensão do sistema monitorado, o TP apresenta uma potência nominal muito baixa, sendo altamente preciso a fim de não distorcer os valores verdadeiros de tensão (CHAPMAN, 2013). Assim, um transformador ideal de potência construído com uma relação de espiras N_1:N_2 possui a relação de transformação de tensão do primário para o secundário dada por:

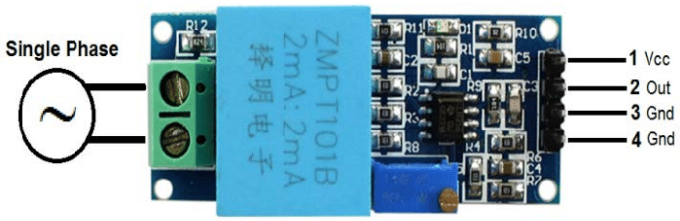
$$\frac{V_1}{V_2} = \frac{N_1}{N_2} \tag{2.8}$$

O condicionamento de sinais de TP é semelhante ao de um TC, com exceção do resistor de carga. Isso porque o sinal de saída no secundário do TP já é um sinal de tensão, ou seja, não é preciso realizar uma conversão de grandeza dos sinais. Dessa forma, o circuito de condicionamento adiciona uma tensão de *offset* e realiza a filtração dos sinais, como detalhado na Seção 2.5.2.

2.6.1.1 ZMPT101B

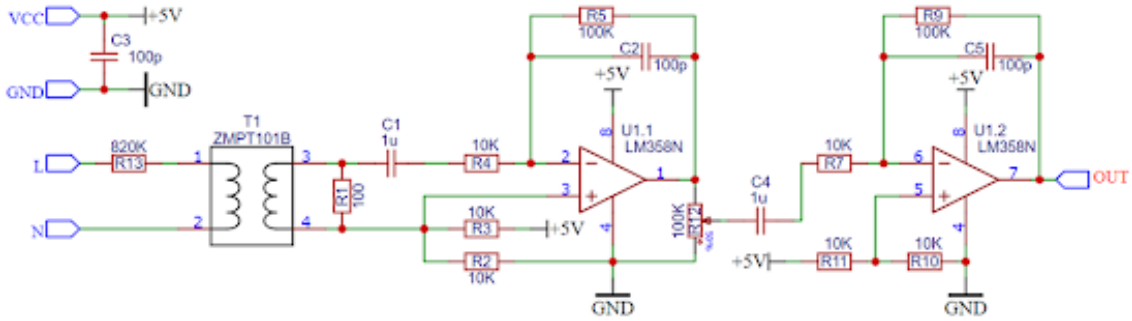
A fim de facilitar e agilizar a prova de conceito, foi utilizado o módulo de sensoriamento de tensão AC (Figura 16), que utiliza o transformador de potencial ZMPT101B. O ZMPT101B é um transformador de tensão de alta precisão, alto isolamento elétrico e extensa faixa de medição (INNOVATORSGURU, 2024). Além disso, esse módulo já contém todo o circuito de condicionamento necessário para adquirir amostras de tensão utilizando um microcontrolador (Figura 17).

Figura 16 – Módulo sensor de tensão AC.



Fonte: (TASTAN, 2019).

Figura 17 – Esquemático do módulo sensor de tensão AC.



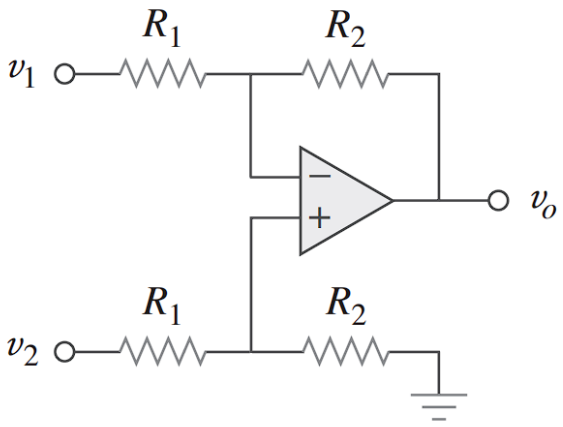
Fonte: (KOYANAGI, 2022).

2.6.2 Medição Diferencial de Tensão

A técnica de medição diferencial de tensão utiliza um amplificador diferencial para gerar uma atenuação no sinal de entrada para níveis que possam ser lidos pelo microcontrolador. O circuito de um amplificador diferencial (Figura 18) é apresentado em (JR, 2015) e possui a relação entre entrada e saída dada pela Equação 2.9.

$$V_o = \frac{R_2}{R_1}(V_2 - V_1) \tag{2.9}$$

Figura 18 – Amplificador diferencial.



Fonte: (ALEXANDER, 2014).

Como não há isolamento elétrico na medição diferencial de tensão, os resistores de valor R_1 foram substituídos por uma malha resistiva[2]. Assim como na medição de corrente, foram acoplados ao amplificador diferencial um divisor de tensão para gerar o deslocamento de nível do sinal e um capacitor para realizar a filtração (Seção 2.5.2).

2.7 PROCESSAMENTO DOS DADOS

Após a amostragem dos sinais, os valores de tensão e corrente adquiridos são processados para calcular as principais grandezas elétricas — como tensão e corrente eficaz, energia, potência, fator de potência, frequência, entre outros — que um medidor de energia deve registrar. Dessa forma, esses parâmetros são calculados utilizando as formulações definidas em (ALEXANDER, 2014):

- **Tensão e Corrente Eficaz**: o valor eficaz de um sinal periódico é a raiz do valor médio quadrático. Assim, o valor eficaz de um sinal de tensão e corrente com N amostras é dado por:

$$V_{rms} = \sqrt{\frac{1}{N}\sum_{N} v^2(n)} \tag{2.10}$$

$$I_{rms} = \sqrt{\frac{1}{N}\sum_{N} i^2(n)} \tag{2.11}$$

- **Potência Ativa**: a potência instantânea do sistema é o produto entre a tensão e a corrente medida em um determinado instante de tempo. Dessa forma, a potência ativa de um sinal de tensão e corrente de N amostras é calculada pela média das potências instantâneas em cada amostra (Equação 2.12).

$$P = \frac{1}{N}\sum_{N} v(n)i(n) \tag{2.12}$$

- **Potência Aparente**: a potência aparente é o produto dos valores eficazes da tensão e da corrente. Assim, a partir dos valores encontrados pelas Equações 2.10 e 2.11, encontra-se a potência aparente por meio de:

$$S = V_{rms}I_{rms} \tag{2.13}$$

- **Potência Reativa**: a potência reativa é dada por:

$$Q = \sqrt{S^2 - P^2} \tag{2.14}$$

[2] As malhas resistivas possuem o mesmo princípio de funcionamento de um divisor de tensão. Portanto, elas consistem em resistores associados em série de modo a proporcionar uma queda de tensão para que o potencial ao fim da malha seja suficientemente inferior ao valor de entrada.

- **Fator de Potência**: encontradas as potências ativa e aparente, o fator de potência é definido pela expressão:

$$FP = \frac{P}{S} \tag{2.15}$$

- **Energia**: o consumo ou geração do sistema são calculados através da equação:

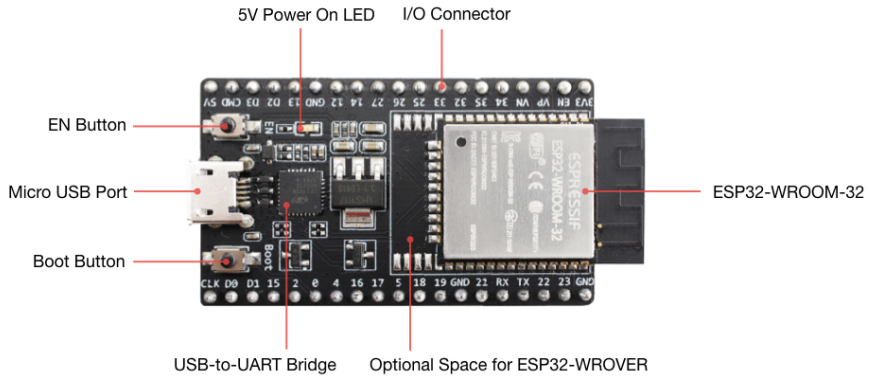
$$E = E_{acumulada} + P\Delta t \tag{2.16}$$

2.8 MICROCONTROLADORES ESP32

O ESP32 é uma família de microcontroladores com conectividade integrada Wi-Fi e Bluetooth em um único chip (SYSTEMS, 2024a). Nesse contexto, esses módulos são escolhas ideais para uma ampla variedade de aplicações IoT, abrangendo automação residencial, construção inteligente, medidores de consumo, sendo especialmente adequados para aplicações em espaços compactos.

Para elaboração da prova de conceito (Seção 3.2.1.5), foi utilizado o kit de desenvolvimento ESP32 DevkitC, pois possui o *hardware* necessário para medição, integração e testes, como mostrado na Figura 19.

Figura 19 – ESP32 DevkitC.



Fonte: (SYSTEMS, 2023).

No projeto da PCB (Seção 3.2.2), o módulo escolhido foi o ESP32 MINI (Figura 20) que possui tamanho reduzido e contém o ESP32-U4WDH, uma CPU de dois núcleos com arquitetura Xtensa de 32 bits LX6 que opera a até 240 MHz. Além disso, este chip tem capacidade de memória *flash* de 4 MB, oferece 28 GPIOs e integra um conjunto amplo de periféricos, como sensor de toque capacitivo, interface de cartão SD, ADC, Ethernet, SPI de alta velocidade, UART, I2S, I2C, entre outros. Além disso, possui uma antena PCB integrada na placa para comunicação de radiofrequência Wifi de 2,4 GHz (SYSTEMS, 2023).

Figura 20 – Microcontrolador ESP32 Mini.

ESP32-MINI-1

Fonte: (SYSTEMS, 2023).

Os periféricos utilizados neste trabalho foram o ADC (*Analogic to Digital Converter*), RTC (*Real Time Clock*), GPIO (*General Purpose Input/Output*) e comunicação serial UART (*Universal Asynchronous Receiver-Transmitter*), além da comunicação Wifi 2,4 GHz (Seção 3.2.2.1).

2.9 PROTOCOLO MQTT

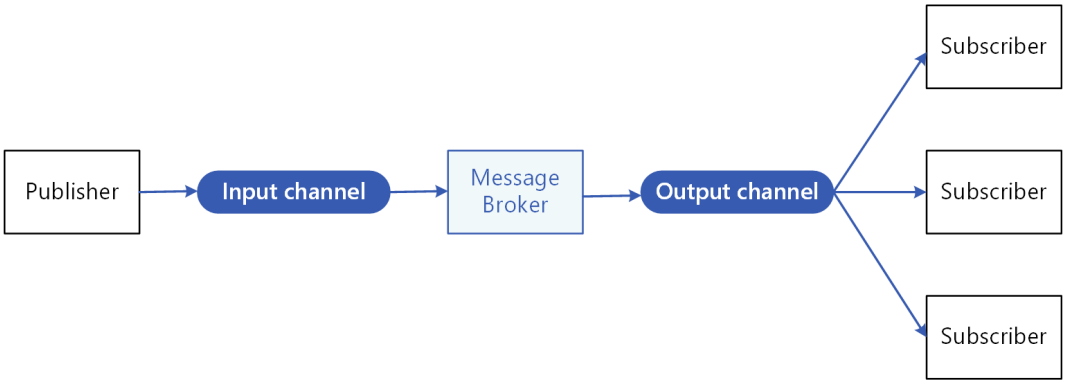
O MQTT (*Message Queuing Telemetry Transport*) é um protocolo de mensagens utilizado para comunicação de máquina para máquina (M2M - *Machine to Machine*) que funciona estruturado no protocolo TCP/IP. O MQTT permite a transmissão e recebimento de dados por meio de uma rede com largura de banda e recursos limitados, sendo ideal para dispositivos da Internet das Coisas (IoT), como sensores inteligentes. Além disso, ele oferece suporte à comunicação bilateral, dos dispositivos para a nuvem e da nuvem para os dispositivos, com baixo consumo de dados e fácil implementação (AWS, 2023b).

2.9.1 Arquitetura do MQTT

A comunicação em rede tradicional funciona de acordo com a arquitetura cliente-servidor (Seção 2.11). Os clientes solicitam recursos ou dados do servidor e o servidor processa e envia uma resposta, ou seja, clientes e servidores se comunicam diretamente entre si. No entanto, o

protocolo MQTT funciona conforme o modelo de publicação-assinatura ou *publisher-subscriber* (Figura 21), que desacopla o remetente da mensagem (publicador ou *publisher*) do destinatário da mensagem (assinante ou *subscriber*), como descrito em (AWS, 2023b).

Figura 21 – Modelo Publicação-Assinatura.



Fonte: (MICROSOFT, 2024).

Nessa perspectiva, é necessário um terceiro componente, chamado agente ou *broker* de mensagens, que gerencia a comunicação entre publicadores e assinantes. O trabalho do agente é filtrar todas as mensagens recebidas dos publicadores e distribuí-las corretamente aos assinantes (AWS, 2023b). Dessa forma, o agente desacopla os publicadores e assinantes de três modos:

- **Desacoplamento espacial:** O publicador e o assinante não estão cientes da localização da rede um do outro e não trocam informações como endereços IP ou números de porta;
- **Desacoplamento temporal:** O publicador e o assinante não executam nem têm conectividade de rede ao mesmo tempo;
- **Desacoplamento de sincronização:** O publicador e o assinante podem enviar ou receber mensagens sem interromper um ao outro. Por exemplo, o assinante não precisa esperar que o publicador envie uma mensagem.

2.9.2 Atores do MQTT

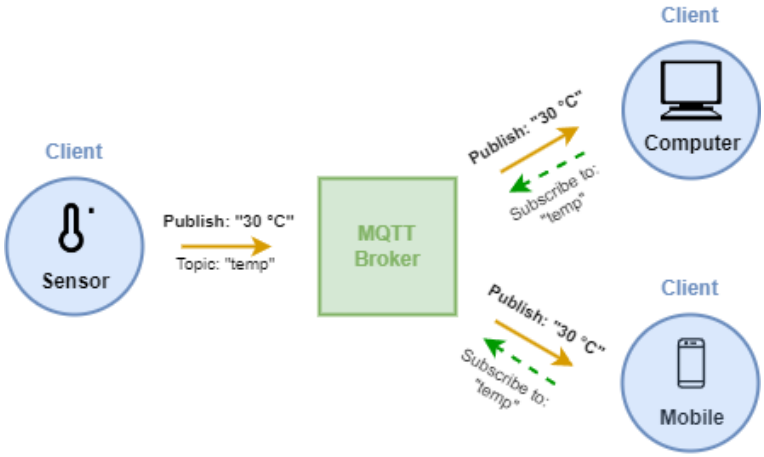
A arquitetura publicação/assinatura define clientes e agentes. Um cliente MQTT é qualquer dispositivo que se comunique por MQTT. Desse modo, se o cliente estiver enviando mensagens, ele funcionará como um publicador; caso esteja recebendo mensagens, atuará como assinante (AWS, 2023b).

O agente MQTT é o servidor que coordena mensagens entre os diferentes clientes. Portanto, ele é responsável por receber e filtrar mensagens, autorizar e autenticar clientes, identificar os clientes inscritos em cada mensagem e distribuí-las corretamente, transmitir mensagens a outros sistemas para análise posterior e solucionar mensagens e sessões de clientes perdidas (AWS, 2023b).

2.9.3 Mensagens MQTT

Cada mensagem enviada em MQTT é associada a um tópico, que identifica o canal ou categoria ao qual a mensagem pertence. Os tópicos são utilizados pelo agente MQTT para filtrar e direcionar as mensagens para os clientes (AWS, 2023b). Dessa forma, um dispositivo publica uma mensagem em um tópico e o *broker* encaminha a mensagem para todos os dispositivos inscritos naquele tópico (Figura 22). Alguns de tipos de mensagens do protocolo podem ser visualizados na Tabela 2.

Figura 22 – Fluxo de informação no protocolo MQTT.



Fonte: O Autor (2024).

Tabela 2 – Principais mensagens do protocolo MQTT.

Nome	Direção	Descrição
CONNECT	Cliente para Servidor	Requisição do cliente para conectar ao servidor
DISCONNECT	Cliente para Servidor	Cliente está desconectado
SUBSCRIBE	Cliente para Servidor	Pedido de inscrição
PUBLISH	Cliente para Servidor ou Servidor para Cliente	Publicar mensagem

Assim, os clientes podem enviar ao agente uma mensagem SUBSCRIBE para assinar um tópico de interesse ou uma mensagem PUBLISH para transmitir informações em um tópico específico. O formato dos dados enviados na mensagem é definido pelo cliente, como dados de texto, dados binários ou JSON[3] (AWS, 2023b).

Em suma, para realizar uma comunicação via protocolo MQTT, um cliente estabelece uma conexão com o agente, depois de conectado, ele pode publicar mensagens, assinar tópicos ou ambos. Então, quando um dispositivo publica uma mensagem, o *broker* a encaminha para todos que estão inscritos no tópico.

[3] Conforme definido em (JSON, 2024), JSON (JavaScript Object Notation) é uma formatação leve de troca de dados construído em formato texto e completamente independente de linguagem.

2.10 *WORLD WIDE WEB*

Em 1989, Tim Berners-Lee propôs a arquitetura de um sistema de informação global que permite o acesso e a interação com documentos, imagens, vídeos e outros recursos multimídia através da Internet[4] (KUROSE, 2021). Esse sistema recebeu a denominação de *World Wide Web*, popularmente conhecida como WWW ou simplesmente Web.

2.10.1 Páginas Web

A Web funciona por meio de uma coleção de documentos, geralmente chamados páginas Web. Cada página Web pode ser interligada por hiperlinks[5], ou seja, cada uma pode conter links para outras páginas em qualquer lugar do mundo. Assim, os usuários podem seguir um link que os levará até a página indicada. Esse processo pode ser repetido indefinidamente (TANENBAUM, 2021).

2.10.2 Navegadores Web

As páginas Web geralmente são visualizadas com o auxílio de um programa denominado navegador Web. O navegador busca a página solicitada, interpreta seu conteúdo e exibe a página na tela do computador de modo apropriado. São exemplos de navegadores Web o Mozilla Firefox, Microsoft Edge e Google Chrome (TANENBAUM, 2021).

2.10.3 Arquitetura da Web

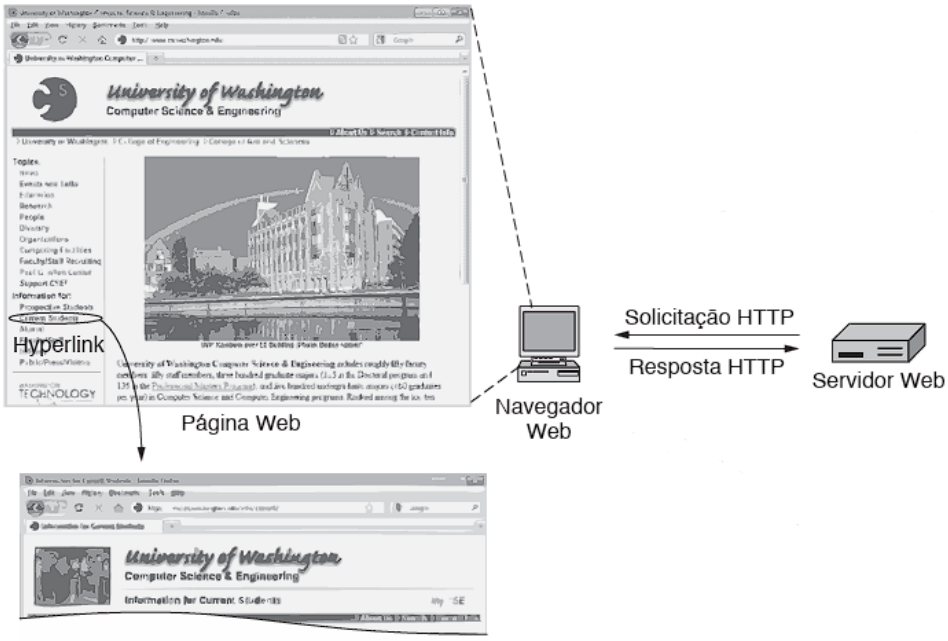
Como exposto em (TANENBAUM, 2021), a Figura 23 apresenta o modelo básico para exibição de páginas Web. O navegador exibe uma página Web na máquina do cliente e cada página envia uma solicitação ao servidor, que responde com o conteúdo da página. O protocolo de solicitação e resposta para buscar páginas é baseado em texto e é conhecido HTTP (HyperText Transfer Protocol), detalhado na Seção 2.11.

---

[4] A Internet é uma rede de computadores que interconecta centenas de milhões de dispositivos de computação ao redor do mundo (KUROSE, 2021).

[5] Os hiperlinks são partes de página Web, como textos, ícones e imagens, que estão associadas a links para outras páginas (TANENBAUM, 2021).

Figura 23 – Arquitetura da Web.



Fonte: Adaptado de (TANENBAUM, 2021).

2.10.4 Localizador Uniforme de Recursos

Para localizar e acessar uma página Web, associa-se a cada uma delas um único Uniform Resource Locator (Localizador Uniforme de Recursos), ou URL. Ou seja, cada página ou recurso na internet é vinculado a uma URL distinta, que representa um endereço utilizado para identificá-los e acessá-los (TANENBAUM, 2021). Um exemplo de URL é dado por:

http://www.exemplo.com/pagina

Um URL como o de exemplo pode ser dividido em três partes: o protocolo (http), o nome DNS[6] do host (www.exemplo.com) e o nome do caminho (pagina).

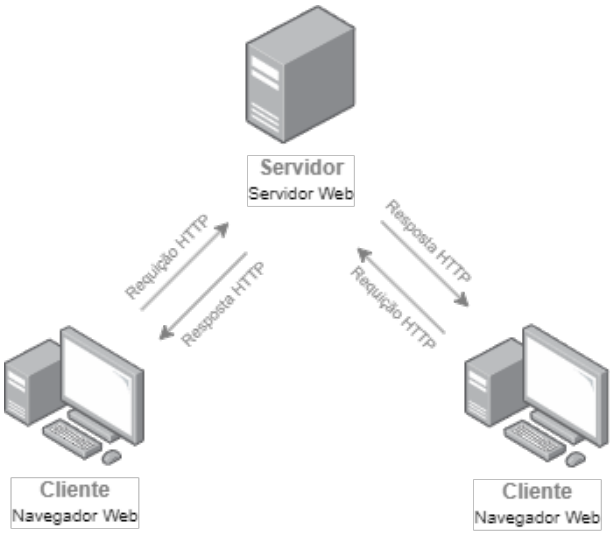
2.11 PROTOCOLO HTTP

O HTTP, ou Protocolo de Transferência de Hipertexto (HyperText Transfer Protocol), é um protocolo da camada de aplicação e é usado para transferir informações na World Wide Web. O HTTP opera em dois programas em execução que estão em sistemas finais distintos: um cliente e um servidor. Estes programas interagem por meio da troca de mensagens HTTP, onde o protocolo estabelece a estrutura dessas mensagens e a maneira como cliente e servidor as compartilham (KUROSE, 2021).

[6] O sistema de nomes de domínio, ou DNS (Domain Name System), foi criado para relacionar nomes de hosts, que são mais amigáveis e fáceis de lembrar, com endereços IP. (TANENBAUM, 2021).

Em outras palavras, ele é baseado na arquitetura cliente-servidor[7] (Figura 24), em que o cliente inicia uma conexão requisitando algum objeto do servidor através de um URL[8] e este responde com o objeto ou mensagens sobre o processamento da requisição, como mensagens de erro. O cliente pode ser um navegador web e o servidor geralmente é um servidor web (KUROSE, 2021).

Figura 24 – Arquitetura cliente-servidor.



Fonte: O Autor (2024).

Além disso, o HTTP é um protocolo sem estado (stateless), o que significa que cada solicitação dos clientes é tratada de forma independente. Isto é, o servidor não mantém informação sobre o estado anterior (KUROSE, 2021).

2.11.1 Mensagens HTTP

Conforme definido em (KUROSE, 2021), as mensagens HTTP são as requisições e as respostas trocadas entre cliente e servidor. Elas possuem uma estrutura básica dividida em três seções: linha inicial, linhas de cabeçalhos e corpo da entidade. O conteúdo dessas seções varia de acordo com o tipo de mensagem.

[7] Em uma arquitetura cliente-servidor há um hospedeiro sempre em funcionamento, denominado servidor, que atende a requisições de muitos outros hospedeiros, denominados clientes. Quando recebe uma requisição de um objeto de um hospedeiro cliente, um servidor web responde enviando o objeto solicitado (KUROSE, 2021).

[8] O URL especifica o caminho para o recurso. Um URL é semelhante ao endereço do site que você digita no navegador para visitar qualquer página da Web. O URL também é chamado de *endpoint* de solicitação e apresenta ao servidor o que o cliente requer.

2.11.1.1 Mensagens de Requisição

Uma requisição é um pedido enviado pelo cliente para solicitar algum recurso do servidor. Uma mensagem de requisição é estruturada da seguinte forma:

- **Linha de Requisição:** Contém o método HTTP — que indica a ação que deve ser executada (Tabela 3) —, o caminho para esse recurso ou URL e a versão do protocolo que está sendo utilizado;

Tabela 3 – Métodos HTTP mais utilizados.

Método	**Descrição**
GET	Solicita dados de um recurso específico, como a lista de estudantes de uma disciplina.
POST	Envia dados para serem processados a um recurso específico. Por exemplo, envio de dados para cadastro de um aluno em curso.
PUT	Atualiza um recurso existente, como a nota de um aluno em uma prova.
DELETE	Remove um recurso. Por exemplo, uma ação de desmatricular um aluno de uma disciplina.

- **Cabeçalhos (*Headers*):** Fornece informações adicionais sobre a solicitação, ajudando na interpretação e no processamento. Por exemplo, pode-se enviar credenciais para autenticação no servidor;
- **Corpo da Entidade (*Body*):** Inclui os dados propriamente ditos que serão processados no servidor. Como nem toda requisição envia dados para serem processados, este campo é opcional. Um exemplo de *Body* são dados enviados para preenchimento de um formulário.

2.11.1.2 Mensagens de Resposta

No lado do servidor, ele é responsável por processar a requisição do cliente e enviar uma resposta. Assim como a requisição, a resposta pode ser dividida em:

- **Linha de Status:** Engloba o código de status (Tabela 4) e a versão do protocolo;
- **Cabeçalhos (*Headers*):** Fornece informações sobre a resposta. Como por exemplo a data e hora que a resposta foi gerada;
- **Corpo da Entidade (*Body*):** Inclui os dados retornados pelo servidor. Por exemplo, a lista de estudantes solicitada pela requisição de método GET.

Tabela 4 – Códigos HTTP mais utilizados.

Código	**Descrição**
2xx (Sucesso)	Indica que a solicitação foi recebida, compreendida e aceita com sucesso. Por exemplo, o código 200 significa que a solicitação foi bem-sucedida.
3xx (Redirecionamento)	Indica que o cliente precisa realizar ações adicionais para concluir a solicitação.
4xx (Erro do Cliente)	Indica que ocorreu um erro por parte do cliente. O mais famoso é o "404 Not Found"que indica recurso solicitado não foi encontrado pelo servidor.
5xx (Erro do Servidor)	Indica que ocorreu um erro no servidor ao processar a solicitação.

2.12 INTERFACE DE PROGRAMAÇÃO DE APLICAÇÃO

Interface de Programação de Aplicação (*Application Programming Interface*) ou API são mecanismos que permitem que dois componentes de software se comuniquem usando um conjunto de definições e protocolos (AWS, 2023c).

No contexto de APIs, a palavra Aplicação refere-se a qualquer software com uma função distinta. A interface pode ser pensada como um contrato de serviço entre duas aplicações. Esse contrato define como as duas se comunicam usando solicitações e respostas (AWS, 2023c).

A arquitetura da API geralmente é explicada em termos de cliente e servidor. A aplicação que envia a solicitação é chamada de cliente e a aplicação que envia a resposta é chamada de servidor (AWS, 2023c). Você pode pensar em uma API da Web como um gateway entre clientes e recursos na Web.

2.12.1 APIs RESTful

A representational state transfer (REST – transferência de estado representacional) é uma arquitetura de software baseada no protocolo HTTP que impõe condições sobre como uma API deve funcionar. A REST foi criada inicialmente como uma diretriz para gerenciar a comunicação em uma rede complexa como a internet, possibilitando uma comunicação confiável e de alta performance em escala (AWS, 2023d).

As APIs que seguem o estilo de arquitetura REST são chamadas de APIs REST. Os serviços da Web que implementam a arquitetura REST são chamados de serviços da Web RESTful. O termo API RESTful geralmente se refere a APIs da Web RESTful. No entanto, pode-se usar os termos API REST e API RESTful de forma intercambiável (AWS, 2023d).

2.12.1.1 Modo de Funcionamento

A função básica de uma API RESTful é permitir que os solicitantes acessem os recursos desejados. O cliente[9] entra em contato com o servidor usando a API quando requer um recurso[10]. Como descrito em (AWS, 2023d), as etapas gerais para uma chamada de uma API RESTful são:

1. O cliente envia uma solicitação ao servidor. O cliente segue a documentação da API para formatar a solicitação de modo que o servidor entenda;
2. O servidor autentica o cliente e confirma que o cliente tem o direito de fazer essa solicitação;
3. O servidor recebe a solicitação e a processa internamente;
4. O servidor retorna uma resposta ao cliente. A resposta contém informações que indicam ao cliente se a solicitação foi bem-sucedida. A resposta também inclui informações solicitadas pelo cliente.

2.12.1.2 Chamadas de API RESTFul

Como dito anteriormente, uma API RESTFul é envolvida pelo HTTP. Dessa forma, as solicitações do cliente (Tabela 5) e as respostas dos servidores (Tabela 6) possuem uma estrutura básica semelhante à do protocolo.

O identificador de recurso exclusivo é chamado de *endpoint*, ou seja, cada *endpoint* é uma URL que fornece a localização de um recurso no servidor da API. Um exemplo de *endpoint* é:

http://endpoint.com/dispositivo

Além disso, uma API pode receber uma requisição de diferentes métodos no mesmo *endpoint*. Assim, uma rota é uma associação entre um método HTTP e uma URL responsável por processar e gerar a resposta dessa requisição. A Tabela 7 exemplifica como as rotas se associam a *endpoint* e métodos HTTP para realizar uma função específica.

---

[9] Clientes são usuários que desejam acessar informações da Web. O cliente pode ser uma pessoa ou um sistema de software que usa a API (AWS, 2023d).

[10] Recursos são as informações que diferentes aplicações fornecem aos seus clientes. Os recursos podem ser imagens, vídeos, textos, números ou qualquer tipo de dado. A máquina que fornece o recurso ao cliente também é chamada de servidor. As organizações usam APIs para compartilhar recursos e fornecer serviços da Web, mantendo a segurança, o controle e a autenticação. Além disso, as APIs ajudam a determinar quais clientes têm acesso a recursos internos específicos (AWS, 2023d).

Tabela 5 – Componentes principais de uma solicitação do cliente da API RESTful.

Componente	**Descrição**
Identificador de recurso exclusivo (*Endpoint*)	O servidor identifica cada recurso com identificadores de recursos exclusivos. Para serviços REST, o servidor normalmente realiza a identificação de recursos usando um uniform resource locator (URL – localizador de recurso uniforme).
Método	Um método HTTP que informa ao servidor o que ele precisa fazer com o recurso, como por exemplo: GET, POST, PUT e DELETE.
Cabeçalhos HTTP	São os metadados trocados entre o cliente e o servidor. Por exemplo, o cabeçalho da solicitação indica o formato dos dados da requisição e do corpo da resposta.
Dados	Informações necessárias para POST, PUT e outros métodos HTTP funcionarem com êxito.
Parâmetros	Os parâmetros que fornecem ao servidor mais detalhes sobre o que precisa ser feito.

Tabela 6 – Componentes principais de uma resposta do servidor da API RESTful.

Componente	**Descrição**
Linha de status	A linha de status contém um código de status HTTP (4) que comunica o sucesso ou a falha da solicitação.
Corpo da mensagem	Representação do recurso. O servidor seleciona um formato de representação apropriado com base no que os cabeçalhos da solicitação contêm. Neste trabalho, as solicitações são no JSON.
Cabeçalhos HTTP	Contém cabeçalhos ou metadados sobre a resposta, que fornecem mais contexto sobre a resposta e incluem informações como servidor, codificação, data e tipo de conteúdo.

Tabela 7 – Exemplos de rotas de um *endpoint*.

Roteamento		
Método	**Endpoint**	**Objetivo**
GET	http://endpoint.com/dispositivo	Consultar a lista de dispositivos
POST	http://endpoint.com/dispositivo	Criar um novo dispositivo

2.13 BANCO DE DADOS

Um banco de dados é uma coleção de dados relacionados. Entende-se por dados as informações que possuem um significado implícito e que podem ser gravadas. De modo mais detalhado, um banco de dados deve representar aspectos do mundo real e conter dados logicamente organizados, coerentes e com significado próprio. Dessa forma, ele é criado para atender a uma finalidade específica — sendo planejado, construído e preenchido com dados — e ser utilizado por um grupo específico de pessoas, com aplicações projetadas para atender aos seus interesses (ELMASRI, 2019).

2.13.1 Sistema Gerenciador de Banco de Dados

Um sistema gerenciador de banco de dados (SGBD) é uma coleção de programas que permite aos usuários criar e manter um banco de dados. O SGBD é, portanto, um sistema de software que facilita os processos de definição, construção, manipulação e compartilhamento de bancos de dados entre vários usuários e aplicações. Outras funções importantes do SGBD são a proteção e a manutenção do banco de dados por longos períodos (ELMASRI, 2019).

2.13.2 Modelos de Banco de Dados

Segundo (HEUSER, 2009), um modelo de banco de dados é uma descrição dos tipos de informações que estão armazenadas em um banco de dados. No projeto de banco de dados, normalmente são considerados dois níveis de abstração: o modelo conceitual e o modelo lógico.

O modelo conceitual é uma representação independente de implementação de um banco de dados em um SGBD. Em outras palavras, trata-se de um modelo de dados abstrato que descreve a estrutura do banco de dados sem depender de um SGBD específico. A técnica de modelagem conceitual utilizada neste trabalho é a abordagem entidade-relacionamento (ER). Nesta técnica, um modelo conceitual é usualmente representado através de um diagrama, chamado diagrama entidade-relacionamento (DER).

Um modelo lógico é uma descrição de um banco de dados no nível de abstração visto pelo usuário do SGBD. Assim, ele é dependente do tipo particular de SGBD que está sendo usado. Em suma, ele define como o banco de dados será implementado em um SGBD específico.

2.13.3 Modelo Entidade Relacionamento

O modelo Entidade-Relacionamento (ER), proposto por Peter Chen em 1976, baseia-se na percepção de um universo constituído por um grupo básico de objetos chamados entidades e por relacionamentos entre estes objetos (CHEN, 1976). Um esquema de dados feito com o modelo ER indica quais dados serão representados, não como serão armazenados. Nessa perspectiva, é apresentado o conceito de entidades, relacionamentos e atributos, além de uma notação gráfica para diagramas ER.

2.13.3.1 Entidades

De acordo com (HEUSER, 2009), uma entidade representa um conjunto de objetos da realidade modelada. Ou seja, ela é o objeto básico do modelo ER que representa algo do mundo real, com uma existência independente. Uma entidade pode ser um objeto com uma existência física — um livro ou uma pessoa — ou um objeto com uma existência conceitual — uma disciplina ou um departamento — (ELMASRI, 2019). Em um DER, uma entidade é representada através de um retângulo que contém o nome da entidade (Figura 25).

Figura 25 – Representação Gráfica de Entidades.

Fonte: O Autor (2024).

2.13.3.2 Relacionamentos

Um relacionamento é como duas entidades se associam. Em um DER, um relacionamento é representado através de um losango, ligado por linhas aos retângulos representativos das entidades que participam do relacionamento (HEUSER, 2009). A Figura 26 apresenta um DER, que exemplifica um relacionamento entre as entidades da Figura 25. Ou seja, a entidade aluno se relaciona com a entidade curso através de uma inscrição.

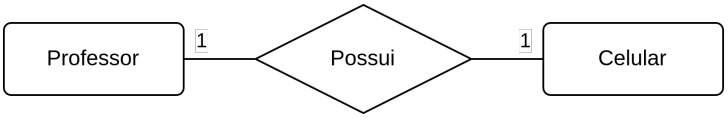
Figura 26 – Representação Gráfica de Relacionamentos entre Entidades.

Fonte: O Autor (2024).

Os relacionamentos podem ser classificados em:

- **Relacionamento 1:1 (um-para-um):** cada uma das duas entidades envolvidas se associa a apenas uma unidade da outra. Por exemplo, uma pessoa pode possuir somente um celular e um celular só pode pertencer a uma pessoa (Figura 27).

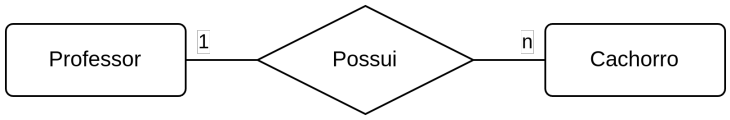
Figura 27 – Representação Gráfica de Relacionamento 1:1.



Fonte: O Autor (2024).

- **Relacionamento 1:n (um-para-muitos)**: uma das entidades envolvidas podem se associar a várias unidades da outra, porém, do outro lado cada uma das várias unidades associadas só podem estar ligadas a uma unidade da outra entidade. Por exemplo, uma pessoa pode ter vários cachorros, mas os cachorros só podem ter um dono (Figura 28).

Figura 28 – Representação Gráfica de Relacionamento 1:n.



Fonte: O Autor (2024).

- **Relacionamento n:n (muitos-para-muitos)**: neste tipo de relacionamento cada entidade, de ambos os lados, pode se associar a múltiplas unidades da outra. Por exemplo, um professor pode ensinar a vários alunos e um aluno pode ser ensinado por vários professores (Figura 29).

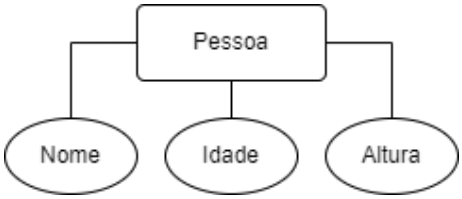
Figura 29 – Representação Gráfica de Relacionamento n:n.

Fonte: O Autor (2024).

2.13.3.3 Atributos

Os atributos são características e propriedades particulares de uma entidade. Uma dada entidade possui um valor para cada um de seus atributos. Os valores dos atributos que descrevem cada entidade são a maior parte dos dados armazenados no banco de dados (ELMASRI, 2019). O atributo também é utilizado para associar informações de relacionamento entre entidades (HEUSER, 2009). A Figura 30 mostra a representação gráfica dos atributos — Nome, Idade e Altura — da entidade Pessoa. Na prática, atributos não são representados graficamente, para não sobrecarregar os diagramas, já que muitas vezes entidades possuem um grande número de atributos.

Figura 30 – Representação Gráfica dos Atributos de uma Entidade.



Fonte: O Autor (2024).

2.13.4 Abordagem Relacional

O modelo relacional representa o banco de dados como uma coleção de tabelas, em que cada tabela consiste em linhas, que representam um fato que corresponde a uma entidade ou relacionamento do mundo real, e colunas representam os atributos ou características específicas (HEUSER, 2009).

Na terminologia do modelo relacional formal, uma linha é chamada tupla, um cabeçalho de coluna é conhecido como atributo, e a tabela é chamada relação. O tipo de dado que descreve os tipos de valores que podem aparecer em cada coluna é representado pelo domínio de valores possíveis (ELMASRI, 2019).

2.13.4.1 Chaves

O conceito básico para estabelecer relações entre linhas de tabelas de um banco de dados relacional é o da chave. Em um banco de dados relacional, há ao menos três tipos de chaves a considerar: a chave primária, a chave estrangeira e a chave alternativa (HEUSER, 2009).

- **Chave Primária:** Uma chave primária é uma coluna ou uma combinação de colunas cujos valores distinguem uma linha das demais dentro de uma tabela. Além disso, exige-se que a chave primária seja mínima. Uma chave é mínima quando todas suas colunas forem efetivamente necessárias para garantir o requisito de unicidade de valores da chave;
- **Chave Estrangeira:** Uma chave estrangeira é uma coluna ou uma combinação de colunas, cujos valores aparecem necessariamente na chave primária de uma tabela. A chave estrangeira é o mecanismo que permite a implementação de relacionamentos em um banco de dados relacional;
- **Chave Alternativa:** Em alguns casos, mais de uma coluna ou combinações de colunas podem servir para distinguir uma linha das demais. Uma das colunas (ou combinação de colunas) é escolhida como chave primária. As demais colunas ou combinações são denominadas chaves alternativas.

Neste trabalho será utilizada a abordagem relacional para modelagem de dados e o MySQL como sistema de gerenciamento de banco de dados.

2.14 COMPUTAÇÃO NA NUVEM

A computação em nuvem é a entrega sob demanda de poder computacional, banco de dados, armazenamento, aplicações e outros recursos de TI por meio de uma plataforma de serviços em nuvem pela Internet. Nesse contexto, uma plataforma de serviços em nuvem fornece acesso a infraestruturas flexíveis e de baixo custo, com acesso sob demanda e pago conforme o uso dos serviços. Essa abordagem é fundamental para o modelo de computação em nuvem (AWS, 2023a).

A nuvem AWS foi utilizada como infraestrutura de TI dos serviços, como servidores[11], APIs e banco de dados, implementados neste trabalho.

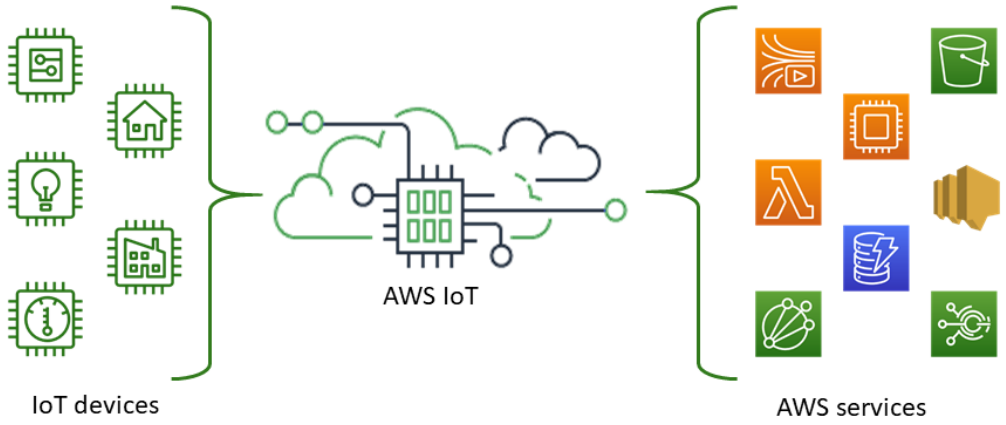
2.14.1 *Amazon Web Services*

A *Amazon Web Services* (AWS) é a plataforma de nuvem mais adotada e mais abrangente do mundo, ou seja, ela oferece serviços tecnológicos sob demanda pela Internet com preços conforme o uso. A Nuvem AWS abrange um amplo conjunto de soluções escaláveis, incluindo computação, armazenamento, bancos de dados, análises, redes, dispositivos móveis, ferramentas de desenvolvedor, ferramentas de gerenciamento, IoT e segurança (AWS, 2024a). Neste trabalho, foram utilizados os seguintes serviços: AWS IoT Core, AWS Lambda, Amazon EC2, Amazon RDS e Amazon API Gateway.

2.14.1.1 AWS IoT Core

Os serviços em nuvem fornecidos pela AWS IoT, como o AWS IoT Core, possibilitam a conexão de dispositivos a outros dispositivos ou a serviços em nuvem da AWS. Assim, caso os dispositivos possam se conectar à AWS IoT, esta pode conectá-los aos outros serviços em nuvem oferecidos pela AWS (Figura 31). Nessa perspectiva, o serviço AWS IoT Core é um servidor de gerenciamento de mensagens que oferece suporte aos protocolos MQTT, HTTPS e LoRaWAN, auxiliando na gestão e suporte de dispositivos em campo (AWS, 2024e).

Figura 31 – Funcionalidade do AWS IoT.



Fonte: (AWS, 2024e).

Dessa forma, como o protocolo MQTT foi adotado na comunicação do *smartmeter*, é utilizado o *broker* de mensagens MQTT (Seção 2.9.2) do AWS IoT Core para receber as

[11] Servidores referem-se a computadores ou sistemas dedicados a fornecer serviços, recursos ou informações. Existem diversos tipos de servidores, como servidores web, servidores de banco de dados, servidores de e-mail, entre outros.

mensagens enviadas pelo medidor e direcioná-las para outros serviços da AWS através de IoT Rules[12], bem como encaminhar as mensagens enviadas da aplicação web para dispositivo.

2.14.1.2 AWS Lambda

O AWS Lambda é um serviço de computação que executa um código, chamados de funções lambda, em resposta a eventos e gerencia automaticamente os recursos de computação. Isso porque ele realiza a execução em uma infraestrutura de computação de alta disponibilidade, realizando toda a administração dos recursos computacionais, incluindo manutenção de servidor e sistema operacional, provisionamento de capacidade, escalabilidade automática e registro (AWS, 2024f).

Portanto, é possível programar funções lambda utilizando linguagens como python e javascript que serão acionadas somente quando determinados eventos ocorrerem. Por exemplo, sempre que uma mensagem for recebida pelo *broker* MQTT do IoT Core, pode-se executar uma função lambda que armazena o conteúdo da mensagem em um banco de dados instanciado no Amazon RDS.

2.14.1.3 Amazon RDS

O *Amazon Relational Database Service* (Amazon RDS) é um serviço da Web que facilita a configuração, a operação e escalabilidade de um banco de dados relacional na Nuvem AWS. Ele fornece capacidade econômica e redimensionável para um banco de dados relacional e gerencia tarefas comuns de administração de banco de dados (AWS, 2024d). Neste trabalho, foi utilizado o Amazon RDS como ambiente em nuvem de execução do banco de dados modelado.

2.14.1.4 Amazon EC2

O Amazon Elastic Compute Cloud (Amazon EC2) oferece a plataforma de computação com diversas opções de processadores, armazenamentos, redes e sistemas operacionais. Desse modo, ele proporciona capacidade de computação escalável e sob demanda, reduzindo os custos de hardware, permitindo o desenvolvimento e implantação mais rápidos de aplicativos.

Assim, pode-se utilizar o Amazon EC2 para criar tantos servidores virtuais quanto necessário, configurar segurança e redes, e gerenciar armazenamento. Além disso, é possível adicionar mais capacidade para lidar com tarefas intensivas, como processos periódicos ou picos de tráfego em sites. Não obstante, quando o uso diminui, a capacidade pode ser reduzida novamente.

Em síntese, o Amazon EC2 possibilita que os usuários aluguem instâncias de computação virtual para executar suas próprias aplicações (AWS, 2024c). Os servidores desenvolvidos neste trabalho utilizam as instâncias da Amazon EC2 como ambiente de execução em nuvem.

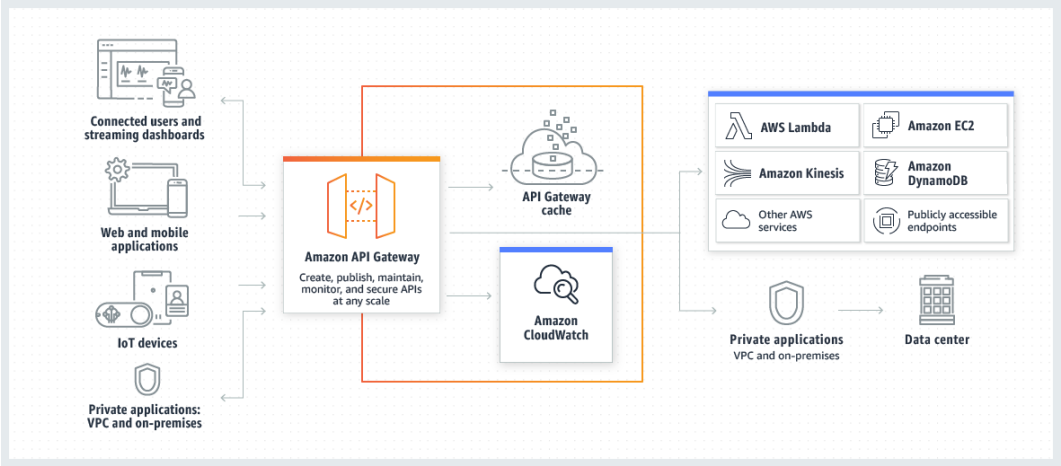
[12] Regras que definem como o *broker* deve direcionar as mensagens recebidas para outros serviços da AWS (AWS, 2024e).

2.14.1.5 Amazon API Gateway

O Amazon API Gateway é um serviço da AWS para criar, publicar, manter, monitorar e proteger APIs REST, HTTP e WebSocket. Dessa forma, ele pode criar APIs RESTful (Seção 2.12.1) que implementam métodos padrão do HTTP, como GET, POST, PUT e DELETE, oferecendo uma abordagem eficaz e padronizada para a construção de interfaces de programação de aplicações que são interoperáveis e escaláveis. As APIs criadas podem acessar serviços da AWS ou de outras fontes na Web, assim como dados armazenados na Nuvem da AWS (AWS, 2024b).

A Figura 32 ilustra como as APIs construídas no Amazon API Gateway proporcionam uma experiência integrada e consistente para a construção de aplicações serverless na AWS. Isso porque ele lida com todas as tarefas envolvidas na aceitação e processamento de chamadas de API simultâneas, tais como gerenciamento de tráfego, autorização e controle de acesso, monitoramento e gerenciamento de versão da API (AWS, 2024b).

Figura 32 – Diagrama de funcionamento do Amazon API Gateway.



Fonte: (AWS, 2023c).

Neste trabalho, o API Gateway atua como uma interface entre a aplicação Web e o AWS IoT. Ou seja, para enviar uma mensagem ao medidor, a aplicação realiza uma chamada para o API Gateway, que encaminha os dados para o *broker* MQTT. Este, por sua vez, direciona as informações para o dispositivo.

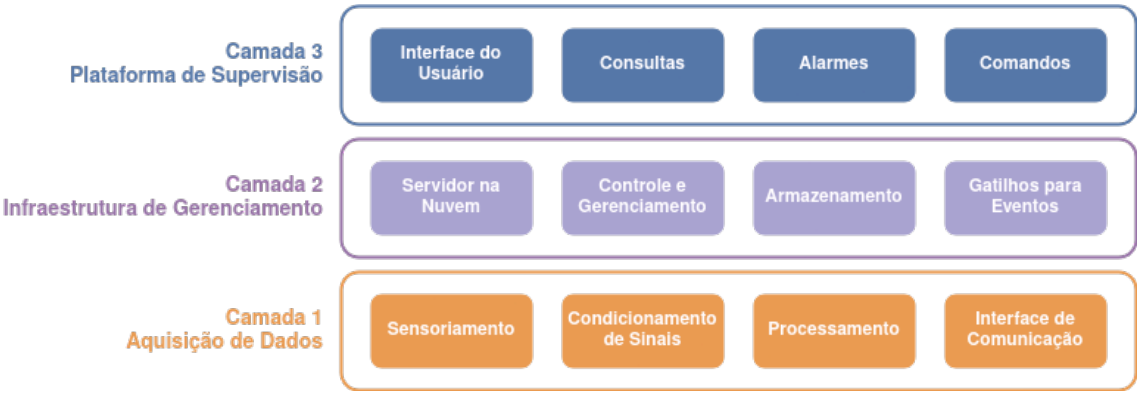
3 METODOLOGIA

3.1 ARQUITETURA GERAL DO SISTEMA

O sistema de monitoramento inteligente é composto por diversos módulos, tais como sensores, hardware, comunicações, sistema de alarmes e software de gerenciamento de dados. Esses elementos possibilitam comunicações bilaterais com o medidor, permitindo o envio de comandos para finalidades diversas, como monitoramento em tempo real, alteração da frequência de leituras e ajuste de calibrações, emissão de alertas, entre outras funcionalidades, como descrito em (LLORET et al., 2016).

Nessa perspectiva, (LLORET et al., 2016) propôs que esses sistemas sejam construídos em uma arquitetura de camadas, classificando os componentes e interfaces em diferentes categorias de acordo com suas características e propósitos. Assim, o sistema desenvolvido neste trabalho é dividido em três camadas: Aquisição de Dados, Infraestrutura de Gerenciamento e Plataforma de Supervisão (Figura 33).

Figura 33 – Arquitetura em camadas do sistema.



Fonte: O Autor (2024).

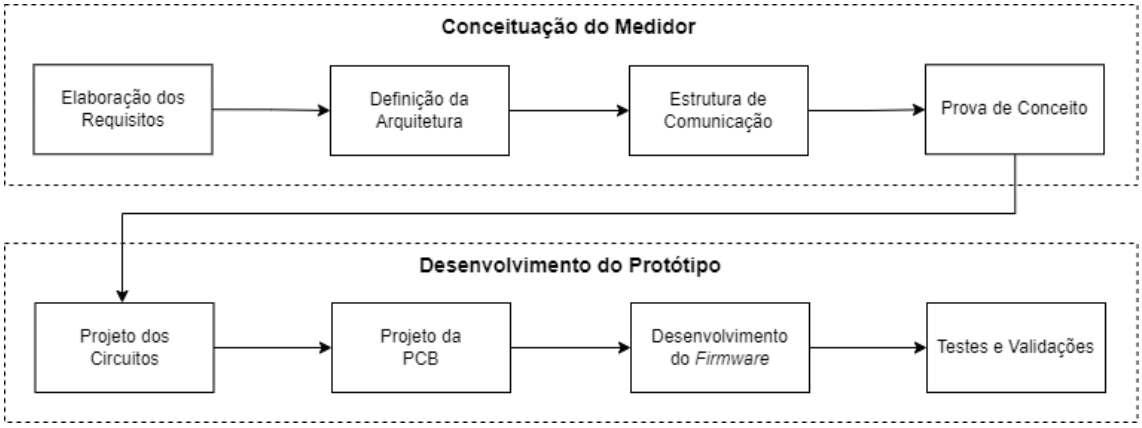
A camada de Aquisição de Dados é responsável por realizar o sensoriamento, condicionamento e filtração dos sinais, além de processar e transmitir os dados para a nuvem. Por sua vez, a Infraestrutura de Gerenciamento assume a função de receber, armazenar e interpretar as informações, desencadeando gatilhos para eventos na interface do usuário. Por fim, a Plataforma de Supervisão é composta pelos serviços necessários para o funcionamento da aplicação web.

3.2 AQUISIÇÃO DE DADOS

O medidor inteligente desempenha um papel fundamental ao obter dados de consumo e geração de energia. Além disso, o dispositivo registra em tempo real essas informações e permite comunicações bidirecionais entre o medidor e o sistema central, aprimorando a eficiência do monitoramento e gerenciamento da energia (ZHENG; GAO; LIN, 2013). Dessa forma, o desenvolvimento de um medidor inteligente é essencial para atender as funcionalidades da camada 1.

O projeto do *smartmeter* foi elaborado seguindo o fluxo da Figura 34. Inicialmente, foram definidos os requisitos de projeto do medidor, isto é, as grandezas a serem medidas, modo de funcionamento, tecnologia de comunicação, entre outros. Então, buscou-se uma validação preliminar através de uma prova de conceito, ou seja, foram utilizados módulos e placas de desenvolvimento comerciais para verificar se o sistema era capaz de realizar as medições e transmitir os dados para a nuvem. Em seguida, elaborou-se o projeto e esquemático dos circuitos de medição; o desenvolvimento do firmware[1]; e o *design* e a fabricação da placa de circuito impresso (PCB) do dispositivo. Por fim, foram conduzidas calibrações e testes para validar o protótipo.

Figura 34 – Metodologia de projeto do medidor inteligente.



Fonte: O Autor (2024).

3.2.1 Concepção do Medidor

O projeto conceitual do medidor objetivou descrever os principais requisitos e funcionalidades do dispositivo. Ou seja, foram definidas características fundamentais do funcionamento do medidor, tais como os parâmetros medidos, as formas de medição, a interface de comunicação, entre outros.

[1] O *firmware* é o *software* no dispositivo de *hardware* que executa funções como tarefas básicas de entrada/saída e oferece as instruções necessárias para que o dispositivo se comunique com outros.

3.2.1.1 Requisitos

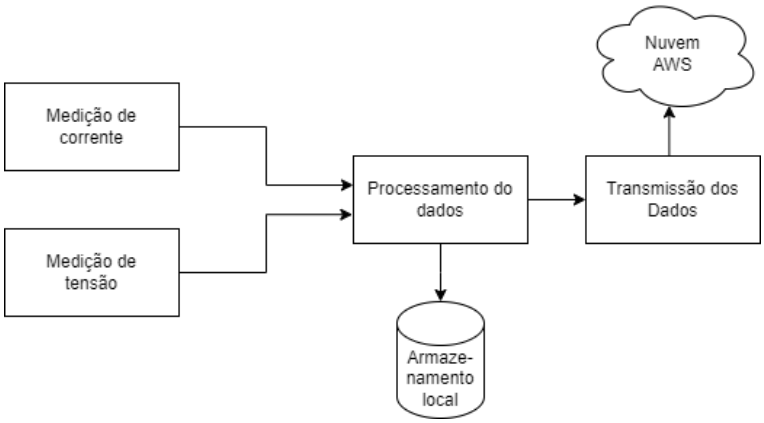
Visando atender os objetivos do trabalho, foram listados os requisitos mínimos que o *smartmeter* deve atender, além dos requisitos adicionais desejáveis:

- Requisitos Mínimos:
    1. Medição de tensão;
    2. Medição de corrente de forma não invasiva;
    3. Cálculo de consumo e geração, ou seja, medição bidirecional;
    4. Medição do fator de potência;
    5. Medição da potência ativa, reativa e aparente;
    6. Transmissão de dados sem fio;
    7. Medição em tempo real com atualização mínima a cada 15 minutos.
- Requisitos Desejados:
    1. Medição de temperatura;
    2. Armazenar as informações localmente;
    3. Emissão de alertas para eventos determinados;
    4. Possuir uma autonomia para caso de queda de energia;
    5. Acionamento de relés de forma remota;
    6. Comunicação com redes industriais cabeadas.

3.2.1.2 Arquitetura do Medidor

A partir dos requisitos, decidiu-se projetar um medidor bidirecional monofásico não invasivo. Dessa forma, serão utilizados um sensor de tensão e dois de corrente, de forma que seja possível medir o consumo e geração a depender do sistema. O medidor é dividido em dois subsistemas: de sensoriamento e de processamento e transmissão. O subsistema de sensoriamento é composto pelos sensores, circuitos de condicionamento de sinais e filtração, e é responsável por amostrar os valores tensão e corrente. O subsistema de processamento e transmissão, é constituído pelo microcontrolador e circuito de comunicação, e é encarregado de realizar a leitura dos sensores, cálculo dos parâmetros elétricos e transmitir as informações para nuvem (Figura 35).

Figura 35 – Arquitetura conceitual do medidor.



Fonte: O Autor (2024).

Portanto, como mostrado na Figura 5 (Seção 2.3), o medidor poderá ser utilizado em diferentes pontos de uma microrrede, como levantamento do consumo de um carregador de carro elétrico ou de uma residência, perfil de geração de um sistema fotovoltaico, o fluxo de potência de um banco de baterias, entre outros.

3.2.1.3 Protocolo de Comunicação

A comunicação entre servidor e dispositivo se deu através mensagens utilizando o protocolo MQTT. Nesse contexto, a comunicação se deu em dois tópicos MQTT:

device_to_server
server_to_device

Dessa forma, o medidor atua como *publisher* no tópico *device_to_server* e como *subscriber* do tópico *server_to_device*. Assim, quando o dispositivo publicar uma mensagem, o agente será responsável por encaminhá-la para o servidor gerenciador de mensagens (Seção 3.3.2). Do mesmo modo, para enviar um comando ao *smartmeter*, a plataforma de supervisão faz uma requisição que dispara um evento que faz uma publicação no tópico em que o dispositivo é assinante, então, o agente direciona a mensagem ao medidor.

3.2.1.4 Estrutura da Mensagem

O *smartmeter* envia dois tipos de mensagens: síncronas e assíncronas. As mensagens síncronas são enviadas a cada 15 minutos com informações gerais sobre as medições do dispositivo, tais como corrente, tensão, fator de potência e consumo. As mensagens assíncronas são enviadas ao acontecer um evento em específico, como por exemplo bateria baixa, fator de potência abaixo de um limite estabelecido, início e fim de geração ou consumo, entre outros. A Tabela 8 descreve alguns eventos que geram uma mensagem assíncrona.

Tabela 8 – Exemplo de eventos que enviam uma mensagem assíncrona.

Identificador	**Evento**	**Descrição**
00	Periódica	Mensagem enviada a cada intervalo de 15 minutos
01	Início de Consumo	O dispositivo detecta o início do fluxo de corrente na linha
02	Fim de Consumo	O dispositivo detecta o fim de um fluxo corrente na linha
03	Alerta Fator de Potência	O fator de potência está abaixo do limite crítico determinado
04	Alerta Queda de Energia	O dispositivo identifica uma tensão nula
05	Alerta de Bateria Baixa	A bateria interna do dispositivo está abaixo de um limite estabelecido

Em contrapartida, a plataforma de supervisão envia apenas mensagens assíncronas. Isso porque elas só são enviadas quando o usuário realizar alguma ação que acione um gatilho de envio de mensagem, como por exemplo alterar calibração e acionar um relé. Abaixo um exemplo de mensagem assíncrona da plataforma.

Tabela 9 – Exemplo de mensagens que podem ser enviadas para o dispositivo.

Identificador	**Evento**	**Descrição**
50	Modifica Estado do Relé	Envia um comando para acionar ou desativar o relé.
51	Calibração Corrente A	Modifica curva de calibração do sensor de corrente A
52	Calibração Corrente B	Modifica curva de calibração do sensor de corrente B
53	Fator de Potência Limite	Define um limite mínimo de fator de potência

O formato das mensagens enviadas pelo dispositivo e pela aplicação transmitem é o JSON. Os dados enviados consistem na combinação do identificador único do medidor, o identificador da mensagem, a data e hora do envio e os parâmetros elétricos, para mensagens enviadas pelo dispositivo (Figura 36), ou comandos a serem executados, para mensagens enviadas pela aplicação (Figura 37).

Figura 36 – Exemplo de mensagem periódica enviada pelo dispositivo.

```
{
    "serial": "XXXXXXXXXX",              // Identificador único
    "datetime" : "23-02-2024 12:00:00", // Data e hora
    "msg" : "00",                        // Identificador da messagem
    "data" : {                           // Parâmetros elétricos
        "vrms": 220,                     // Tensão RMS (V)
        "irmsA": 30,                     // Corrente A RMS (A)
        "irmsB": 0,                      // Corrente B RMS (A)
        "p": 13300,                      // Potência Ativa (W)
        "q": 400                         // Potência Reativa (var)
        "s": 13310,                      // Potência Aparente (VA)
        "fp": 0.96,                      // Fator de Potência
        "kwhA": 5000,                    // Consumo ou Geração (kWh)
        "kwhB": 0,                       // Consumo ou Geração (kWh)
        "f": 60                          // Frequência (Hz)
    }
}
```

Fonte: O Autor (2024).

Figura 37 – Exemplo de mensagem enviada para acionar um relé.

```
{
    "serial": "XXXXXXXXXX",              // Identificador único
    "datetime" : "23-02-2024 12:00:00", // Data e hora
    "msg" : "06",                        // Identificador da messagem
    "data" : {                           // Parâmetros elétricos
        "rele": 1                        // Aciona relé
    }
}
```

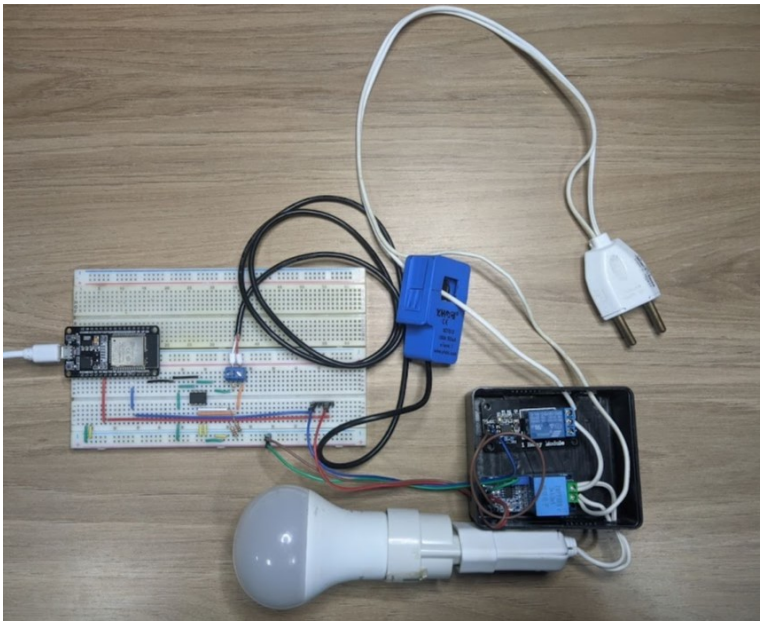
Fonte: O Autor (2024).

3.2.1.5 Prova de Conceito

Para validar o conceito do projeto, foi montado um circuito em *protoboard* integrando o módulo sensor de tensão ZMPT101B, o sensor de corrente SCT-013 em conjunto com o LM741[2], o módulo relé de 10 A e o kit de desenvolvimento ESP32 DevkitC. Então, desenvolveu-se um *firmware* preliminar para realizar medições, acionar o relé e estabelecer a comunicação com o *broker* MQTT do AWS IoT, verificando o envio e recebimento de mensagens. Desse modo, essa abordagem buscou validar a capacidade da arquitetura planejada de atender os requisitos.

[2] O LM741 é um circuito integrado composto por um amplificador operacional. Ele foi utilizado para montar o circuito de condicionamento de sinais do SCT-013 (Seção 2.5.2).

Figura 38 – Montagem em *protoboard* da prova de conceito.



Fonte: O Autor (2024).

3.2.2 Desenvolvimento do Protótipo

Após a prova de conceito, procedeu-se com o projeto de uma placa de circuito impresso para o protótipo. Inicialmente, foi realizado o dimensionamento dos componentes e a montagem dos circuitos de medição. Em seguida, elaborou-se o *design* e a confecção da PCB. Então, ao adquirir a PCB, foi desenvolvido o *firmware* e efetuado testes para calibração e validação do medidor.

3.2.2.1 Arquitetura de Hardware

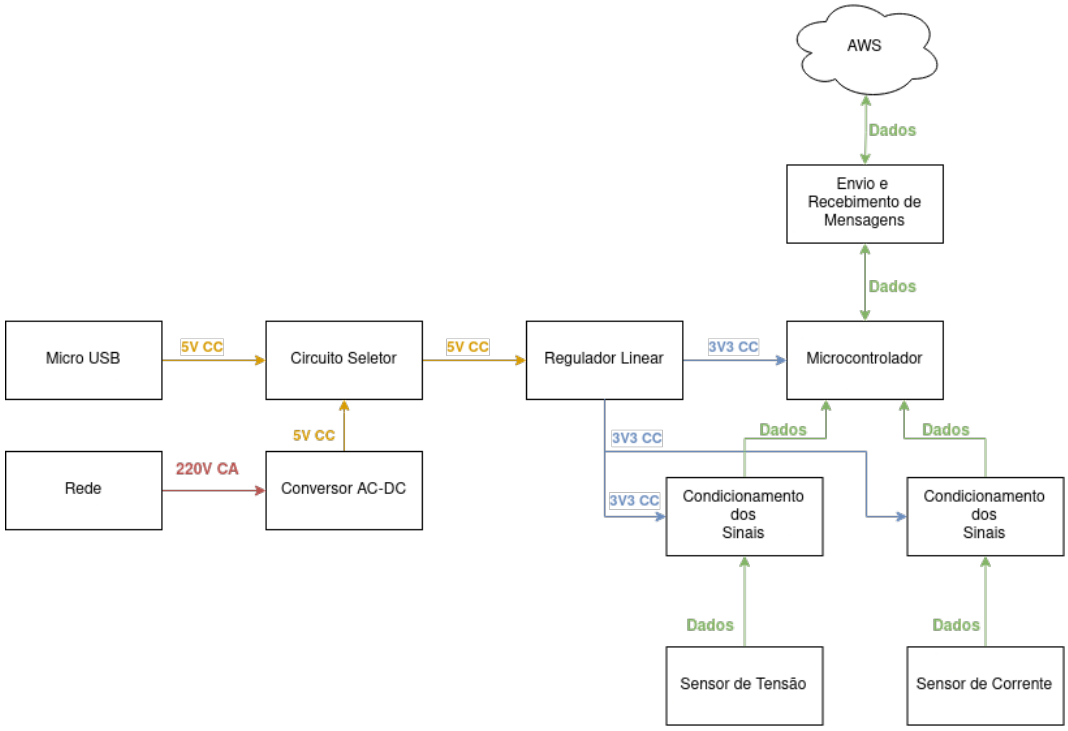
Os circuitos do *smartmeter* podem ser divididos em subsistemas de alimentação, de processamento e de sensoriamento. A arquitetura de *hardware* apresentada na Figura 39 explicita o fluxo de energia e informação entre os circuitos da placa.

3.2.2.2 Subsistema de Alimentação

O subsistema de alimentação é responsável por fornecer a energia necessária para todos os circuitos da placa, isto é, deve ser capaz de manter os níveis de tensão dos componentes na faixa de operação.

Nesse contexto, a PCB pode receber energia pela rede elétrica ou pela interface Micro USB. Quando alimentada pela rede, a tensão AC de entrada é convertida para uma tensão DC de 5 V por meio de um conversor AC-DC. Entretanto, ao ser energizada pela USB, é necessário utilizar uma fonte ou conversor externo que forneça o mesmo nível de tensão DC. Além disso, foi implementado um circuito seletor para priorizar o fornecimento de energia pela rede, isolando a entrada USB, caso o sistema seja alimentado pelas duas interfaces.

Figura 39 – Arquitetura de *hardware* do medidor.



Fonte: O Autor (2024).

3.2.2.3 Subsistema de Processamento e Transmissão

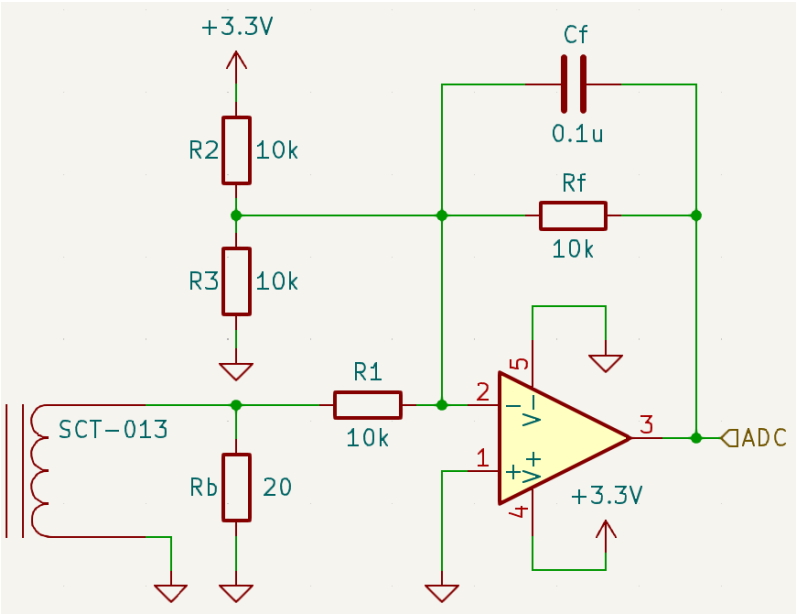
O subsistema de processamento e transmissão é composto pelo microcontrolador e seus periféricos. Assim, ele foi encarregado de realizar todo processamento de dados (Seção 2.7) e rotinas lógicas, tais como conexão com a nuvem AWS, interpretar os comandos recebidos pela plataforma, aplicar filtros digitais, entre outros.

Para isso, foram utilizados diversos periféricos do ESP32, como descrito na Seção 2.8. O ADC foi utilizado para leitura de tensão e corrente a partir dos circuitos de medição. Os GPIOs foram configurados como saídas digitais para acender LEDs indicadores de conectividade e consumo. O módulo RTC foi responsável por informar a data e hora em que os dados foram enviados ou recebidos pelo dispositivo. Além disso, utilizou-se comunicação UART tanto para a gravação do *firmware* quanto para *debug* em um monitor serial. Por fim, a comunicação entre o microcontrolador e a nuvem foi estabelecida através da rede Wifi.

3.2.2.4 Subsistema de Sensoriamento

O subsistema de sensoriamento é composto pelos circuitos de medição de corrente (Seção 2.5) e de tensão (Seção 2.6). Para sensoriamento de corrente, foi utilizado o SCT-013 em conjunto com circuito condicionador de sinais descrito na Seção 2.5.2 (Figura 40). Assim, o resistor de carga foi dimensionado considerando a referência de tensão de 3 V do ADC do microcontrolador ESP32 e a relação de transformação de 1:2000 do TC. Adicionalmente, o capacitor de filtro foi calculado para uma frequência de corte de 800 Hz.

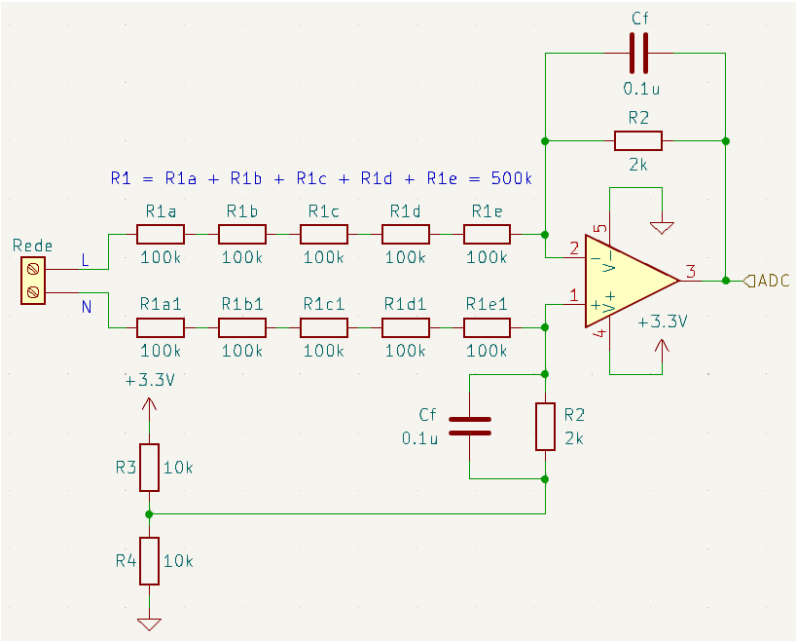
Figura 40 – Esquemático do circuito de medição de corrente.



Fonte: O Autor (2024).

Ademais, foi projetado um circuito de medição diferencial para amostrar a tensão do sistema (Figura 41). Dessa forma, como detalhado na Seção 2.6.2, foram calculados os resistores R_1 e R_2 de forma que o potencial da rede tenha uma atenuação de 0,004. Como a medição diferencial é não isolada, adicionou-se uma malha resistiva com equivalentes ao valor de R_1 a fim de proporcionar maior segurança. Além disso, também foi adicionado um capacitor de filtro para frequência de corte de 800 Hz.

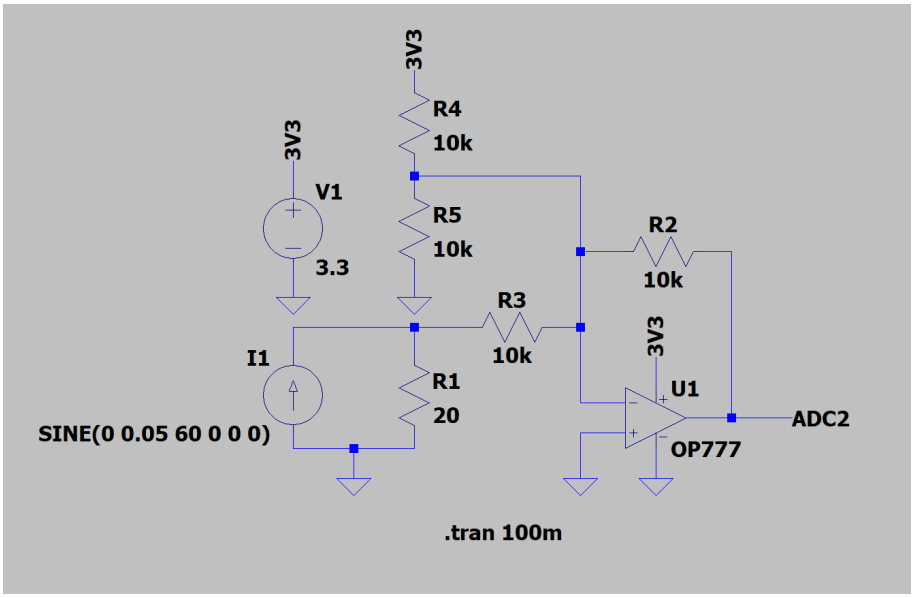
Figura 41 – Esquemático de medição de tensão.



Fonte: O Autor (2024).

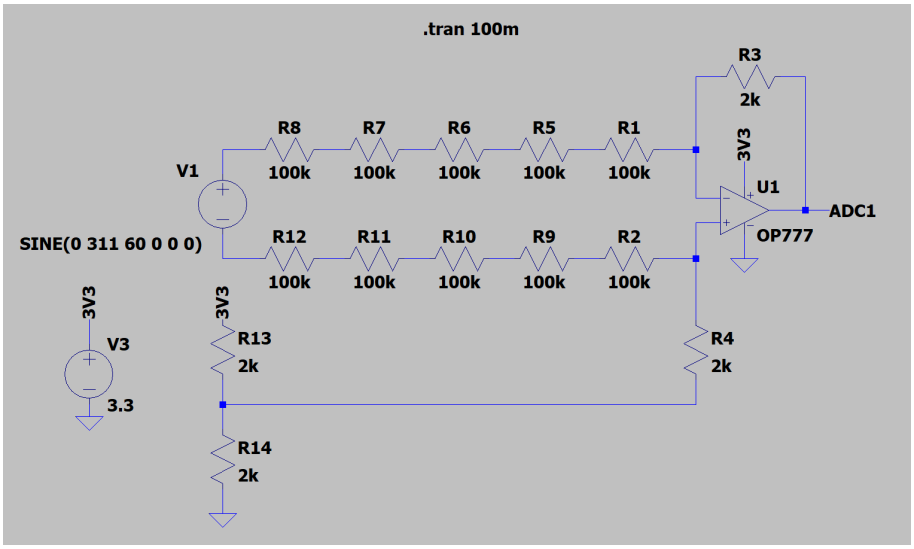
Após o dimensionamento dos componentes e definição dos circuitos, foi realizada uma simulação através do *software* LTSpice a fim de verificar se os circuitos estavam condicionando corretamente os sinais. O TC foi modelado como uma fonte de corrente ideal de frequência de 60 Hz e amplitude de 50 mA (Figura 42). A rede foi modelada como uma fonte de tensão ideal de frequência de 60 Hz e amplitude de 311 V (Figura 43). Assim, foi efetuada uma análise de transiente durante 100 milissegundos.

Figura 42 – Circuito de condicionamento do sinal de corrente simulado no LTSpice.



Fonte: O Autor (2024).

Figura 43 – Circuito de condicionamento do sinal de tensão simulado no LTSpice.



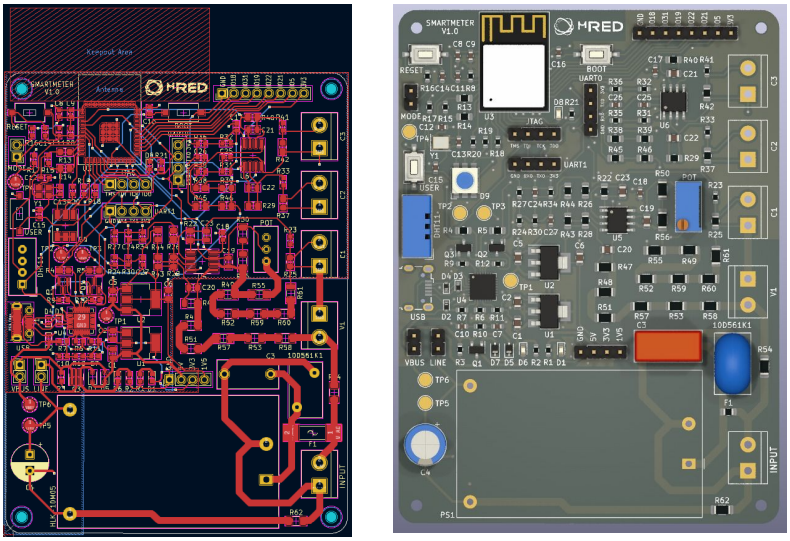
Fonte: O Autor (2024).

3.2.2.5 *Hardware Design*

Definidos os circuitos e seus componentes, projetou-se a PCB utilizando o *software* KiCad. Inicialmente, foi elaborado o esquemático de cada subsistema, adicionando *features* com o objetivo de facilitar os testes e de permitir a evolução do projeto a posteriori, como o posicionamento de *test points*, adição de *headers* para integração de outros módulos no futuro, entre outros. Em seguida, desenvolveu-se a PCB do medidor (Figura 44).

Figura 44 – *Design* da PCB.

(a) Projeto no KiCad. (b) Visualização em 3D.

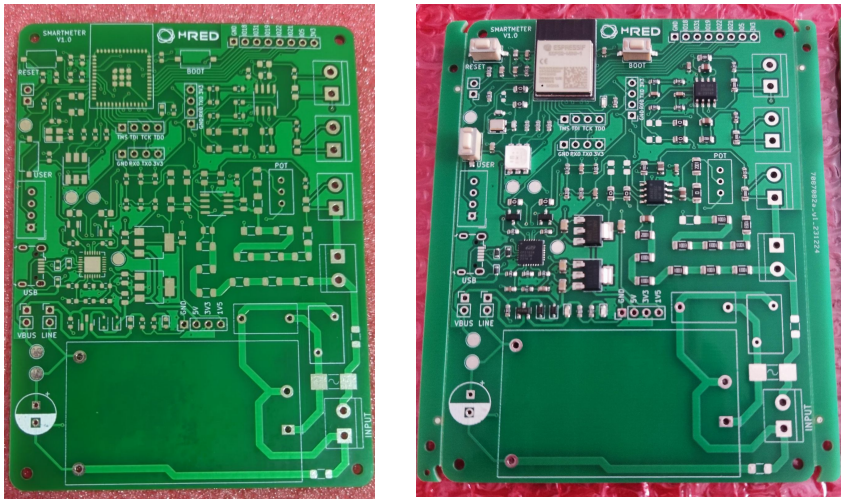


Fonte: O Autor (2024).

Dessa forma, a fabricação da placa foi realizada, e então, montaram-se os componentes essenciais para conduzir os testes iniciais (Figura 45).

Figura 45 – PCBs fabricadas.

(a) PCB sem componentes montados. (b) PCB montada com os principais componentes.



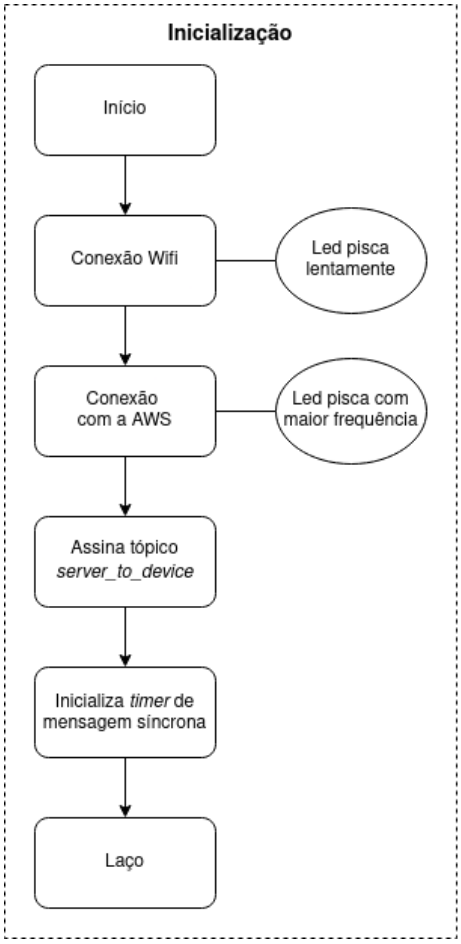
Fonte: O Autor (2024).

3.2.2.6 *Firmware*

A programação do *firmware* foi realizada utilizando o Arduino IDE, já que, ela fornece toda ferramenta necessária para compilação e gravação do código no microcontrolador. Desse modo, como explicado em (SYSTEMS, 2024b), foi instalado o pacote de suporte do ESP32 na IDE. A linguagem de programação utilizada no Arduino é a linguagem C++ com pequenas modificações. Além disso, o programa foi implementado utilizando o FreeRTOS, que é um sistema operacional em tempo real para dispositivos embarcados, a fim de gerenciar e executar as rotinas paralelamente.

O *firmware* do dispositivo pode ser dividido em duas partes: a inicialização e o laço infinito. Ao iniciar, o medidor se conecta na rede wifi configurada previamente. Em seguida, ele estabelece uma conexão com o *broker* do AWS IoT e se inscreve no tópico para receber comandos. Por fim, ele inicializa um *timer* responsável por marcar os intervalos de mensagem síncrona e, então, entra no *loop* infinito. Ademais, durante o processo de conexão, o led indica o estado da comunicação. Assim, durante a conexão com o wifi ele pisca lentamente, por sua vez, no processo de conectividade com a nuvem AWS, ele pisca com uma maior frequência. A Figura 46 esquematiza o fluxo lógico de inicialização do medidor inteligente.

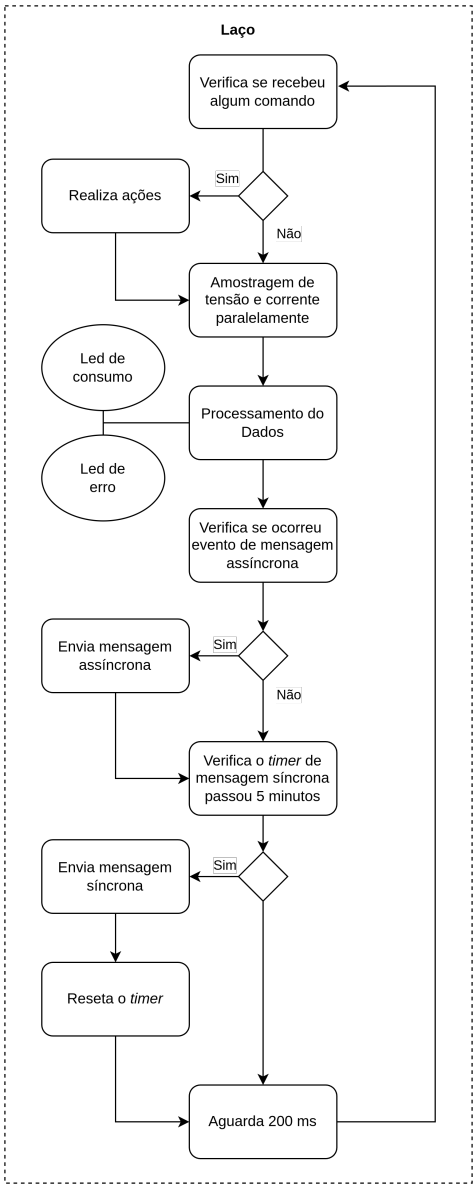
Figura 46 – Fluxograma de inicialização do *firmware*.



Fonte: O Autor (2024).

Ao entrar no laço infinito, o dispositivo verifica se o agente MQTT o encaminhou alguma mensagem. Caso sim, ele realiza as ações decorrentes do comando recebido. Posteriormente, são adquiridas, concomitantemente, 240 amostras de tensão e corrente a uma frequência de 1kHz. Essas amostras passam por um filtro de média móvel para n igual a 10 amostras. Então, calculam-se os parâmetros elétricos, como detalhado na Seção 2.7. Ao interpretar os dados encontrados, o *smartmeter* avalia se ocorreu algum evento de mensagem assíncrona. Se sim, uma mensagem assíncrona é enviada. Assim, é analisado se o valor do *timer* de mensagem síncrona atingiu o intervalo de tempo definido. Caso positivo, uma mensagem síncrona é enviada e o *timer* é resetado. Finalmente, após aguardar 200 milissegundos, o laço se inicia novamente (Figura 47).

Figura 47 – Fluxograma do laço infinito do *firmware*.

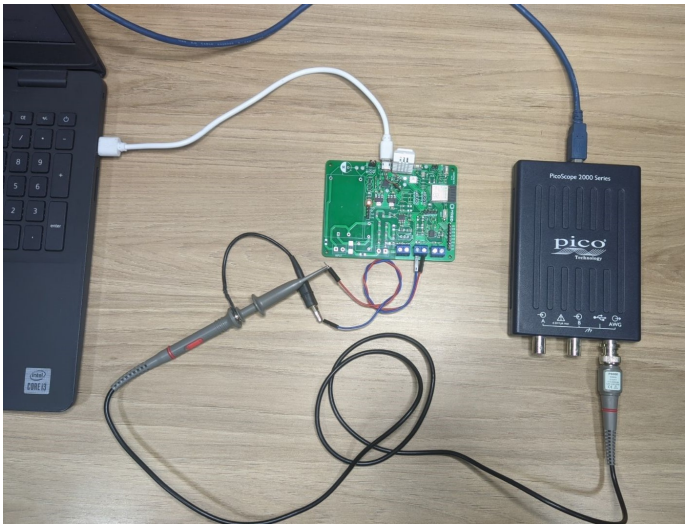


Fonte: O Autor (2024).

3.2.2.7 Validação do Protótipo

Inicialmente, para verificar se o circuito de condicionamento de sinais da placa estava operando como projetado, foi utilizado um gerador de funções para injetar uma corrente senoidal simulando o secundário de um transformador de corrente e, então, foram efetuadas medições com um osciloscópio para avaliar as formas de onda na saída do circuito (Figura 48).

Figura 48 – Banca de testes com gerador de funções.



Fonte: O Autor (2024).

Em seguida, o Laboratório de Mobilidade (LAM) da Universidade Federal de Pernambuco foi adotado como ambiente de calibração, testes e validação do medidor (Figuras 49). Como descrito em (MARTINS, 2022), o LAM é composto por uma microrrede híbrida que permite emular vários cenários e aplicações, tais como sistemas fotovoltaicos, gerador a diesel, sistemas de armazenamento e carregamento veicular, podendo operar conectada à rede ou de modo ilhado (Figura 50). A Figura 51 apresenta um esquemático da microrrede do LAM.

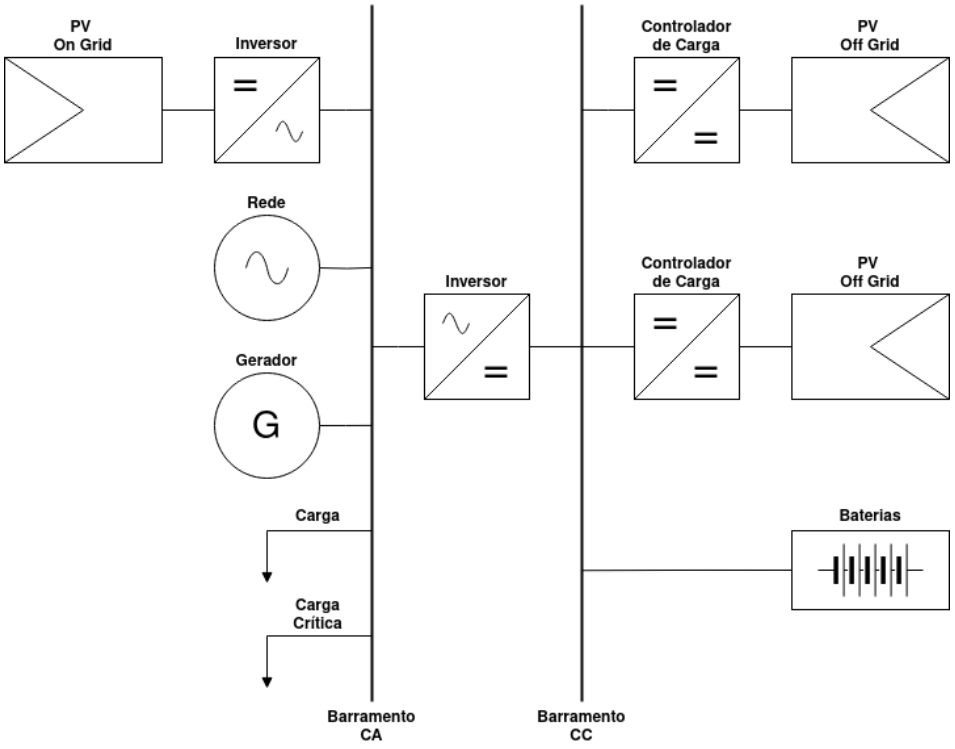
Figura 49 – Laboratório de Mobilidade - UFPE.

Fonte: O Autor (2024).

Figura 50 – Equipamentos da microrrede do LAM.

Fonte: O Autor (2024).

Figura 51 – Esquemático da microrrede do LAM.



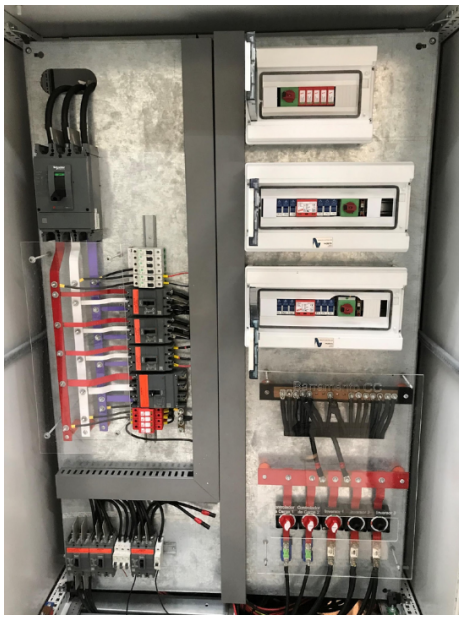
Fonte: O Autor (2024).

Dessa forma, quando conectada à rede, a microrrede do LAM pode absorver potência a fim de suprir a demanda ou injetar o excedente produzido para a rede. Nessa perspectiva, o fluxo bidirecional na microrrede flui através do barramento CA, onde estão conectados os inversores, carregadores e outros componentes. Por sua vez, no modo ilhado, a operação é realizada de forma segmentada ou isolada da rede, portanto, a carga é suprida pela geração fotovoltaica, pelo sistema de armazenamento ou pelo gerador. Além disso, a geração fotovoltaica pode ser utilizada para carregar as baterias (MARTINS, 2022).

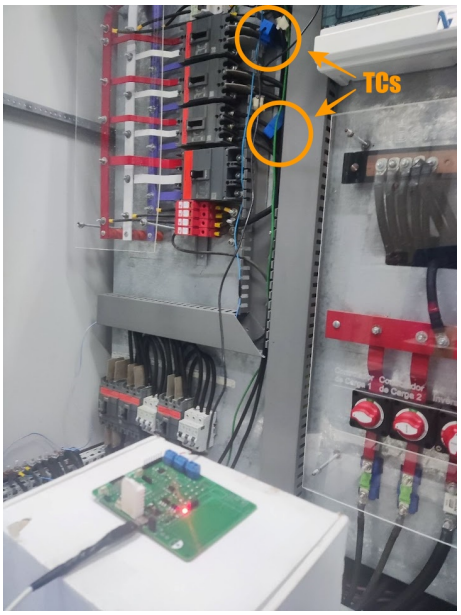
Nesse contexto, realizou-se a calibração da medição de corrente do protótipo. Para isso, foram utilizadas as baterias da microrrede para inserir correntes e um equipamento de medição de referência. Desse modo, como mostrado na Figura 52, a calibração se deu conectando o medidor de referência e o protótipo no barramento CA e, então, injetou-se gradualmente correntes no circuito. Assim, associando o valor medido pelo ADC do *smartmeter* com o valor do dispositivo de referência, obteve-se a curva de calibração.

Figura 52 – Ambiente de calibração.

(a) Quadros de Energia.



(b) Testes de medição no barramento CA.



Fonte: O Autor (2024).

Por fim, o medidor inteligente foi conectado de forma a monitorar a geração do sistema fotovoltaico e o consumo da carga. Assim, obteve-se as informações de geração e consumo nesses cenários, que foram comparadas com os dados registrados pelo inversor.

3.3 INFRAESTRUTURA DE GERENCIAMENTO

A infraestrutura de gerenciamento é o sistema responsável por receber, processar e armazenar as mensagens enviadas pelos medidores. Ou seja, ela é a conexão entre o *smartmeter* (camada 1) e a plataforma de visualização de dados (camada 3). Assim, ela é composta pelos serviços da plataforma de nuvem da AWS e pelo servidor de gerenciamento de mensagens.

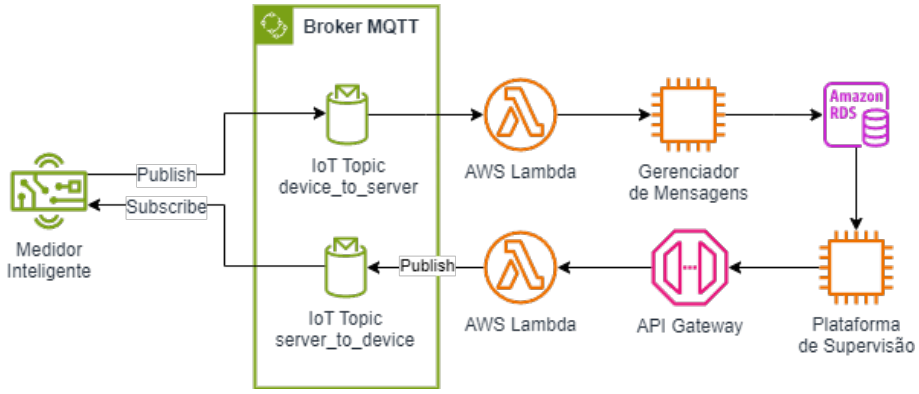
3.3.1 Arquitetura de Gerenciamento

Conforme exibido na Figura 53, a arquitetura da infraestrutura de gerenciamento pode ser dividida em dois fluxos: o envio de mensagens do dispositivo para o servidor e envio da mensagem da plataforma para o dispositivo.

No primeiro, o dispositivo envia a mensagem para a AWS por meio do protocolo MQTT e seus serviços a encaminham para o gerenciador de mensagens (Seção 3.3.2), que é o servidor encarregado de decodificar, processar e armazenar as mensagens no banco de dados. Posteriormente, a plataforma de monitoramento consulta essa mesma base de dados para exibir as informações.

Já o segundo, a aplicação Web realiza uma chamada HTTP com o conteúdo da mensagem para os serviços da AWS que são responsáveis por publicar a mensagem no tópico em que o dispositivo está inscrito. Os serviços utilizados para viabilizar esses fluxos são detalhados nas próximas seções.

Figura 53 – Arquitetura da Infraestrutura de Gerenciamento de Mensagens.

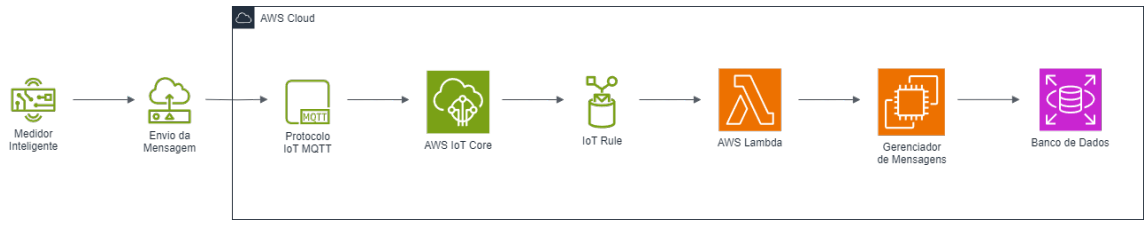


Fonte: O Autor (2024).

3.3.1.1 Envio de Mensagem do Dispositivo

Inicialmente, como apresentado na Seção 3.2.1.4, para realizar o envio de informações ao AWS IoT Core via protocolo MQTT, o dispositivo se conecta ao *broker* e envia uma mensagem para o tópico *device_to_server*. No AWS IoT Core foi definida uma IoT Rule que direciona o conteúdo da mensagem recebida ao serviço AWS Lambda. No AWS Lambda, é executada uma função que realiza uma requisição HTTP de método POST para um *endpoint* da API do servidor gerenciador de mensagens, enviando a mensagem do medidor para o servidor. Por fim, ele processa e armazena a mensagem no banco de dados (Figura 54).

Figura 54 – Fluxo de informação do dispositivo para o servidor.

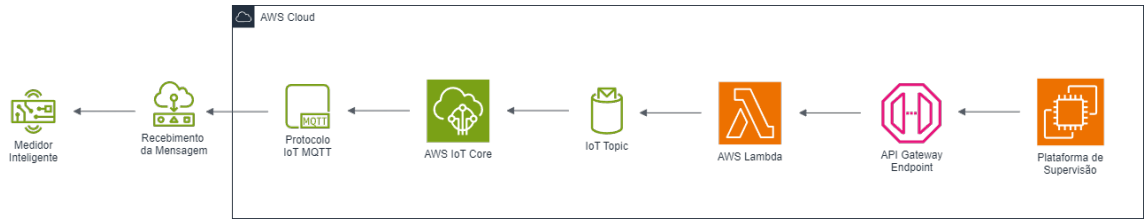


Fonte: O Autor (2024).

3.3.1.2 Envio de Mensagem da Plataforma

A aplicação Web envia as informações através de uma chamada HTTP para uma rota de uma API RESTful criada no API Gateway. Ao receber a requisição, a API aciona uma função lambda que obtém as informações da chamada e publica uma mensagem no tópico *server_to_device*. Desse modo, o agente MQTT encaminha a mensagem para todos os dispositivos assinantes, como evidenciado na Figura 55. Observa-se que o gerenciador de mensagens atua exclusivamente na comunicação do dispositivo para a plataforma.

Figura 55 – Fluxo de informação do servidor para o dispositivo.



Fonte: O Autor (2024).

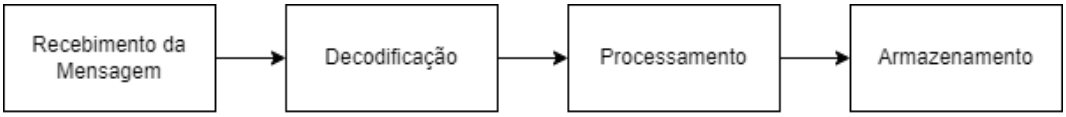
Como definido na Seção 3.2.1.4, em toda mensagem enviada há um serial identificador único do medidor. Dessa forma, a plataforma pode especificar para qual *smartmeter* a mensagem é destinada. Em outras palavras, todos os dispositivos assinantes recebem a mensagem, no entanto, apenas o medidor com o serial especificado a interpreta. Não obstante, também é possível fazer uma transmissão *broadcasting*, isto é, em que todos os dispositivos recebam e interpretem a mensagem.

3.3.2 Gerenciador de Mensagens

O gerenciador de mensagens é uma API que é executada em uma instância EC2. Ele possui apenas uma rota definida que é responsável por receber as requisições da função lambda que envia o conteúdo da mensagem de um dispositivo (Seção 3.3.1.1).

Em primeira instância, ao receber a mensagem, o servidor realiza a decodificação, ou seja, extrai os parâmetros fornecidos a partir do tipo da mensagem. Em seguida, é realizado o processamento, que inclui rotinas como a identificação do dispositivo que enviou a mensagem e a regularidade do seu cadastrado na plataforma, verificação dos eventos e ações associadas à mensagem, validação da integridade dos dados, entre outras. Por último, o sistema de gerenciamento armazena as informações recebidas no banco de dados (Figura 56).

Figura 56 – Diagrama de blocos de funcionamento do gerenciador de mensagens.



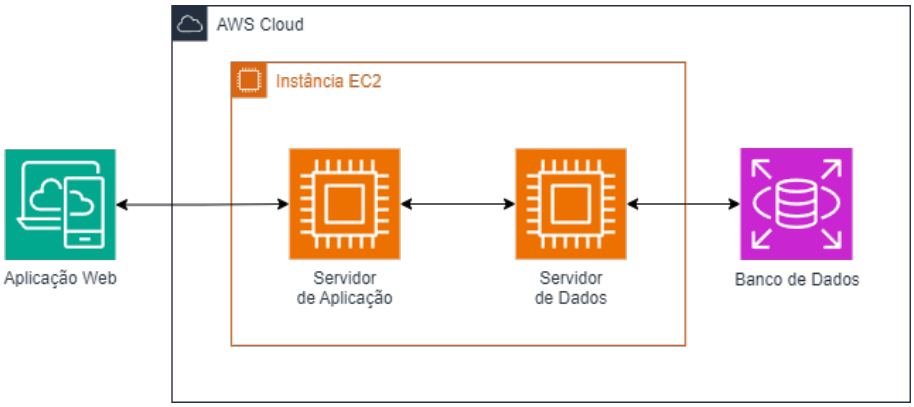
Fonte: O Autor (2024).

3.4 PLATAFORMA DE SUPERVISÃO

A plataforma de supervisão é dividida em três componentes: banco de dados, servidor de dados e servidor de aplicação. Para esse sistema utiliza-se duas instâncias na Amazon EC2 e uma na Amazon RDB, que são utilizadas como um ambiente de execução dos servidores e do banco de dados, respectivamente. Além disso, a aplicação Web foi desenvolvida para ser exibida no computador do cliente ou usuário por meio de um navegador Web (Seção 2.10.2).

O diagrama de funcionamento da plataforma de supervisão pode ser visualizado na Figura 57. Inicialmente, o cliente envia chamadas HTTP para o servidor de aplicação, que funciona como um servidor Web. Esse servidor, por sua vez, realiza requisições HTTP ao servidor de dados, que atua como uma API RESTful. Essa API efetua consultas e alterações no banco de dados, retornando as informações solicitadas ao servidor de aplicação. Então, o servidor de aplicação fornece ao cliente os elementos necessários para o navegador exibir a aplicação Web aos usuários.

Figura 57 – Arquitetura da Plataforma Web de Supervisão.



Fonte: O Autor (2024).

3.4.1 Modelagem do Banco de Dados

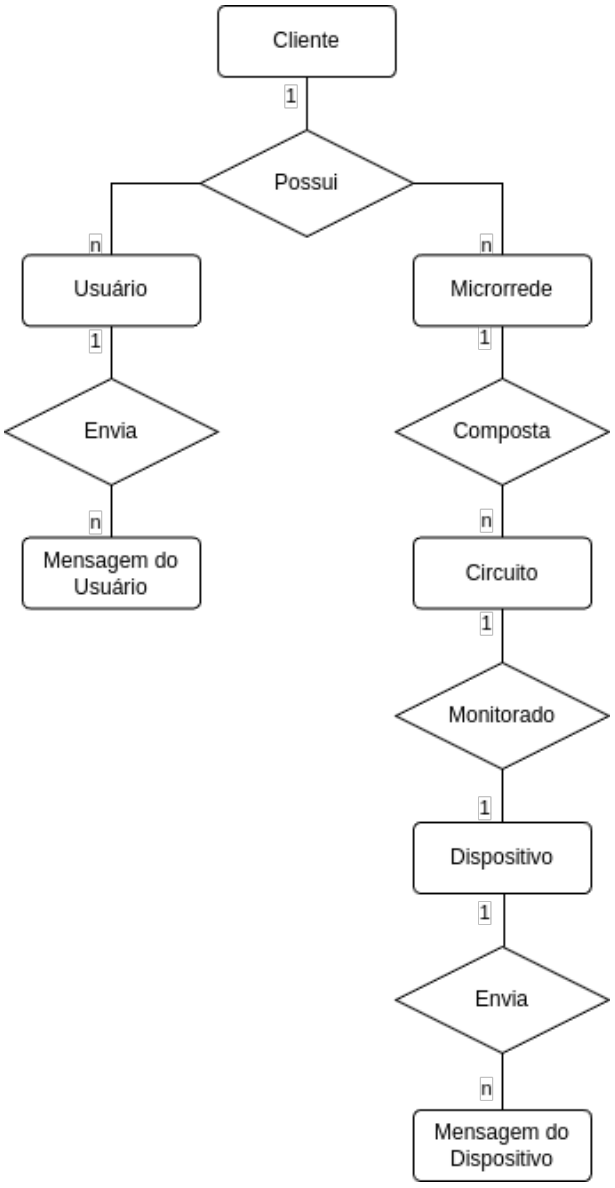
Segundo (HEUSER, 2009), o projeto de um banco de dados é realizado em duas fases: modelagem conceitual e projeto lógico. Na primeira fase, é construído um modelo conceitual, na forma de um diagrama entidade-relacionamento. Em segunda instância, o projeto lógico objetiva transformar o modelo conceitual em um modelo lógico (Seção 3.4.1). Neste trabalho, foi modelado um banco de dados relacional utilizando mySQL.

3.4.1.1 Modelagem Conceitual

O banco de dados deve conter uma entidade fundamental chamada cliente, de onde se deriva todas as outras entidades. Em outras palavras, cada entidade na plataforma está associada a um cliente específico. Os clientes têm a capacidade de registrar diferentes microrredes, e cada microrrede pode conter diversos circuitos a serem monitorados, como circuitos de cargas, estações de carregamento de veículos elétricos, sistemas fotovoltaicos, entre outros. Por fim, é vinculado ao circuito um dispositivo que envia mensagens com informações sobre ele.

Em segundo plano, para acessar as informações na plataforma, cada cliente pode ter diversos usuários. Além disso, cada usuário pode enviar diferentes comandos para os medidores. Assim, a Figura 58 exibe o diagrama entidade-relacionamento desenvolvido para o sistema.

Figura 58 – Diagrama Entidade Relacionamento básico do sistema.



Fonte: O Autor (2024).

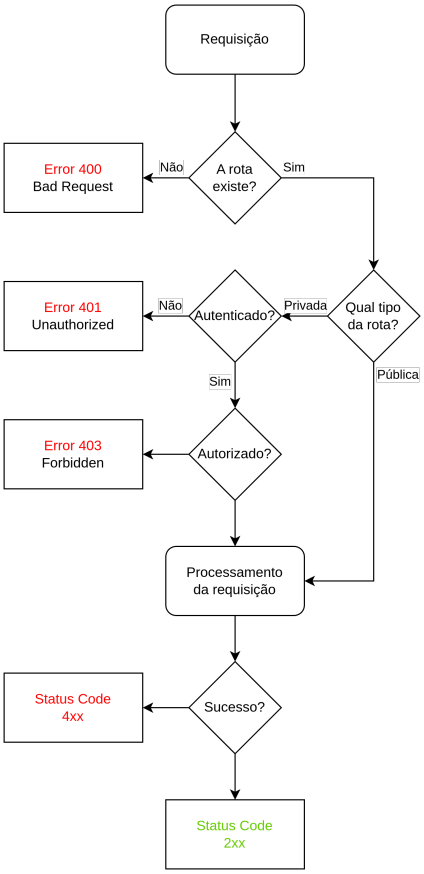
3.4.1.2 Projeto Lógico

Em consonância com (HEUSER, 2009), foi utilizado o MySQL para construir um banco de dados relacional que possua boa *performance* e que simplifique o desenvolvimento e a manutenção da aplicação. Dessa forma, buscou-se criar tabelas que possuam uma única coluna como chave primária, além de evitar atributos opcionais e ter os dados necessários a uma consulta em uma única linha. Assim, o projeto lógico se deu traduzindo as entidades, os relacionamentos e seus respectivos atributos em tabelas, especificando o tipo e a obrigatoriedade de cada coluna. O diagrama entidade-relacionamento estendido do banco de dados construído para o sistema pode ser visualizado no Apêndice A.

3.4.2 Servidor de Dados

O servidor de dados é uma API RESTful responsável por consultar e fazer modificações no banco de dados. Dessa forma, foram criadas rotas para interação entre os usuários e o sistema, permitindo que eles realizem ações como criar, editar ou informações existentes. Outrossim, o servidor é responsável por gerenciar as autenticações, isto é, garantir que a informação só possa ser acessada por usuários devidamente autorizados. A Figura 59 apresenta o fluxograma lógico de funcionamento da API.

Figura 59 – Diagrama lógico do servidor de dados.



Fonte: O Autor (2024).

As rotas do servidor foram divididas em dois tipos: rotas públicas e rotas privadas. As rotas públicas podem ser acessadas por qualquer usuário que possua a URL. Entretanto, as privadas são acessadas somente via autenticação. As rotas de *Sign In* e de recuperação de senha são públicas, consequentemente, as restantes são privadas. A Tabela 10 apresenta os *endpoints* bases para o roteamento.

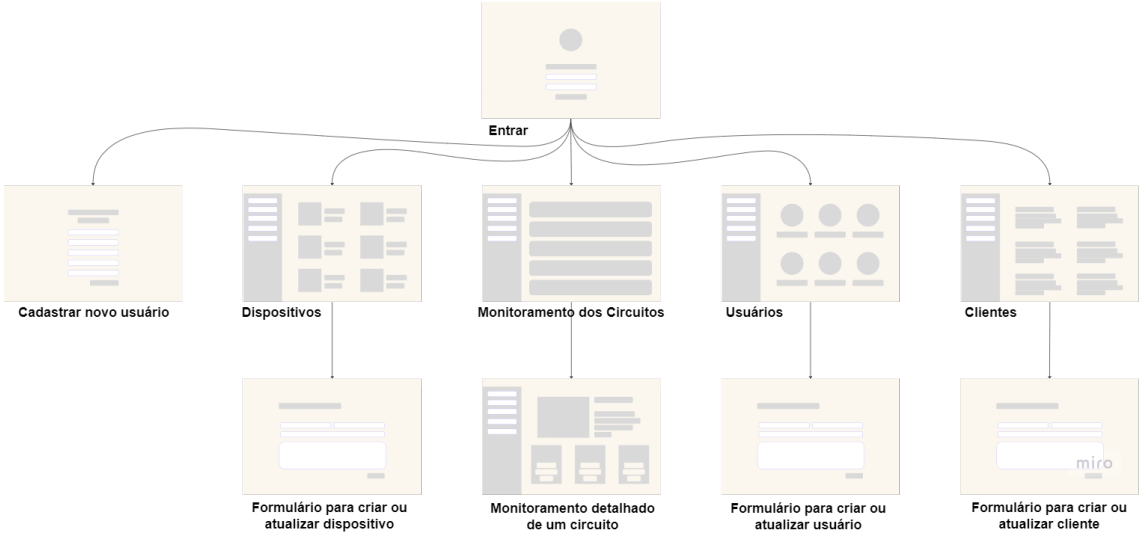
Tabela 10 – *Endpoints* das rotas implementadas.

Endpoint	**Tipo**	**Funções**
api/client/	Privada	CRUD clientes
api/account/	Privada	CRUD usuários
api/auth/	Pública	Autenticação
api/microrred/	Privada	CRUD microrredes
api/circuit/	Privada	CRUD circuitos
api/device/	Privada	CRUD dispositivos

3.4.3 Servidor de Aplicação

O servidor de aplicação oferece uma interface gráfica que permite aos usuários gerenciar e monitorar as informações obtidas pelo *smartmeter*. Desse modo, foi implementada uma tela de acesso para realizar a autenticação e permitir que os usuários vinculados aos clientes acessem a plataforma. Além disso, foram desenvolvidas interfaces para gerenciar dispositivos, clientes e usuários, oferecendo funcionalidades como criação, edição, remoção e listagem desses elementos. Também foi criado um *dashboard* de monitoramento que proporciona uma visão geral da microrrede, além de permitir a criação, edição e remoção de circuitos. Por fim, foi elaborada uma tela para visualização detalhada dos dados obtidos pelo medidor.

Figura 60 – Diagrama das telas desenvolvidas para a aplicação web.



Fonte: O Autor (2024).

4 RESULTADOS

4.1 PROVA DE CONCEITO

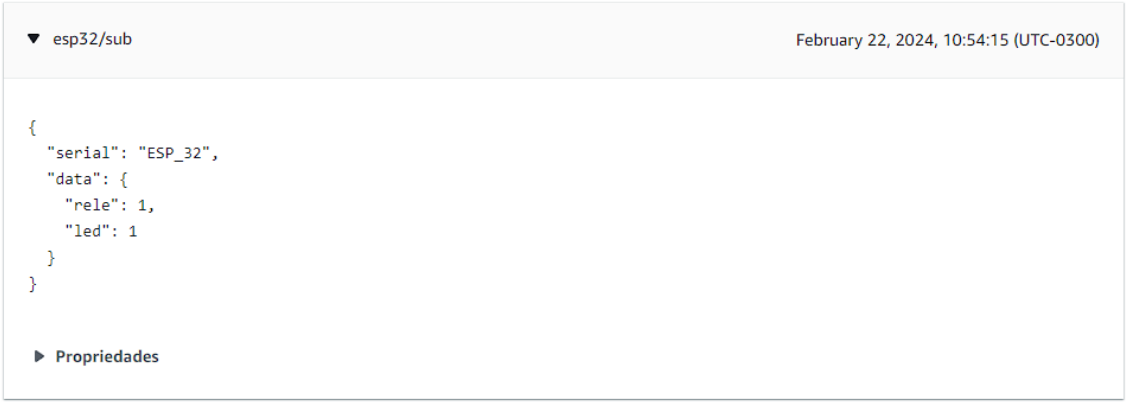
O circuito elaborado na Seção 3.2.1.5 foi utilizado para confirmar que o sistema embarcado idealizado era capaz de fazer medições, bem como transmitir e receber dados da nuvem via protocolo MQTT. Assim, foi desenvolvido um *firmware* que permitiu o microcontrolador enviar amostras de tensão e corrente para AWS. Além disso, o circuito contém um relé a fim de acender e apagar uma lâmpada remotamente.

Para isso, foi utilizado o MQTT *Test Client*, que é uma ferramenta do AWS IoT para monitorar as mensagens MQTT recebidas pelo agente MQTT, além de permitir a assinatura e a publicação de mensagens em tópicos. Assim, foi possível visualizar as mensagens enviadas pelo medidor e controlar o estado do relé remotamente. A Figura 61 exibe uma mensagem enviada pelo dispositivo, enquanto as Figuras 62 e 63 apresentam a mensagem publicada pelo MQTT *Test Client* e recebida pelo microcontrolador.

Figura 61 – Mensagem enviada pelo ESP32 ao AWS IoT.

Fonte: O Autor (2024).

Figura 62 – Mensagem enviada por meio do AWS IoT.



Fonte: O Autor (2024).

Figura 63 – Registro em um monitor serial dos dados recebidos pelo ESP32.

```
rst:0x1 (POWERON_RESET),boot:0x13 (SPI_FAST_FLASH_BOOT)
configsip: 188777542, SPIWP:0xee
clk_drv:0x00,q_drv:0x00,d_drv:0x00,cs0_drv:0x00,hd_drv:0x00,wp_drv:0x00
mode:DIO, clock div:1
load:0x3fff0030,len:1344
load:0x40078000,len:13964
load:0x40080400,len:3600
entry 0x400805f0
Connecting to Wi-Fi.........................Wifi Connected!
Connecting to AWS IOT.........AWS IoT Connected!
Receive data!
topic: server_to_device
data:
{rele:1,led:1}
```

Fonte: O Autor (2024).

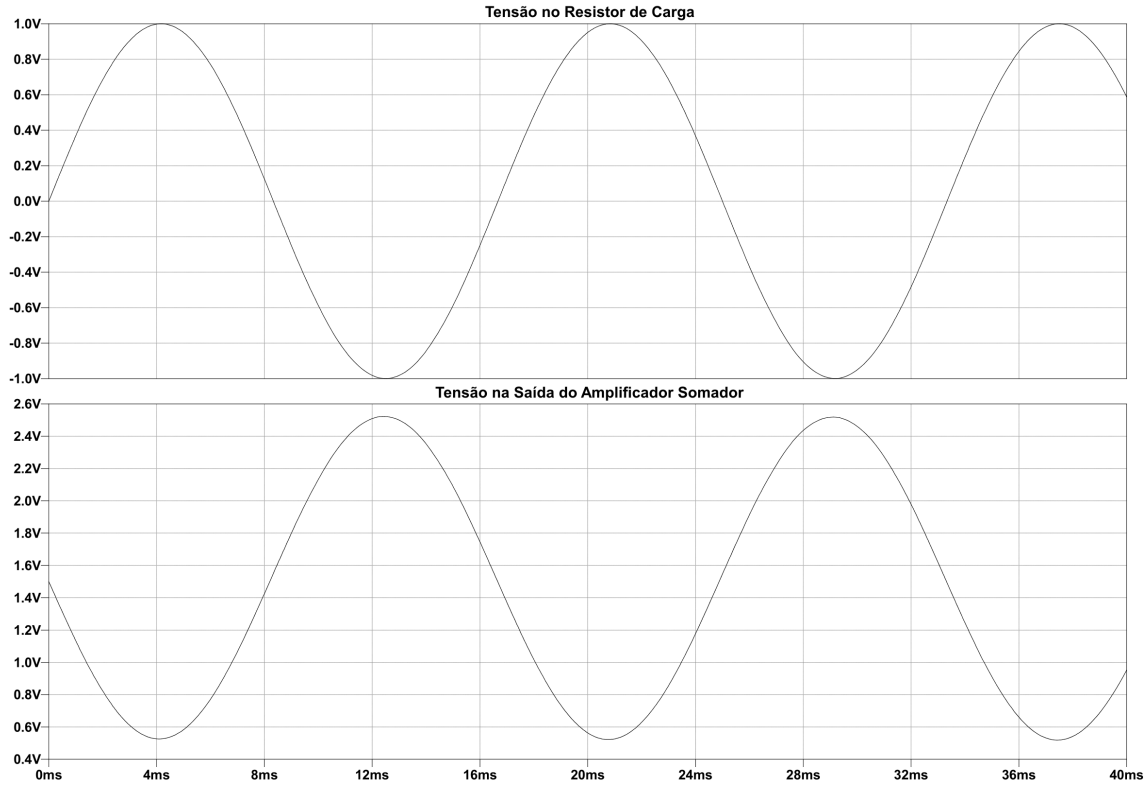
4.2 PROTÓTIPO DO MEDIDOR

Antes da fabricação da PCB, os circuitos de medição foram simulados para garantir que o dimensionamento foi feito adequadamente. Em seguida, após a fabricação, efetuaram-se testes para validar o funcionamento dos subsistemas do medidor.

4.2.1 Simulações do Circuito de Condicionamento de Sinais

Como apresentado na Seção 3.2.2.4, foram realizadas simulações no LTSpice a fim de verificar qualitativamente o comportamento dos circuitos de medição. Primeiramente, o sensoriamento de corrente foi simulado utilizando uma fonte de corrente senoidal de 50 mA de amplitude e 60 Hz de frequência para representar o sinal de saída do TC. Então, as tensões encontradas no resistor de carga e na saída do amplificador somador são exibidas na Figura 64.

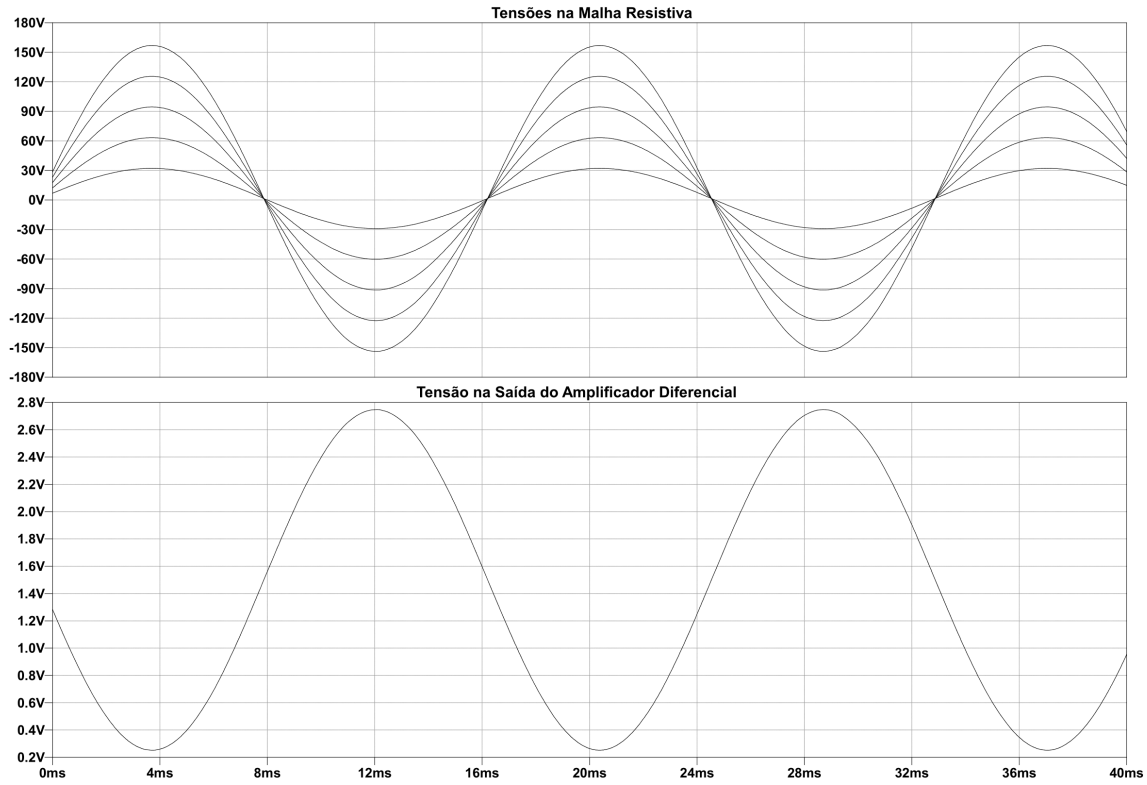
Figura 64 – Tensão no resistor de carga e na saída do amplificador somador.



Fonte: O Autor (2024).

Posteriormente, foi simulado o circuito de medição de potencial e, então, observou-se as tensões na malha resistiva e na saída do amplificador diferencial (Figura 65).

Figura 65 – Queda de tensão na malha resistiva e sinal na saída do amplificador diferencial.



Fonte: O Autor (2024).

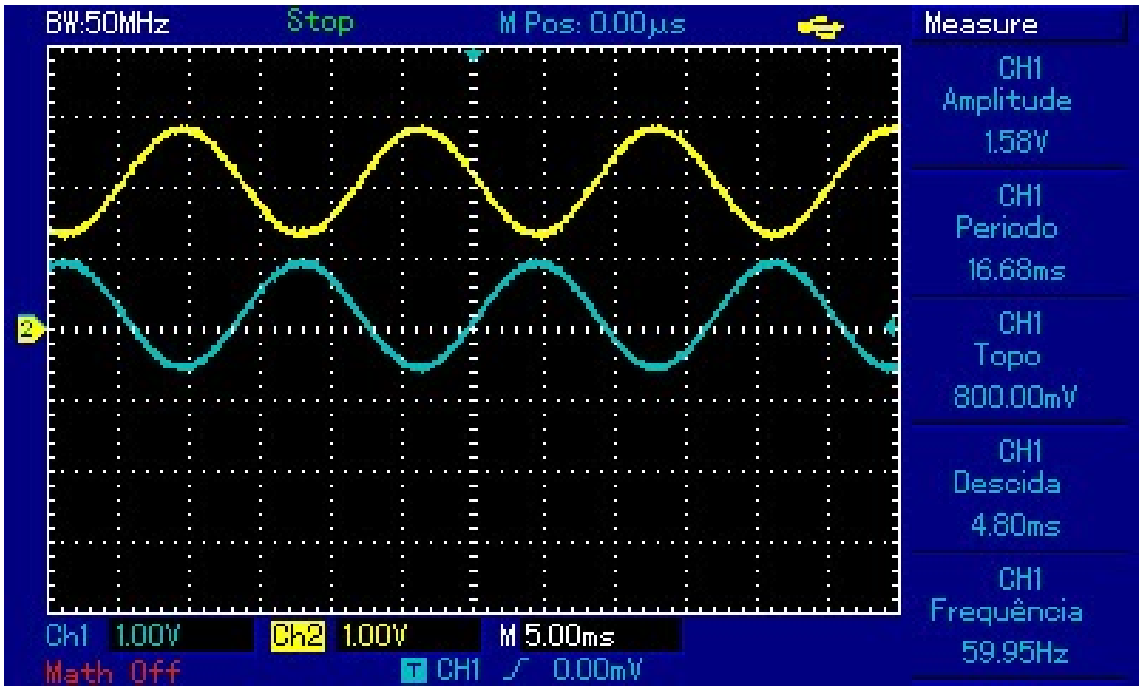
4.2.2 Testes da PCB

Inicialmente, foram efetuados os testes no subsistema de alimentação. Para isso, a placa foi energizada através da interface USB e foi realizada a medição de tensão em pontos críticos como saída do regulador, alimentação do microcontrolador e dos amplificadores operacionais, divisor de tensão, entre outros.

4.2.2.1 Testes de Medição de Corrente

Para validar o condicionamento de sinais, foi utilizado um gerador de funções para emular o sinal de saída do transformador de corrente, como descrito na Seção 3.2.2.7. Dessa forma, foi injetada uma corrente no conector de entrada do TC e, então, mediu-se com um osciloscópio a tensão no resistor de carga e na saída do circuito de condicionamento. A Figura 66 constata que o circuito se comportou conforme projetado.

Figura 66 – Em azul, a tensão gerada no resistor de carga e, em amarelo, a tensão na saída do amplificador somador do circuito de condicionamento.



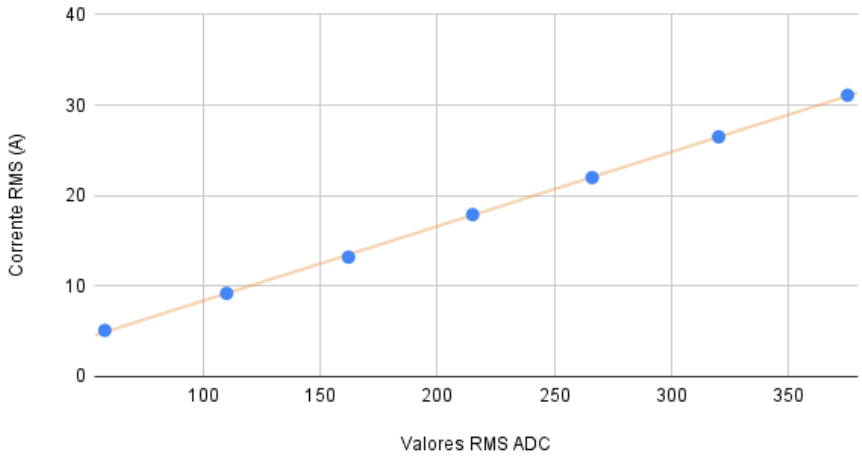
Fonte: O Autor (2024).

Em seguida, prosseguiu-se com a calibração. Para isso, foi programado um firmware que adquire e calcula o valor RMS do sinal de entrada do ADC, ou seja, o sinal do TC condicionado. Além disso, conectou-se o TC e o alicate amperímetro Fluke 302+ no barramento CA da microrrede do LAM. Assim, foram injetados fluxos de corrente variados no barramento através de um banco de baterias, mostrados na Tabela 11. Dessa forma, criou-se a curva de calibração realizando uma aproximação linear dos valores RMS encontrados pelo ADC associados aos valores de corrente RMS exibidos no equipamento de referência (Figura 67).

Tabela 11 – Valores RMS calculados pelo ADC e medidos pelo amperímetro.

Valor RMS ADC	**Corrente RMS Fluke**
58	5,1 A
110	9,2 A
162	13,2 A
215	17,9 A
266	22 A
320	26,5 A
375	31,1 A

Figura 67 – Curva de calibração.

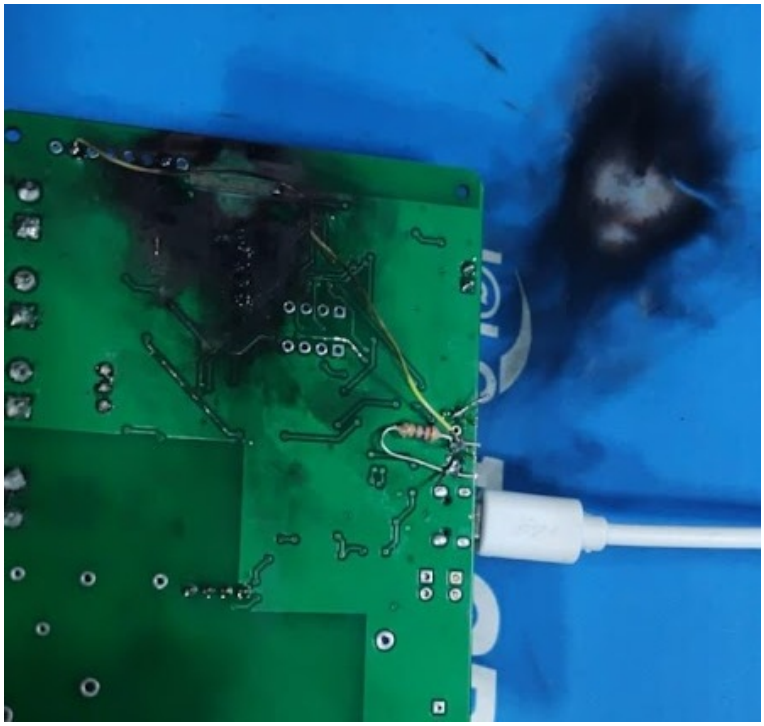


Fonte: O Autor (2024).

4.2.2.2 Testes de Medição de Tensão

Para realizar os testes de tensão, conectou-se a placa na rede elétrica. Entretanto, por uma falha no projeto da PCB, houve um curto-circuito entre o neutro da rede e o plano *ground* de referência da placa (Figura 68).

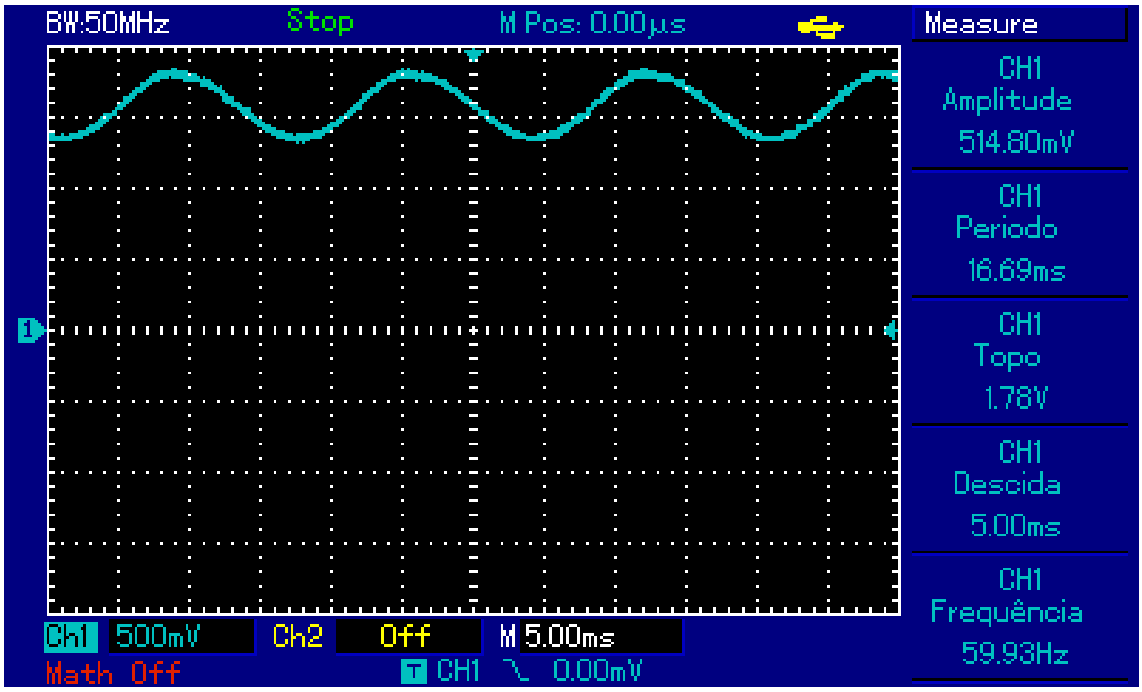
Figura 68 – PCB danificada após curto-circuito.



Fonte: O Autor (2024).

Após identificação e correção do problema, verificou-se o sinal de saída do circuito de medição de tensão para uma entrada de 220 V eficazes (Figura 69).

Figura 69 – Tensão na saída do amplificador diferencial.

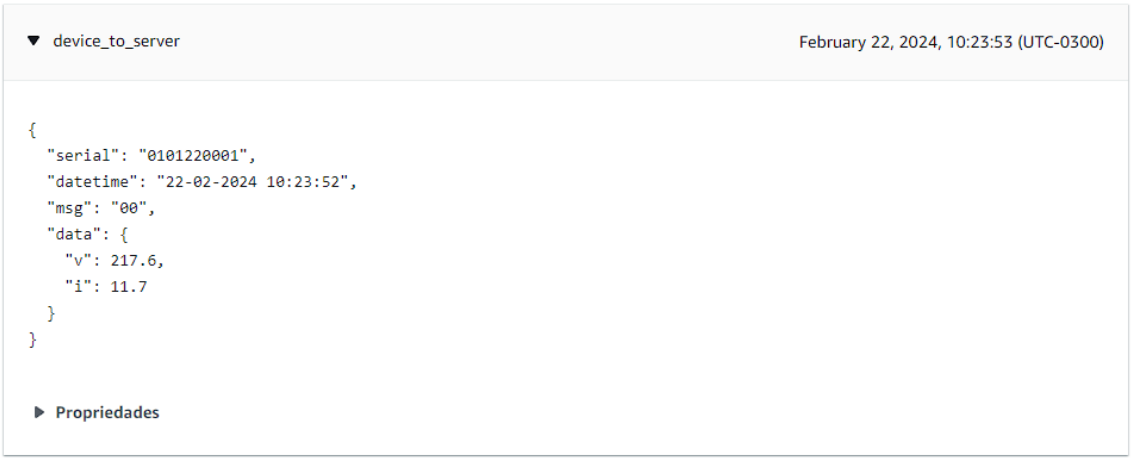


Fonte: O Autor (2024).

4.2.2.3 Testes de Comunicação

Por fim, utilizando o mesmo *firmware* da prova de conceito, valores de corrente e tensão foram amostrados e enviados para o *broker* MQTT do AWS IoT (Figura 70). Assim, foram validados todos os subsistemas do medidor.

Figura 70 – Mensagem recebida pelo agente MQTT do AWS IoT.



Fonte: O Autor (2024).

4.3 GERENCIADOR DE MESSAGENS

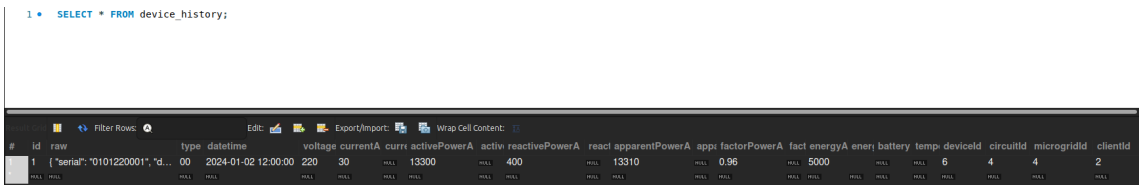
Após configuração do ambiente na Nuvem AWS, isto é, configuração do *Broker* MQTT do AWS IoT, programação da função do AWS Lambda para direcionamento das mensagens ao servidor gerenciador, criação de uma instância de banco de dados no Amazon RDS e a execução do gerenciador de mensagens no ambiente do EC2, efetuou-se um teste para validar se uma mensagem enviada por um dispositivo era corretamente processada e armazenada.

Para isso, utilizou-se o MQTT *Test Client* para emular a publicação de uma mensagem de um medidor (Figura 71) e, então, verificou-se que ela foi armazenada no banco de dados (Figura 72).

Figura 71 – Simulação de uma mensagem do dispostivo publicada através do MQTT *Test Client*.

Fonte: O Autor (2024).

Figura 72 – Mensagem armazenada no banco de dados.



Fonte: O Autor (2024).

4.4 PLATAFORMA WEB

Foram desenvolvidas rotas para API, além de uma interface gráfica para aplicação web.

4.4.1 Rotas API

As tabelas abaixo apresentam as rotas implementadas no projeto, descrevendo suas funcionalidades como método HTTP utilizado, *path* ou *endpoint*, e uma breve descrição de sua função e quem possui autorização de acessar determinado recurso. Nesse contexto, existem rotas que podem ser acessadas por qualquer usuário e outras que só podem ser acessadas pelo administrador.

Tabela 12 – Roteamento de clientes.

Nome	**Método**	**Path**	**Autorização**	**Descrição**
list	GET	/client	Administrador	Lista todos os clientes cadastrados
detail	GET	/client/[id]	Administrador	Obtém informações de um cliente específico
create	POST	/client	Administrador	Cadastra cliente
update	PUT	/client/[id]	Administrador	Atualiza dados de um cliente
enable	PUT	/client/enable/[id]	Administrador	Ativa um cliente (status = 1)
disable	PUT	/client/disable/[id]	Administrador	Desativa um cliente (status = 0)
remove	DELETE	/client/[id]	Administrador	Remove um cliente

Tabela 13 – Roteamento de usuários.

Nome	**Método**	**Path**	**Autorização**	**Descrição**
list	GET	/account	Administrador	Lista todos os usuários cadastrados
detail	GET	/account/[id]	Administrador	Obtém informações de um usuário
create	POST	/account	Administrador	Cadastra usuário
update	PUT	/account/[id]	Administrador	Atualiza dados de um usuário
enable	PUT	/account/enable/[id]	Administrador	Ativa usuário (status = 1)
disable	PUT	/account/disable/[id]	Administrador	Desativa usuário (status = 0)
remove	DELETE	/account/[id]	Administrador	Remove usuário

Tabela 14 – Roteamento de microrredes.

Nome	**Método**	**Path**	**Autorização**	**Descrição**
list	GET	/microgrid	Todos	Lista todas as microrredes cadastradas
detail	GET	/microgrid/[id]	Todos	Obtém informações de uma microrrede
create	POST	/microgrid	Administrador	Cadastra uma microrrede
update	PUT	/microgrid/[id]	Administrador	Atualiza dados de uma microrrede
enable	PUT	/microgrid/enable/[id]	Administrador	Ativa uma microrrede (status = 1)
disable	PUT	/microgrid/disable/[id]	Administrador	Desativa uma microrrede (status = 0)
remove	DELETE	/microgrid/[id]	Administrador	Remove uma microrrede

Tabela 15 – Roteamento de circuitos.

Nome	**Método**	**Path**	**Autorização**	**Descrição**
list	GET	/circuit	Todos	Lista todos os circuitos cadastrados
list by BG	GET	circuit/microgrid/[id]	Todos	Lista os circuitos de uma microrrede específica
detail	GET	/circuit/[id]	Todos	Obtém informações de um circuito
create	POST	/circuit	Administrador	Cadastra circuito
update	PUT	/circuit/[id]	Administrador	Atualiza dados de um circuito
enable	PUT	/circuit/enable/[id]	Administrador	Ativa um circuito (status = 1)
disable	PUT	/circuit/disable/[id]	Administrador	Desativa um circuito (status = 0)
remove	DELETE	/circuit/[id]	Administrador	Remove um circuito

Tabela 16 – Roteamento de dispositivos.

Nome	**Método**	**Path**	**Autorização**	**Descrição**
list	GET	/device	Todos	Lista todos os dispositivos cadastrados
list by circuit	GET	/device/circuit/[id]	Todos	Lista os dispositivos de um circuito específico
list by MG	GET	/device/microgrid/[id]	Todos	Lista os dispositivos de uma microrrede específica
detail	GET	/device/[id]	Todos	Obtém informações de um device
create	POST	/device	Administrador	Cadastra um device
update	PUT	/device/[id]	Administrador	Atualiza dados de um device
link	PUT	/device/link/[id]	Administrador	Vincula um dispositivo a um circuito
unlink	PUT	/device/unlink/[id]	Administrador	Desvincula um dispositivo de um circuito
enable	PUT	/device/enable/[id]	Administrador	Ativa um device (status = 1)
disable	PUT	/device/disable/[id]	Administrador	Desativa um device (status = 0)
remove	DELETE	/device/[id]	Administrador	Remove um device

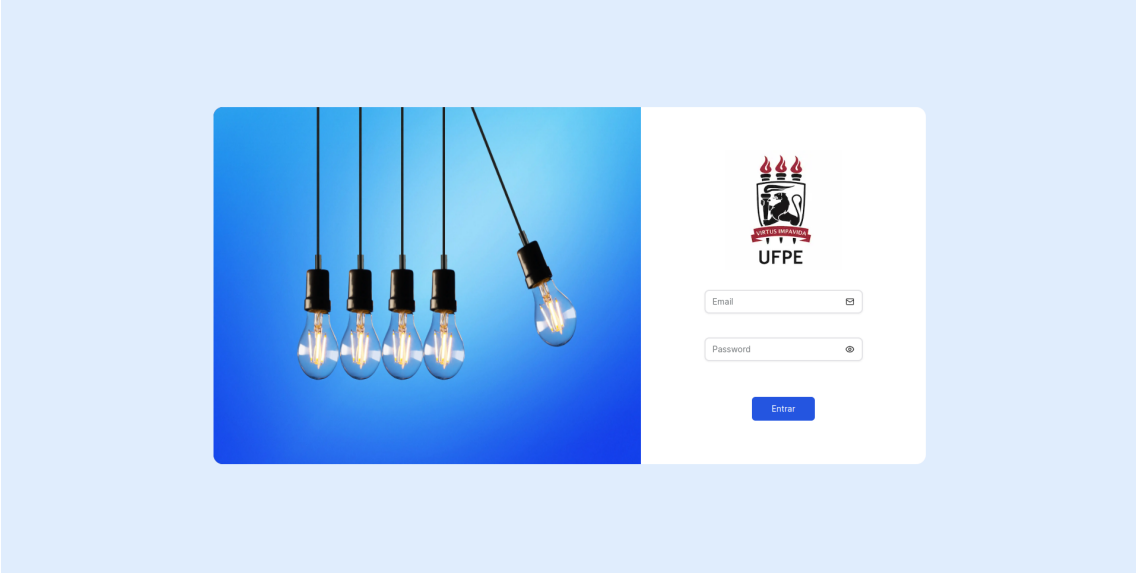
Tabela 17 – Roteamento de autenticação.

Nome	**Método**	**Path**	**Autorização**	**Descrição**
login	POST	/auth/login	Todos	Efetua login do usuário
logout	POST	/auth/logout	Todos	Efetua login do usuário
me	POST	/auth/me	Todos	Verifica se o usuário possui sessão aberta e realiza login automaticamente
recover	POST	/auth/recover	Todos	Recuperação de senha

4.4.2 Interfaces de Usuário

Como apresentado na Seção 3.4.3, foi desenvolvida uma interface gráfica para permitir a integração do usuário com a plataforma. Inicialmente, para efetuar a autenticação do usuário e permitir o acesso aos recursos da aplicação web, criou-se uma tela de *Sign In* (Figura 73).

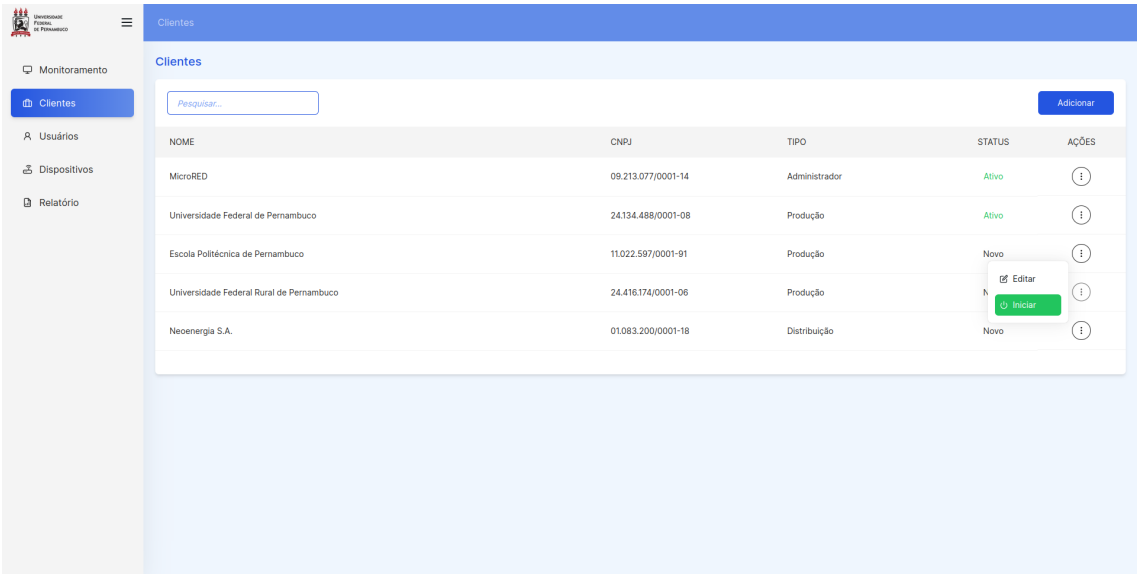
Figura 73 – Página de autenticação da plataforma.



Fonte: O Autor (2024).

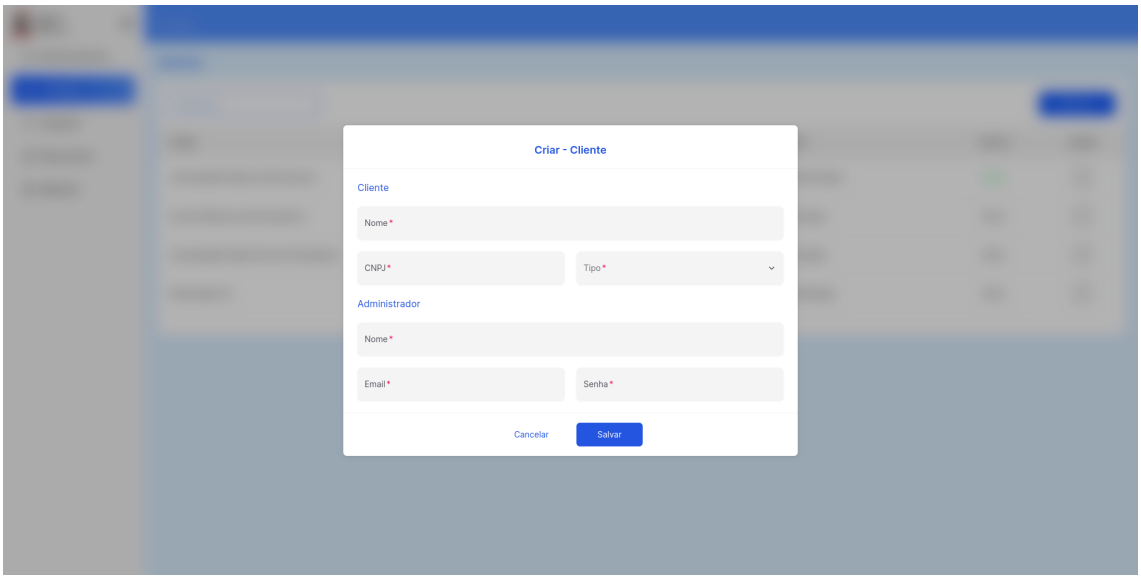
Dentro do ambiente, é possível navegar entre os menus de monitoramento, clientes, usuários, dispositivos e relatório. As páginas de clientes, usuários e dispositivos são responsáveis por fornecer capacidade de visualizar, criar, editar e deletar essas entidades do banco de dados. Dessa forma, suas telas são semelhantes e possuem as seguintes funcionalidades: listagem dos elementos, filtro de pesquisa e formulários para criar, editar e deletar. As Figuras 74 e 75 mostram um exemplo da tela de listagem e de um formulário de criação da entidade clientes.

Figura 74 – Listagem dos clientes cadastrados na plataforma.



Fonte: O Autor (2024).

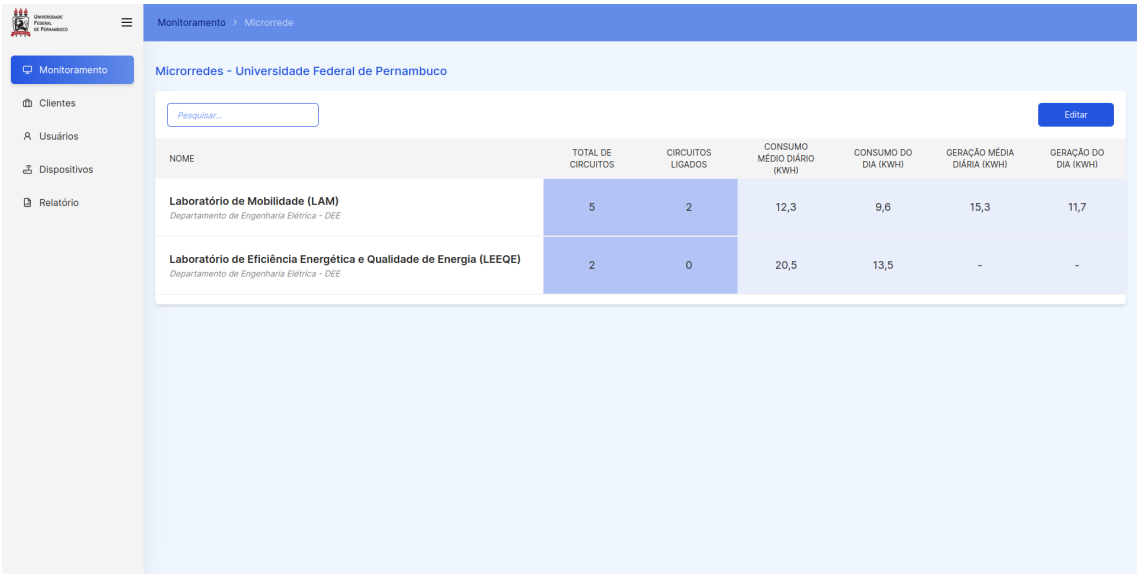
Figura 75 – Formulário para adicionar ou editar clientes.



Fonte: O Autor (2024).

Além disso, foram implementadas interfaces para o monitoramento das microrredes e circuitos. A partir delas é possível ver as informações enviadas pelo medidor, como consumo e geração. A Figura 76 apresenta a tela de monitoramento das microrredes cadastradas no cliente Universidade Federal de Pernambuco. Já a Figura 77, exibe o detalhamento dos dados elétricos adquiridos pelo sistema.

Figura 76 – Tela de monitoramento das microrredes de um cliente.



Fonte: O Autor (2024).

Figura 77 – Detalhamento de um exemplo de microrrede.

Fonte: O Autor (2024).

5 CONCLUSÕES

O sistema de monitoramento inteligente de energia aplicado a microrredes desenvolvido nesse trabalho atende os principais objetivos propostos. Isso porque a PCB do *smartmeter* foi projetada e testada, validando sua funcionalidade, enquanto o sistema Web foi capaz de armazenar, processar e exibir as informações medidas.

Durante os testes da PCB, foram encontradas pequenas falhas de projeto em que o circuito de medição de tensão desenhado na placa provocou um curto-circuito. Entretanto, como a PCB foi projetada adicionando redundâncias em diversos subsistemas, para garantir uma possível ratificação em caso de eventuais problemas, foi possível realizar correções manualmente por meio de soldas e substituição de componentes. Nessa perspectiva, o protótipo atendeu os objetivos propostos do trabalho, adquirindo as grandezas elétricas e enviando para a nuvem. Não obstante, a fim de otimizar o medidor, é importante corrigir no projeto as falhas observadas nos testes e, então, realizar a fabricação de uma segunda versão.

O Arduino IDE se mostrou uma ferramenta interessante para desenvolvimento do *firmware* a fim de validação das funcionalidades do medidor. Entretanto, o *software* apresenta limitações por abstrair camadas de implementação de *drivers*. Dessa forma, com o aumento da complexidade do processamento realizado pelo microcontrolador, isto é, com a adição de novas *features*, mais sensores e etc, é necessário buscar outras alternativas para a programação do *firmware*. Uma opção é utilizar o *Espressif IoT Development Framework* (ESP-IDF), que fornece todas as funcionalidades necessárias para manipulação de hardware em baixo nível.

O sistema Web desenvolvido consegue gerenciar as mensagens enviadas por diferentes dispositivos e realizar a exibição no navegador. Isso porque a infraestrutura de banco de dados, do sistema de gerenciamento de mensagens, do servidor de dados e do servidor de aplicação foi implementada em ambientes escaláveis na nuvem AWS. Dessa forma, ele pode ser utilizado para monitorar dezenas de microrredes e outras aplicações. Ademais, com o monitoramento de diversos sistemas, pode-se verificar as principais informações relevantes e assim construir *features* para realizar análises de falhas e controle de operação.

Por fim, os testes realizados mostraram que o medidor é capaz de adquirir e enviar as principais grandezas elétricas, enquanto a plataforma é responsável por gerenciar e exibir. Entretanto, não foi possível devido ao tempo de desenvolvimento do projeto realizar testes exaustivos com os medidores em campo a fim de detectar falhas no sistema e averiguar a precisão e acurácia do medidor. Dessa forma, faz-se necessário avaliar a precisão e a confiabilidade do *smartmeter* em diferentes cenários.

5.1 TRABALHOS FUTUROS

A medição inteligente de energia pode ser aplicada em diversos setores da engenharia. Dessa forma, o *smartmeter* desenvolvido neste trabalho pode ser implementado em diferentes aplicações, tais como medição de equipamentos, medição residencial, etc. Assim, é fundamental realizar adequações tecnológicas na comunicação, medição e exibição dos dados. Dessa forma, as melhorias e evoluções propostos para trabalhos futuros são:

- Avaliar a precisão e confiabilidade do medidor em diferentes cenários;
- Desenvolvimento do firmware utilizando uma ferramenta que permita o completo controle das funcionalidades do microcontrolador;
- Estudos e implementação de outras tecnologias de comunicação IoT para o medidor, como LoRaWAN, NB-IoT, CAT-M1, entre outras;
- Integração com protocolos de comunicação industrial, como Modbus;
- Desenvolvimento de um medidor trifásico;
- Análise dos dados obtidos pelo *smartmeter* por meio de inteligência artificial a fim de obter diagnósticos sobre os sistemas monitorados;
- Construção de novas *features* para o sistema Web.

REFERÊNCIAS

AKOREDE, M. F.; HIZAM, H.; POURESMAEIL, E. Distributed energy resources and benefits to the environment. *Renewable and sustainable energy reviews*, Elsevier, v. 14, n. 2, p. 724–734, 2010.

ALEXANDER, M. S. C. K. *Fundamentos de Circuitos Elétricos*. [S.l.]: Bookman, 2014.

AWS, A. W. S. *Fundamentos da Nuvem AWS*. 2023. [Online; accessed 31-January-2024]. Disponível em: <https://aws.amazon.com/pt/getting-started/cloud-essentials/>.

AWS, A. W. S. *O que é MQTT?* 2023. [Online; accessed 31-January-2024]. Disponível em: <https://aws.amazon.com/pt/what-is/mqtt/>.

AWS, A. W. S. *O que é uma API (interface de programação de aplicações)?* 2023. [Online; accessed 31-January-2024]. Disponível em: <https://aws.amazon.com/pt/what-is/api/>.

AWS, A. W. S. *O que é uma API RESTful?* 2023. [Online; accessed 31-January-2024]. Disponível em: <https://aws.amazon.com/pt/what-is/restful-api/>.

AWS, A. W. S. *O que é AWS? Como funciona Amazon Web Services?* 2024. [Online; accessed 31-January-2024]. Disponível em: <https://aws.amazon.com/pt/what-is-aws/>.

AWS, A. W. S. *What is Amazon API Gateway?* 2024. [Online; accessed 31-January-2024]. Disponível em: <https://docs.aws.amazon.com/apigateway/latest/developerguide/welcome.html>.

AWS, A. W. S. *What is Amazon EC2?* 2024. [Online; accessed 31-January-2024]. Disponível em: <https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/concepts.html>.

AWS, A. W. S. *What is Amazon Relational Database Service (Amazon RDS)?* 2024. [Online; accessed 31-January-2024]. Disponível em: <https://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/Welcome.html>.

AWS, A. W. S. *What is AWS IoT?* 2024. [Online; accessed 31-January-2024]. Disponível em: <https://docs.aws.amazon.com/iot/latest/developerguide/what-is-aws-iot.html>.

AWS, A. W. S. *What is AWS Lambda?* 2024. [Online; accessed 31-January-2024]. Disponível em: <https://docs.aws.amazon.com/lambda/latest/dg/welcome.html>.

BARNES, M.; KONDOH, J.; ASANO, H.; OYARZABAL, J.; VENTAKARAMANAN, G.; LASSETER, R.; HATZIARGYRIOU, N.; GREEN, T. Real-world microgrids-an overview. In: *2007 IEEE International Conference on System of Systems Engineering*. [S.l.: s.n.], 2007. p. 1–8.

BELLIDO, M. M. H. *Microrredes elétricas: Uma proposta de implementação no Brasil*. Tese (Doutorado em Planejamento Energético) — Universidade Federal do Rio de Janeiro, Rio de Janeiro, 2018.

CHAPMAN, S. J. *Fundamentos de Máquinas Elétricas*. [S.l.]: bookman, 2013.

CHEN, P. P. The entity-relationship model - toward a unified view of data. *ACM Trans. Database Syst.*, v. 1, n. 1, p. 9–36, 1976. Disponível em: <http://dblp.uni-trier.de/db/journals/tods/tods1.html#Chen76>.

ELECROW. *SCT-013 Datasheet*. 2011. [Online; accessed 31-January-2024]. Disponível em: <https://www.elecrow.com/download/SCT013-000_datasheet.pdf>.

ELMASRI, S. B. N. R. *Sistemas de Banco de Dados*. [S.l.]: Pearson, 2019.

EPE, E. de P. E. *Plano Decenal de Expansão de Energia 2030*. 2021. [Online; accessed 31-January-2024]. Disponível em: <https://www.epe.gov.br/sites-pt/publicacoes-dados-abertos/publicacoes/PublicacoesArquivos/publicacao-490/PDE%202030_RevisaoPosCP_rv2.pdf>.

FILHO, S. de M. *Medição de Energia Elétrica*. [S.l.]: Universidade Federal de Pernambuco, 1980.

HEUSER, C. A. *Projeto de Banco de Dados*. [S.l.]: Bookman, 2009.

INNOVATORSGURU. *ZMPT101B - MICRO PRECISION VOLTAGE TRANSFORMERS*. 2024. [Online; accessed 31-January-2024]. Disponível em: <https://innovatorsguru.com/wp-content/uploads/2019/02/ZMPT101B.pdf>.

JR, A. P. *Amplificadores Operacionais e Filtros Ativos*. [S.l.]: Bookman, 2015.

JSON. *Introdução ao JSON*. 2024. [Online; accessed 31-January-2024]. Disponível em: <https://www.json.org/json-pt.html>.

KOYANAGI, F. *Medidor de tensão e corrente AC com ESP32 e Arduino*. 2022. [Online; accessed 31-January-2024]. Disponível em: <https://innovatorsguru.com/wp-content/uploads/2019/02/ZMPT101B.pdf>.

KUROSE, K. W. R. J. F. *Redes de Computadores e a Internet: Uma Abordagem Top-Down*. [S.l.]: Bookman, 2021.

LASSETER, R. Microgrids. In: *2002 IEEE Power Engineering Society Winter Meeting. Conference Proceedings (Cat. No.02CH37309)*. [S.l.: s.n.], 2002. v. 1, p. 305–308 vol.1.

LLORET, J.; TOMAS, J.; CANOVAS, A.; PARRA, L. An integrated iot architecture for smart metering. *IEEE Communications Magazine*, v. 54, n. 12, p. 50–57, 2016.

LOPES FILIPE JOEL SOARES, P. M. R. A. J. A. P. Integration of electric vehicles in the electric power system. *Proceedings of the IEEE*, IEEE, v. 99, n. 1, p. 168–183, 2010.

MARTINS, M. C. D. G. N. *Implantação de Microrrede Dotada de Sistema de Geração Solar Fotovoltaica e Sistema de Armazenamento de Energia. Estudo de Caso: Microrrede do Laboratório de Armazenamento e Mobilidade.* Dissertação (Mestrado) — Universidade Federal de Pernambuco - UFPE, 2022.

MICROSOFT. *Publisher-Subscriber pattern*. 2024. [Online; accessed 31-January-2024]. Disponível em: <https://learn.microsoft.com/en-us/azure/architecture/patterns/publisher-subscriber>.

NICOLAU, C. T.; SOUZA, R. C.; FROTA, M. N. Medição de energia elétrica: impactos da mudança tecnológica no setor jurídico de uma concessionária distribuidora de energia elétrica. *Rio de Janeiro*, 2013.

PINHO, M. A. G. J. T. *Manual de Engenharia para sistemas fotovoltaicos*. [S.l.]: CEPEL - CRESESB, 2014.

RIVERA, R.; ESPOSITO, A. S.; TEIXEIRA, I. Redes elétricas inteligentes (smart grid): oportunidade para adensamento produtivo e tecnológico local. Banco Nacional de Desenvolvimento Econômico e Social, 2013.

SEMICONDUCTOR, O. *TVS/Zener Theory and Design Considerations - Handbook*. 2005. [Online; accessed 31-January-2024]. Disponível em: <http://www.reallyreallyrandom.com/zener/media/Zener_Theory_and_Design.pdf>.

SYSTEMS, E. *ESP32 MINI Datasheet*. 2023. [Online; accessed 31-January-2024]. Disponível em: <https://docs.espressif.com/projects/arduino-esp32/en/latest/installing.html#installing-using-arduino-ide>.

SYSTEMS, E. *ESP32 Series Datasheet*. 2024. [Online; accessed 31-January-2024]. Disponível em: <https://www.espressif.com/sites/default/files/documentation/esp32_datasheet_en.pdf>.

SYSTEMS, E. *Installing using Arduino IDE*. 2024. [Online; accessed 31-January-2024]. Disponível em: <https://www.espressif.com/sites/default/files/documentation/esp32-mini-1_datasheet_en.pdf>.

TANENBAUM, N. F. A. *Redes de Computadores*. [S.l.]: Bookman, 2021.

TASTAN, M. Internet of things based smart energy management for smart home. *KSII Transactions on Internet and Information Systems (TIIS)*, Korean Society for Internet Information, v. 13, n. 6, p. 2781–2798, 2019.

VIJAYAPRIYA, T.; KOTHARI, D. P. Smart grid: an overview. *Smart Grid and Renewable Energy*, Scientific Research Publishing, v. 2, n. 4, p. 305–311, 2011.

ZHENG, J.; GAO, D. W.; LIN, L. Smart meters in smart grid: An overview. In: *2013 IEEE Green Technologies Conference (GreenTech)*. [S.l.: s.n.], 2013. p. 57–64.

ZIEGLER, S.; WOODWARD, R. C.; IU, H. H.-C.; BORLE, L. J. Current sensing techniques: A review. *IEEE Sensors Journal*, IEEE, v. 9, n. 4, p. 354–376, 2009.

APÊNDICE A – DIAGRAMA ENTIDADE-RELACIONAMENTO ESTENDIDO

Os diagramas entidade-relacionamento estendidos (EER) são representações avançadas de banco de dados bastante semelhantes aos ER regulares. Esses diagramas apresentam mais informações sobre o banco de dados modelado como subtipos, supertipos, especialização e generalização, categoria, herança de atributos e relacionamento. A Figura apresenta o diagrama EER desenvolvido neste trabalho, destacando as tabelas, os relacionamentos, os atributos, os tipos de dados, bem como as chaves primárias e estrangeiras.

Figura 78 – Diagrama EER do banco de dados desenvolvido no trabalho.