Universidade Federal de Campina Grande
Centro de Engenharia Elétrica e Informática
Departamento de Engenharia Elétrica

Trabalho de Conclusão de Curso

Sistema de Monitoramento de Consumo de Energia para Carregamento de Veículos Elétricos em Condomínios

José Tayrone Santos de Oliveira

Campina Grande - PB

Outubro de 2024

José Tayrone Santos de Oliveira

Sistema de Monitoramento de Consumo de Energia para Carregamento de Veículos Elétricos em Condomínios

Trabalho de Conclusão de Curso submetido à Coordenação de Graduação em Engenharia Elétrica da Universidade Federal de Campina Grande como parte dos requisitos necessários para a obtenção do grau de Engenheiro Eletricista.

Universidade Federal de Campina Grande - UFCG

Centro de Engenharia Elétrica e Informática - CEEI

Departamento de Engenharia Elétrica - DEE

Coordenação de Graduação em Engenharia Elétrica - CGEE

Jaidilson Jó da Silva, D.Sc.

(Orientador)

Campina Grande - PB

Outubro de 2024

José Tayrone Santos de Oliveira

Sistema de Monitoramento de Consumo de Energia para Carregamento de Veículos Elétricos em Condomínios

Trabalho de Conclusão de Curso submetido à Coordenação de Graduação em Engenharia Elétrica da Universidade Federal de Campina Grande como parte dos requisitos necessários para a obtenção do grau de Engenheiro Eletricista.

Aprovado em ____ /____ /____

Marcus Marinho Bezerra
Universidade Federal de Campina Grande
Avaliador

Jaidilson Jó da Silva
Universidade Federal de Campina Grande
Orientador

Campina Grande - PB
Outubro de 2024

Dedico este trabalho à minha mãe, Tânia, que batalhou, muitas vezes em silêncio, para que seu filho tivesse o que ela não teve.

Agradecimentos

Sou grato, antes de tudo, a Deus, por me conceder a capacidade de sonhar e por sempre me mostrar que cada desafio enfrentado tem um propósito. Nada é em vão.

Minha eterna gratidão às minhas duas mães, Tânia e Terezinha. Não é todo mundo que tem o privilégio de contar com duas figuras maternas, fontes de amor e apoio incondicional. Mulheres fortes e de corações enormes. Tudo o que conquistei e o que ainda virá, será sempre, por vocês.

Sou profundamente grato a toda minha família, sem exceções, inclusive aqueles que não tive a oportunidade de conhecer. Não podemos esquecer que, nossas raízes estão no sangue. Agradeço especialmente à minha irmã, Deborah, por ser minha fonte constante de inspiração, pelo cuidado e por sempre estar ao meu lado.

Sou grato à minha namorada, que sonha ao meu lado, me incentiva e está sempre celebrando todas minhas conquistas, me fazendo lembrar que sou capaz de grandes realizações.

Agradeço a todos os amigos que fiz durante essa jornada, sem vocês não seria possível chegar aonde cheguei. Sei da capacidade de cada um e desejo todo sucesso do mundo a todos vocês. Um agradecimento especial à Ana Beatriz, que esteve comigo desde a concepção desse sonho até vê-lo se tornar realidade.

Agradeço a meu orientador, professor Jaidilson Jó da Silva, por acreditar no meu trabalho e por ser uma inspiração como profissional, tornando a graduação mais leve e enriquecedora. Além disso, minha gratidão ao professor Marcus Marinho Bezerra, um homem de coração generoso, que mesmo sem me conhecer me ofereceu todo suporte que estava em seu alcance. Gostaria, ainda, de agradecer a todos os bons professores que através da compreensão e paciência, me ofereceram apoio ao longo dessa jornada. Em especial, aos meus antigos professores do Petrônio Colégio e Curso; a dedicação e o diferencial de cada um de vocês deixaram marcas no meu desenvolvimento que refletem no homem que sou hoje.

Por fim, um agradecimento mais do que especial à criança que desmontava os objetos de casa apenas para entender como funcionavam. Essa criança é a razão de eu ter chegado até aqui.

"E um sonho de menino é maior que de gente grande,
porque fica mais próximo da realidade."
(José Lins do Rego)

Resumo

É evidente a constante expansão da presença de veículos elétricos nas estradas ao redor do mundo. De forma que, a infraestrutura e as tecnologias que acomodam essa expansão vem sendo continuadamente desenvolvidas. Porém, como tudo que é novo, surge um debate acerca da acomodação dessas infraestruturas dentro de ambientes residenciais. Pensando nisso, e tendo como motivação a atenuação dessa problemática, surge o desenvolvimento deste trabalho, utilizando a placa de desenvolvimento ESP32 e sensoriamento de corrente e tensão, buscando trazer uma solução para distribuição justa dos custos tarifários conforme o consumo da energia durante a utilização de estações de recarga instaladas em condomínios. Além disso, foi empregado tecnologias diversas como a utilização de um banco de dados e o monitoramento do consumo através de uma interface web.

Palavras-chave: Veículos Elétricos; Estação de Recarga; ESP32; Sensoriamento; Monitoramento; Banco de Dados.

Abstract

It is evident that there is a constant expansion of the presence of electric vehicles on roads around the world. Therefore, the infrastructure and technologies that accommodate this expansion have been continually developed. However, as anything new, a debate arises regarding the accommodation of these infrastructures within residential environments. Keeping this in mind, and motivated by the aim of mitigating this issue, the development of this work emerged, using the ESP32 development board along with current and voltage sensing, seeking to provide a solution for the fair distribution of tariff costs according to energy consumption during the use of charging stations installed in condominiums. Moreover, various technologies were employed, such as the use of a database and the monitoring of consumption through a web interface.

Keywords: Electric Vehicles; Charging Station; ESP32; Sensing; Monitoring; Database.

Lista de ilustrações

Lista de abreviaturas e siglas

EV	*Electric Vehicle*
VC	Veículo a Combustão
ABVE	Associação Brasileira do Veículo Elétrico
IEA	*International Energy Agency*
BEV	*Battery Electric Vehicle*
HEV	*Hybrid Electric Vehicle*
PHEV	*Plug-in Hybrid Electric Vehicle*
FCEV	*Fuel Cell Electric Vehicle*
KERS	*Kinetic Energy Recovery System*
MCI	Motor de Combustão Interna
ABNT	Associação Brasileira de Normas Técnicas
NBR	Norma Brasileira Regulamentadora
IEC	*International Electrotechnical Commission*
AC	*Alternating Current*
DC	*Direct Current*
V	Volt
ICCPD	*In-Cable Control and Protection Device*
ADC	*Analog-to-Digital Converter*
mV	Milivolt
RMS	*Root Mean Square*
IoT	*Internet of Things*
GPIO	*General Purpose Input/Output*
UART	*Universal Asynchronous Receiver-Transmitter*

SPI	*Serial Peripheral Interface*
SDIO	*Secure Digital Input Output*
I2C	*Inter-Integrated Circuit*
I2S	*Inter-IC Sound*
PWM	*Pulse Width Modulation*
NA	Normalmente Aberto
NF	Normalmente Fechado
CI	Circuito Integrado
RFID	*Radio-Frequency Identification*
UID	*Unique Identifier*
IDE	*Integrated Development Environment*
DB	*Database*

Sumário

1 Introdução

Depois de anos de investimento e expectativa, vários sinais indicam que a tecnologia utilizada para funcionamento dos veículos elétricos (EV, do inglês *Eletric Vehicle*) está se expandindo rapidamente. Espera-se que, nas próximas duas décadas, ocorra uma substituição gradual dos veículos a combustão (VC) por modelos elétricos (OZORIO et al., 2021).

Ademais, a enorme presença desses veículos nas estradas é consequência do forte investimento das empresas automobilísticas, em pesquisa e desenvolvimento para tornar os EV mais acessíveis, eficientes, autônomos e atrativos para o consumidor final.

A partir de dados publicados na "*The Global EV Outlook*", publicação anual desenvolvida pela *International Energy Agency*, também denominada IEA, com o apoio dos membros da *Electric Vehicles Initiative*, houve um registro de aproximadamente 14 milhões de vendas de carros elétricos no ano de 2023, totalizando 40 milhões de carros elétricos presentes em todo o mundo.

No Brasil, de acordo com a Associação Brasileira do Veículo Elétrico (ABVE), que visa incentivar o desenvolvimento e a utilização de EV no país, foi constatado que o número de veículos leves eletrificados emplacados nos seis primeiros meses de 2024, atingiu 79 mil unidades. Dessa forma, quase alcançando as 93 mil unidades emplacadas durante todo o ano de 2023. (FROMER, 2024)

Assim como possuir um VC está totalmente atrelado a necessidade de abastecê-lo, possuir um EV está atrelado a realizar o seu carregamento regularmente. Porém, diferentemente dos VC, que em poucos minutos são reabastecidos, o processo de recarga das baterias de um veículo elétrico demanda tempo, dependendo do método de carregamento utilizado. Sendo assim, o usuário deve se planejar antecipadamente, podendo optar pelo método ideal de acordo com suas necessidades, levando em consideração as estações de recarga públicas e privadas ou o carregamento residencial, feito em casa.

À luz disso, é estimado que a infraestrutura brasileira deva aumentar anualmente 45% até o ano de 2040 para que seja possível acomodar o crescente número de EV nas ruas do país (Mckinsey & Company, 2023). Ou seja, a implementação de tecnologias que facilitem o carregamento e atendam a necessidade dos usuários para um carregamento seguro e cômodo é crucial para o processo de crescimento da eletromobilidade.

No entanto, existe um impasse quanto à utilização e instalação dessas estações de recarga em determinados condomínios. Os proprietários de EV que residem em condomínios enfrentam uma série de obstáculos, incluindo a falta de acordo entre os moradores,

os altos custos para aquisição de estações individuais, problemas com regulamentação, infraestrutura e os custos tarifários da energia.

Nesse contexto, com a popularização dos microcontroladores, o desenvolvimento de um sistema embarcado que seja capaz de monitorar, em tempo real, o consumo de energia elétrica em uma estação de carregamento, realizar cálculos automáticos dos custos e fazer a autenticação dos moradores aptos, pode proporcionar ao ecossistema do condomínio uma maior transparência e gestão eficiente dos gastos e dos recursos energéticos. Em vista disso, com a contabilização precisa dos gastos individuais, o sistema tornará o condomínio um ambiente equitativo e auxiliará na popularização dos veículos elétricos, ajudando ainda a atenuar o debate sobre a implementação dos centros de recarga nesses locais.

1.1 Objetivos

1.1.1 Objetivo Geral

O objetivo neste trabalho é desenvolver um sistema, com o auxílio de um microcontrolador, que sirva de protótipo de um sistema em maior escala. O sistema deve monitorar a utilização da energia elétrica de forma equitativa e transparente em estações de recarga para carros elétricos localizados em condomínios, possibilitando ainda a utilização do sistema em qualquer outro ambiente ou cenário em que sua aplicação seja relevante.

1.1.2 Objetivos Específicos

- Fazer a identificação e autenticação dos moradores;
- Coletar, processar e registrar, em tempo real, determinadas informações e parâmetros operacionais de uma estação de carregamento durante o seu funcionamento;
- Implementar um algoritmo para cálculo automático dos custos com a energia consumida por cada morador;
- Implementar um banco de dados para armazenar os dados coletados;
- Desenvolver uma interface para auxiliar a administração do condomínio na visualização dos dados;

1.2 Organização do Trabalho

Além do capítulo introdutório atual, o trabalho em questão possui mais 4 capítulos cruciais, conforme descrito a seguir.

Capítulo 2: Fundamentação Teórica - Responsável por apresentar a base teórica que possibilitará o entendimento do propósito e das decisões do projeto.

Capítulo 3: Metodologia do Projeto - Apresentação dos materiais utilizados e das fases de desenvolvimento.

Capítulo 4: Resultados e Discussões - Exposição dos resultados obtidos e exposição das limitações e falhas do projeto.

Capítulo 5: Considerações Finais - Análise do projeto final, assim como de possíveis alterações futuras e melhorias que podem tornar o projeto mais completo.

2 Fundamentação Teórica

Neste capítulo em particular, será evidenciado todo trabalho de revisão da literatura acerca da eletromobilidade e dos componentes envolvidos no processo de recarga de um EV, com ênfase no contexto brasileiro. Inicialmente será discutido a respeito da categorização dos veículos que se utilizam da eletricidade para gerar propulsão, incluindo os veículos elétricos a bateria (BEV, do inglês *Battery Electric Vehicles*), veículos elétricos híbridos (HEV, do inglês *Hybrid Electric Vehicles*), veículos híbridos plug-in (PHEV do inglês *Plug-in Hybrid Electric Vehicles*) e veículos movidos por células de combustível (FCEV, do inglês *Fuel Cell Electric Vehicle*). Logo após isso, será abordado sobre a infraestrutura voltada para o carregamento dos EVs, distinguindo os tipos de carregadores e os modos de carga.

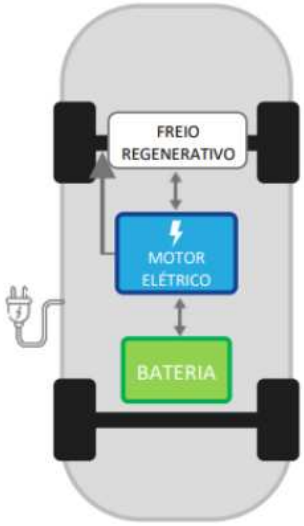
2.1 Veículos Elétricos

Para atender as necessidades e as preferências de uma população é normal visualizar no mercado uma alta diversidade de produtos. De modo equivalente, o mercado de EV segue a mesma tendência, que, surgindo como uma alternativa ao veículo movido a combustão, já estão presentes no Brasil através de 250 modelos diferentes de veículos leves eletrificados, colocados à disposição por 40 montadoras (JORDAO; KAKUTA, 2023), cumprindo as mais variadas demandas que o mercado exige. Sabendo disso, é possível categorizar os veículos eletrificados com base em sua configuração e no modo como se utilizam da energia elétrica. Nos tópicos seguintes serão expostos algumas dessas categorias de veículos presentes no mercado brasileiro.

2.1.1 Veículos Elétricos a Bateria

Os BEV são conhecidos por serem veículos puramente elétricos, no sentido de que o veículo transforma energia elétrica advinda exclusivamente do banco de baterias em energia cinética para tracionar as rodas (NeoCharge, 2024). Sendo assim, não geram qualquer tipo de poluentes no ar já que não há combustão envolvida no processo. Porém, é um veículo que depende exclusivamente da capacidade de suas baterias para o seu funcionamento, portanto é necessário realizar carregamentos periódicos com o auxílio de um carregador portátil ou através de estações (EV Connect, 2024).

Figura 1 – Esquematização de um BEV.

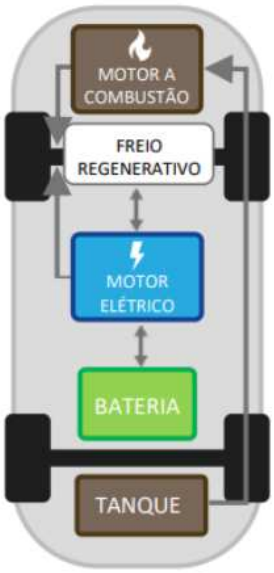


Fonte: Adaptado de (Associação Brasileira de Engenharia Automotiva, 2023)

2.1.2 Veículos Elétricos Híbridos

Os HEV trazem a combinação de dois tipos de processos de geração de energia para produzir propulsão. Ou seja, essa categoria de veículo eletrificado conta com um Motor de Combustão Interna (MCI) e motores elétricos que ajudam tanto na propulsão como no carregamento das baterias. Se utilizam da tecnologia Kers (Kinetic Energy Recovery System), ou Sistema de Recuperação de Energia Cinética, o que contribui com a eficiência energética, já que, ao fazer uso dos freios consegue recuperar parte da energia cinética e armazenar nas baterias, minimizando o uso do combustível sempre que possível e dispensando a utilização dos carregadores portáteis e as estações (Canal VE, 2023).

Figura 2 – Esquematização de um HEV.

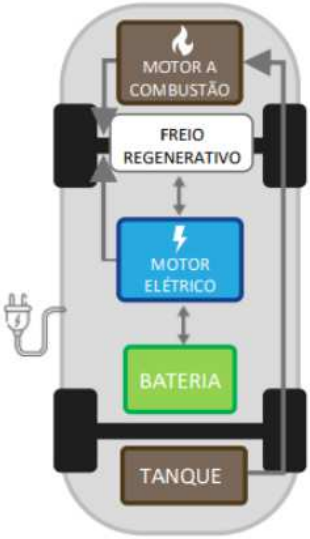


Fonte: Adaptado de (Associação Brasileira de Engenharia Automotiva, 2023)

2.1.3 Veículos Elétricos Híbridos Plug-in

Os PHEV são uma opção intermediária entre os BEV e os HEV, visto que contam com um MCI e um motor elétrico, possibilitando que o veículo opere tanto no modo elétrico quanto no modo híbrido. Contudo, quando comparamos a autonomia de um PHEV durante o modo elétrico com veículos puramente elétricos, é perceptível sua capacidade inferior de se manter autônomo durante determinado período, atingindo de 40 a 60 km, ao passo que os elétricos conseguem ultrapassar 300 km de autonomia (BEZERRA, 2021). Ao fim da carga nas baterias, o PHEV passa a utilizar o MCI, garantindo que o veículo consiga terminar o seu percurso (NITI Aayog, 2024). Além disso, oferece a oportunidade não somente de carregar suas baterias através da tecnologia de freio regenerativo como o HEV, mas também através de carregamento externo como um BEV.

Figura 3 – Esquematização de um PHEV.

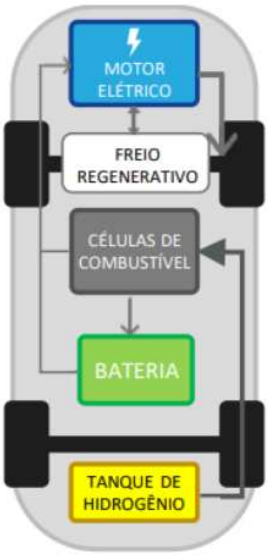


Fonte: Adaptado de (Associação Brasileira de Engenharia Automotiva, 2023)

2.1.4 Veículos Elétricos Movidos a Célula de Combustível

Os FCEV operam através de uma reação eletroquímica entre hidrogênio e oxigênio. Esse processo acaba por gerar água, calor e energia elétrica, sem qualquer liberação de poluentes, o que acaba tornando uma solução eficiente e limpa (ANFAVEA, 2023). Sendo assim, com a eletricidade gerada através das células combustíveis, realizam o carregamento das baterias, sem necessidade de uma fonte externa. Ainda não possuem infraestrutura satisfatória, como os outros tipos de veículos citados anteriormente.

Figura 4 – Esquematização de um FCEV.



Fonte: Adaptado de (Associação Brasileira de Engenharia Automotiva, 2023)

2.2 Modos e métodos de carregamentos

O termo mobilidade elétrica, ou eletromobilidade, se refere à inserção dos EV no cotidiano das cidades, englobando os mais diversos componentes que facilitam e contribuem com a viabilização da transição energética e do avanço dessa tecnologia (CHAVES et al., 2023). Nesse contexto, faz-se necessário destacar os modos e métodos utilizados que possibilitam o processo de carregamento dos EV e que desempenham um papel importante para expansão da eletromobilidade. Deve-se ainda levar em conta o cumprimento das normas técnicas que delimitam os modos de carregamento e os requisitos necessários para suas aplicações, assim como dos padrões de instalações elétricas, como no caso da ABNT NBR 5410, que estabelece as condições necessárias que as instalações elétricas de baixa tensão devem atender para preservação dos bens e das pessoas (ABNT, 2004), assim como da ABNT NBR IEC 61851, que define os requisitos que devem ser aplicados junto ao sistema de alimentação para acomodar o carregamento, delimitando quatro modos distintos de carregamentos que serão apresentados a seguir.

2.2.1 Modo 1 - Carregamento Lento

O modo 1 de carregamento é o mais simples e básico, visto que a conexão com o carro e a rede é efetuada através de uma tomada de corrente doméstica comum por cabo fixo. Desse modo, o EV é conectado a uma rede de corrente alternada (AC), sofrendo limitação de 16A e 250 V, em rede monofásica (LISTON, 2024). Por causa disso, o tempo de recarga do veículo é extenso, classificando-o como um modo de carregamento lento. Ainda, a instalação deve contar com um circuito de proteção, impedindo que ocorra sobrecarga e fuga de corrente, sendo assim, tal modo foi proibido em determinados países devido ao risco de sobreaquecimento dos cabos e uso incorreto do dimensionamento das tomadas (CAETANO, 2021)

Figura 5 – Modo 1 de Carregamento.

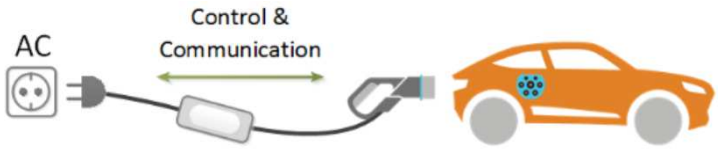
Fonte: Adaptado de (CHARGERS, 2024)

2.2.2 Modo 2 - Carregamento Lento

O modo 2 de carregamento, diferentemente do modo 1, conta com um sistema de proteção e controle chamado ICCPD (do inglês *In-Cable Control and Protection Device*) (NEOCHARGE, 2023). Ou seja, por meio do cabo, o veículo consegue se proteger de

eventuais surtos e ainda se comunicar com a rede, verificando sempre que possível sua integridade, e o estado do aterramento. Neste modo, encontra-se limitações de correntes de até 32A e tensão de 250 V, em rede monofásica, e 480V, em rede trifásica (LISTON, 2024). É normal que o carregador já venha com o EV, são os chamados carregadores portáteis, como pode ser visto na figura 7, e são fáceis de se utilizar, conectando uma extremidade na tomada e a outra no EV.

Figura 6 – Modo 2 de Carregamento.



Fonte: Adaptado de (CHARGERS, 2024)

Figura 7 – Carregador Portátil para Veículos Elétricos.

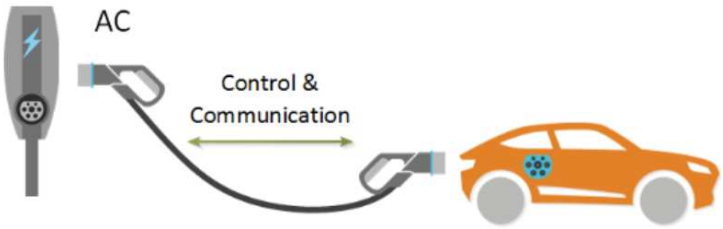
Fonte: (IPEC Carregadores, 2024)

2.2.3 Modo 3 - Semirrápido

O modo 3 de carregamento dispõe de uma instalação específica, normalmente encontrada em estações fixas de carregamento nas vias públicas e nos ambientes privados (Associação Brasileira de Engenharia Automotiva, 2023). Ou seja, a conexão entre a rede e o EV é realizada por uma tomada de uso específico, dimensionada unicamente para o processo de carregamento dos veículos, e que atende as normas definidas pela ABNT NBR 5410 e pela ABNT NBR IEC 61851. Se utiliza de corrente alternada (AC), com corrente de até 32A e potência de 7,4kW monofásico e com corrente 63A e potência aproximada de

43kW trifásica (ELECTROMAPS, 2024). Esse modo pode ser visualizado por meio de dois métodos principais: na configuração de totens fixos no chão, ou fixados na parede como pode ser visto na figura 9, conhecidos como wallboxes. Sendo assim, se tornam uma alternativa popular pela comodidade e facilidade na instalação. Ainda assim, como no modo 2, conseguem realizar a comunicação com a rede.

Figura 8 – Modo 3 de Carregamento.



Fonte: Adaptado de (CHARGERS, 2024)

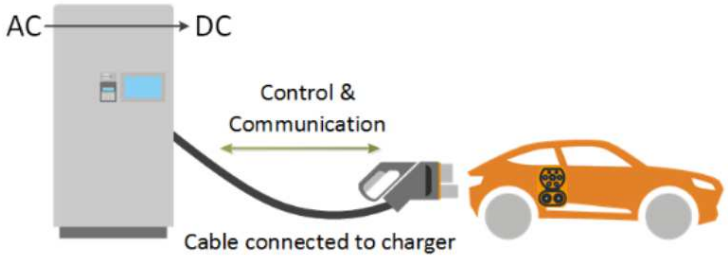
Figura 9 – Carregador Wallbox da BYD juntamente com caixa de proteção.

Fonte: Autoria Própria

2.2.4 Modo 4 - Ultrarrápido

O modo 4 de carregamento é projetado para ser ultrarrápido, realizando a conversão da corrente alternada da rede em corrente contínua (DC). A conexão entre o EV e central de recarga conta com controle e comunicação, além de segurança adicional (FERREIRA et al., 2022). Sendo assim, devido a conversão em corrente contínua (DC), admite correntes de maiores intensidades e potência de até 350kW (LISTON, 2024)).

Figura 10 – Modo 4 de Carregamento.



Fonte: Adaptado de (CHARGERS, 2024)

2.3 Cálculo de Corrente RMS usando o Sensor ACS712 e ESP32

O ESP32 é uma placa de desenvolvimento que possui dois conversões analógico-digital, ADC1 e ADC2, distribuídos entre 18 canais, com resolução de 12 bits (ENGINEERS, 2024). Isso significa que, os valores lidos no pinos analógicos podem variar de 0 a 4095, onde 0 corresponde ao valor 0 de tensão e 4095 o valor máximo da tensão de referência, que neste caso é 3.3V.

O ACS712 é um sensor, que se utiliza do efeito *Hall*, para monitorar a quantidade de corrente, alternada ou contínua, presente na seção do circuito em que o mesmo está conectado.

Tendo como objetivo obter os valores de correntes em um determinado circuito, pode-se realizar a integração entre o ESP32 e o ACS712, para captar e realizar o condicionamento do sinal lido na entrada do pino ADC do ESP32. Sendo assim, para que realizar tal condicionamento, temos que:

O valor da tensão gerado pelo sensor e enviada para o canal de entrada ADC durante o sensoriamento será dado pela Equação 2.1.

$$V = \frac{V_{\text{analog}} \cdot 3.3}{4095} \tag{2.1}$$

Onde V_{analog} se refere ao valor analógico lido na entrada do pino.

Portanto, sabendo que o valor da sensibilidade do ACS712-20A é de 100mV/A, pode-se interpretar que, a cada 0.1V (ou 100 mV) que o sensor gera na entrada do pino, é detectado 1 ampere de corrente no circuito em análise. Logo, o valor de corrente detectado pelo sensor é calculado através da equação 2.2.

$$I = \frac{V}{0.1} \tag{2.2}$$

Porém esse valor é o valor da corrente instantânea, ou seja, o valor da corrente em um instante específico do tempo. Dessa forma, para obter o valor de corrente RMS será necessário calcular uma determinada quantidade suficiente de valores de corrente em um ciclo de onda completo, elevar cada valor ao quadrado, garantindo que os valores positivos e negativos serão considerados, e calcular a média da soma de todos os quadrados calculados. A equação 2.3 formaliza o cálculo da corrente RMS. A figura **??**

$$I_{\text{rms}} = \sqrt{\frac{I_1^2 + I_2^2 + I_3^2 + \cdots + I_T^2}{\text{T}}} \tag{2.3}$$

Onde T se refere ao número de medições realizadas pelo sensor.

3 Metodologia do Projeto

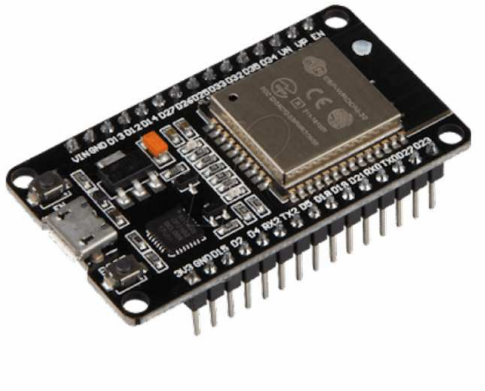
3.1 Materiais

O projeto em questão, se concentra na utilização de materiais para construção de um protótipo. Dessa forma, não são todos os materiais que serão vistos nos tópicos abaixo — que servirão para o desenvolvimento do projeto em escala real. Contudo, a idéia é que o projeto seja desenvolvido pensando no modo de carregamento 3 e utilizado nas estações residenciais, ou nas chamadas "*Wallbox*".

3.1.1 ESP32

O ESP-32 é uma plataforma desenvolvida pela *Espressif Systems*, empresa responsável por elaborar soluções utilizando protocolos Wifi e Bluetooth, focando no baixo consumo de energia e na democratização da tecnologia, o ESP32 se destaca por oferecer um alto desempenho a um custo acessível (ESPRESSIF, 2024).

Figura 11 – Placa de desenvolvimento ESP32.



Fonte: (INSTIPER, 2024)

Devido a essas características, o ESP32 se torna uma escolha popular para o desenvolvimento de aplicações de Internet das Coisas (IoT, do inglês *Internet of Things*), já que consegue proporcionar maior poder de processamento facilitando a execução de multitarefas e de tarefas mais complexas. Ademais, a integração de protocolos *Wifi* e *Bluetooth* facilita a comunicação com outros dispositivos envolvidos, tornando o ESP32 eficiente quando as aplicações requerem conectividade e interação entre módulo e dispositivos (HERCOG et al., 2023).

O ESP32 possui as seguintes especificações:

- Microprocessador Tensilica Xtensa LX6 de 32 bits

  Frequência de clock: de 80MHz até 240 MHz

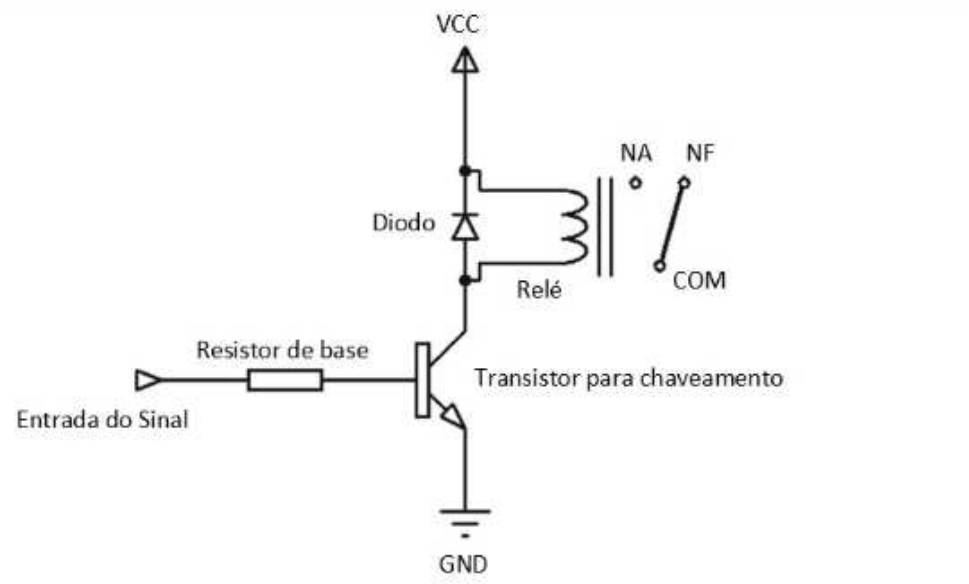
- **Conectividade sem fio**: *Wi-Fi*: 802.11 b/g/n/e/i (802.11n @ 2,4 GHz até 150 Mbit/s)
- *Bluetooth* 4.2, *Bluetooth Low Energy* (BLE)
- **Tensão de nível lógico**: 3,3VDC
- Memória flash de 4mb
- **Alimentação**: via micro USB de 4,5 à 12,0 VDC (Através do pino Vin)
- **GPIO (Pinos de Entrada/Saída de Propósito Geral)**: 34 pinos programáveis, com suporte para interfaces UART (3 canais), SPI (3 canais), SDIO, I2C (2 canais), I2S (2 canais), PWM (2 canais) e PWM motor (3 canais);
- **Capacidades Analógicas**: 18 canais ADC (Conversor Analógico para Digital) de 12 bits e 2 canais DAC (Conversor Digital para Analógico) de 8 bits

Com o auxílio de todas essas especificações, o ESP32 pode ser utilizado para uma gama de aplicações, incluindo *Smart Home*, Automação Industrial, Agricultura Inteligente, entre outras.

3.1.2 Módulo Relé

O relé é um dispositivo, que pode ser eletromecânico ou eletrônico, construído com o intuito de realizar o controle de dispositivos que demandam alta potência com o auxílio de sinais gerados em menores tensões (MakerHero, 2024). O funcionamento desse dispositivo é baseado em um campo magnético que será gerado sempre que for injetada uma corrente no terminal de comando. Tal campo magnético realiza a comutação das conexões no interior do relé.

Figura 12 – Esquema de funcionamento do Relé.



Fonte: (STA Eletrônica, 2024)

Para o protótipo deste trabalho, será utilizado o Módulo Relé SRD-5VDC-SL-C que pode ser visto na figura 13. Ele suporta tensões alternadas de até 250V e uma corrente de 10A.

Figura 13 – Módulo Relé.

Fonte: (Eletrogate, 2024)

O módulo em questão possui uma face de entrada e uma de saída com 3 pontos de conexão cada, são eles:

- Entrada (Comando):

  VCC – Entrada de 5 VCC para alimentação do módulo

  GND – Terminal para conexão do Terra.

IN – Entrada na qual é aplicado o sinal de comando, conectado à saída digital do microcontrolador.

- Saída (Conexões de carga):

  COM - Conexão Comum com as conexões NA e NF

  NA - Estabelece uma conexão normalmente aberta com o terminal Comum. A carga está desativada quando o Relé está desligado.

  NF - Estabelece uma conexão padrão normalmente fechada com o terminal Comum. A carga está ativa quando o Relé está desligado.

Pensando no desenvolvimento do projeto em escala real, vê-se a necessidade de escolha de um relé que suporte os 32A requeridos pelo modo 3 de carregamento e que tenha entrada suficiente para acomodar o cabo de rede.

3.1.3 Sensor de Tensão - ZMPT101B

Para garantir o controle e o monitoramento do fluxo de energia em circuitos é normal se utilizar de sensores que captam a presença de tensão em pontos desejados. Após a captação, é necessário converter a informação em um sinal que possa ser interpretado facilmente. Para o projeto em questão, será utilizado o módulo sensor de tensão ZMPT101B, apresentado na figura 14, para medir a tensão monofásica.

Figura 14 – Sensor de Tensão ZMPT101B.

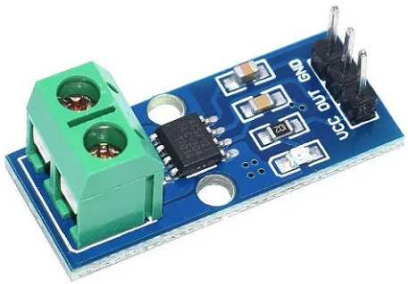
Fonte: (Smart Componentes, 2024)

O módulo sensor ZMPT101B funciona com um divisor de tensão, responsável por converter a tensão de entrada em uma tensão menor na saída. Além disso, conta com um transformador de relação de espiras 1000:1000 e é capaz de medir tensões em corrente alternada de 0 a 250 V com precisão de ±0,5% (Proesi, 2024). Pensando ainda na construção do projeto em escala real, vê-se a necessidade de escolha de um sensor que possua um borne de entrada suficiente para acomodar o cabo de fase da rede.

3.1.4 Sensor de Corrente - ACS712

Assim como é necessário captar os valores de tensão alternada para que o projeto se torne confiável e completo, a obtenção dos valores de corrente é essencial para monitorar o consumo de energia no sistema e analisar o comportamento do equipamento. Sendo assim, foi escolhido o sensor ACS712 com faixa de medição de -20A a +20A para realizar o sensoriamento da corrente alternada durante o processo de carregamento. O sensor pode ser visualizado na figura 15.

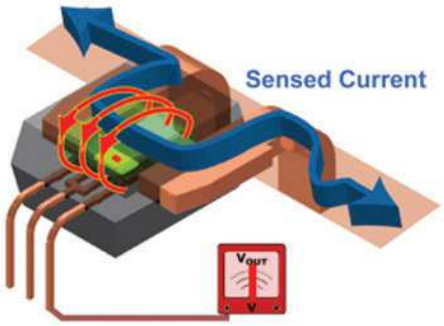
Figura 15 – Sensor de Corrente ACS712.



Fonte: (MasterWalker Shop, 2024)

O sensor em pauta oferece uma boa precisão na detecção de corrente contínua e alternada, com baixo nível de ruído, já que conta com dois capacitores de filtro. Além disso, se baseia no efeito Hall, ou seja, a corrente elétrica medida gera um campo magnético que é rapidamente detectado pelo CI (Circuito Integrado), convertendo em uma tensão proporcional (LI et al., 2010) que pode ser interpretada por um microcontrolador. O esquema de seu funcionamento pode ser visualizado na figura 16.

Figura 16 – Esquema de funcionamento do ACS712.



Fonte: (SparkFun Electronics, 2023)

Idealizando a implementação do projeto em escala real, propõe-se a utilização do sensor de corrente SCT-013 com faixa de leitura de até 100A, por ser capaz de realizar

medições precisas e oferecer segurança. Já que, é um sensor não invasivo, permitindo o sensoriamento da corrente elétrica sem a necessidade de seccionar cabos.

3.1.5 Módulo Leitor RFID

A tecnologia de identificação por radiofrequência é uma técnica de comunicação que não requer contato. O módulo RFID (do inglês, *Radio Frequency Identification*) é capaz de identificar e autenticar usuários através de cartões ou tags com chip RFID. O leitor de rádio frequência detecta uma tag e o sistema captura o seu identificador único, o UID (do inglês, *Unique Identifier*) (KURUNDKAR et al., 2021).

Pensando na construção do protótipo e no desenvolvimento de um controle de acesso à estação de carregamento, será empregado o leitor RFID modelo MFRC522 da marca Mifare, apresentado na figura 17, com tensão de operação de 3,3VDC e frequência de 13,56MHz.

Figura 17 – Módulo Leitor RFID-RC522.

Fonte: (Rytronics, 2024)

3.1.6 Display OLED

Com o objetivo de fornecer transparência na visualização dos dados ao morador que estiver utilizando o sistema, será utilizado um display OLED de 0.96 polegadas para construção do protótipo. O display será responsável por exibir informações relevantes acerca do consumo de energia e dos custos associados, fazendo com que o morador possa acompanhar em tempo real, os dados do seu carregamento. A idéia é que, para desenvolver

o sistema em maior escala, se utilize um display de dimensões ampliadas, ou seja, um display em que o morador consiga visualizar os dados com clareza de uma distância considerável.

Figura 18 – Display OLED 0.96".

Fonte: (PROESI, 2024)

3.1.7 Arduino IDE

O Arduino IDE (do inglês, *Arduino Integrated Development Environment*) é uma plataforma de edição de texto para escrita de códigos, capaz de se conectar e se comunicar com placas Arduino e similares (Arduino, 2024).

Devido sua popularidade e ser de fácil entendimento, será a principal ferramenta a ser utilizada neste projeto, sendo capaz de se comunicar com o ESP32 e oferecer suporte para uma gama de bibliotecas que serão a base do funcionamento dos módulos já apresentados anteriormente. Sua interface pode ser visualizada na figura 19.

Figura 19 – Interface do Arduino IDE.

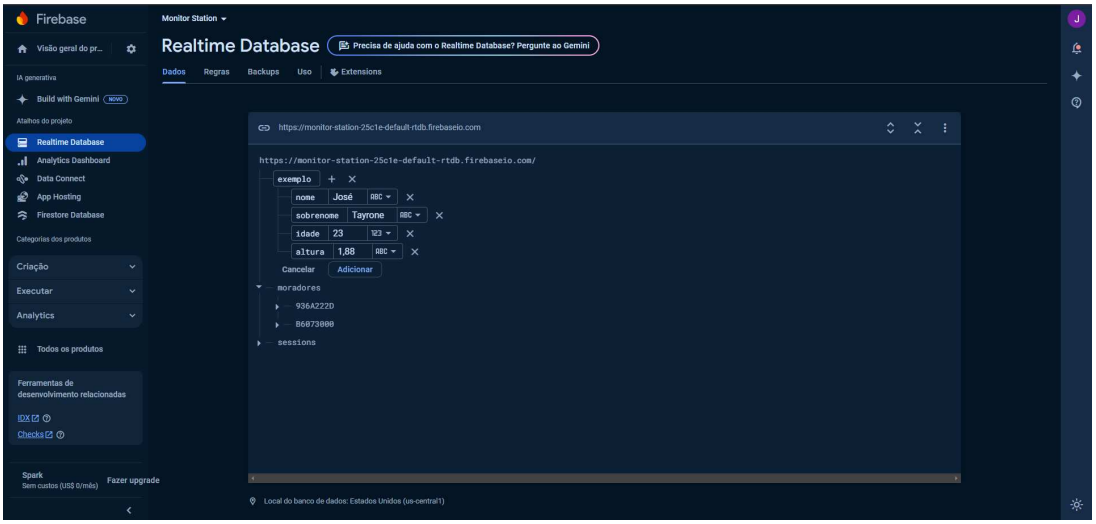
Fonte: Autoria Própria

3.1.8 Firebase - Realtime Database

Desenvolvida pelo Google, o Firebase é uma plataforma que disponibiliza uma gama de serviços voltados ao desenvolvimento de aplicações móveis e web. Entre esses serviços, está o *Realtime Database*, um banco de dados NOSQL que se utiliza da nuvem para armazenar dados no formato JSON (Google, 2024). Sua interface pode ser visualizada na figura 20.

O Realtime Database do Firebase, se mostra uma alternativa interessante para o projeto devido a sincronização em tempo real dos dados com as aplicações. Portanto, isso significa que, ao receber os dados coletados do ESP32, o sistema de banco de dados irá atualizar de forma imediata os dados para o usuário que acessa a interface.

Figura 20 – Interface do Realtime Database do Firebase.



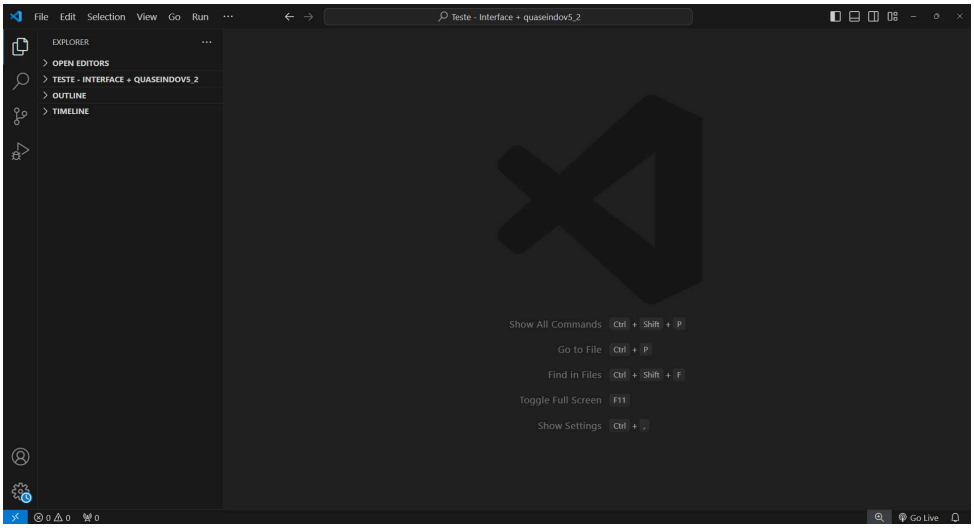
Fonte: Autoria Própria

3.1.9 Visual Studio Code

O Visual Studio Code é um editor de código-fonte, desenvolvido pela Microsoft, disponível para os principais sistemas operacionais, que oferece suporte para as mais conhecidas linguagens de programação, além de dispor de variadas extensões que facilitam o fluxo de trabalho (Microsoft, 2024). Sua interface pode ser visualizada na figura 21.

Levando em consideração os aspectos acima e a construção e implementação dos códigos responsáveis pelo *back-end* e o *front-end* da interface a ser implementada no projeto, será empregado o Visual Studio Code como plataforma de desenvolvimento. Uma vez que, a utilização da extensão *Live Server* irá ser bastante eficaz, permitindo a execução de um servidor local para sincronização com o banco de dados, facilitando os testes.

Figura 21 – Interface do Visual Studio Code.



Fonte: Autoria Própria

3.2 Metodologia

3.2.1 Preparação da IDE

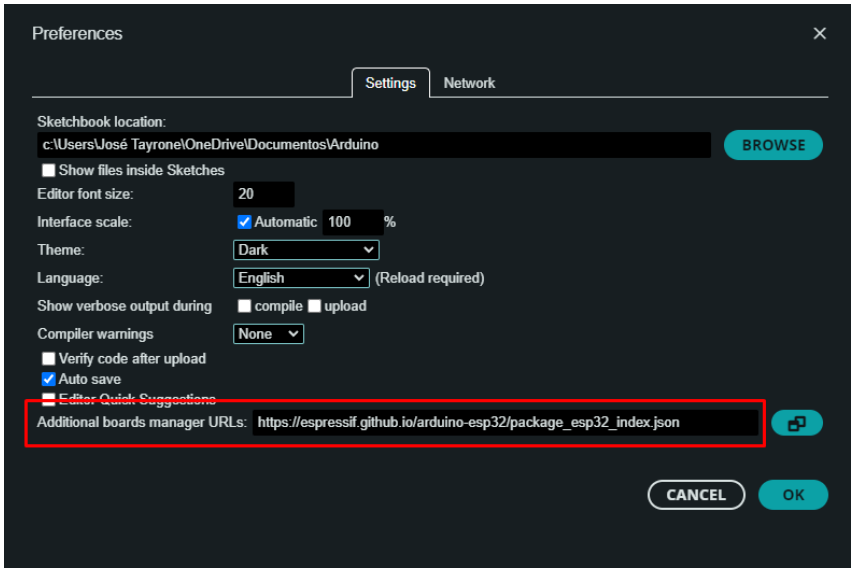
Levando em consideração que o Arduino IDE não reconhece nativamente o pacote de placas de desenvolvimento ESP32, será necessário fornecer a IDE a arquitetura específica na qual a comunicação será estabelecida. Portanto, para isso, com o Arduino IDE aberto é preciso realizar os seguintes passos:

Vá até Arquivo (ou *File*) > Preferências (ou *Preferences*).

Vá até o campo "URLs Adicionais para Gerenciadores de Placas".

Cole a URL "*https://espressif.github.io/arduino-esp32/package_esp32_index.json*".

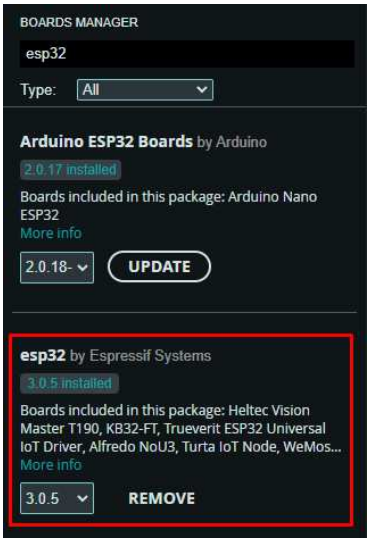
Figura 22 – Configuração da IDE para comunicação com o ESP32.



Fonte: Autoria Própria

Após isso, será necessário ir até Ferramentas (ou *Tools*) > Placa (ou *Boards*) > Gerenciador de Placas (ou *Boards Manager*), pesquisar por "*esp32 by Espressif Systems*" e clicar em Instalar, conforme a figura 23.

Figura 23 – Instalação do pacote esp32.



Fonte: Autoria Própria

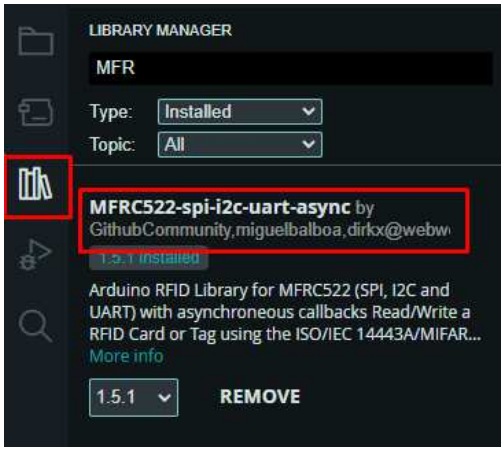
Após os passos anteriores serem concluídos, pode-se seguir em frente e dar início aos testes dos módulos.

3.2.2 Teste entre o módulo RFID e o Relé

O teste entre o módulo RFID e o relé tem como objetivo fazer a verificação da interação entre os dois componentes de modo a garantir que o sistema consiga ativar e desativar o relé com base na leitura de *tags* realizada pelo módulo RFID e da autenticação de chaves previamente cadastradas.

Inicialmente, é necessário instalar corretamente a biblioteca adequada. Para isso, clica-se no terceiro ícone do canto esquerdo do Arduino IDE para entrar no "Gerenciador de Bibliotecas". Logo em seguida, procura-se pela biblioteca do módulo utilizado. Para o projeto em questão, foi utilizada a biblioteca *MFRC522-spi-i2c-uart-async* para lidar com o módulo RFID, como pode ser visto na figura 24.

Figura 24 – Instalação da biblioteca MRFC522.



Fonte: Autoria Própria

Além disso, é comum que as bibliotecas do Arduino IDE disponibilizem exemplos de códigos para facilitar o entendimento de suas funcionalidades e do que são capazes de realizar. Ou seja, tais exemplos ajudam desenvolvedores a aprender como utilizar a biblioteca da melhor maneira de acordo com suas pretensões.

Pensando nisso, foi desenvolvido um código para realizar o teste planejado, consultando a página do GitHub da biblioteca, *Makerspace Leiden RFID*, disponível em: <https://github.com/makerspaceleiden/rfid>, acessado em: 28 set. 2024. Parte do código pode ser visualizada na figura 25.

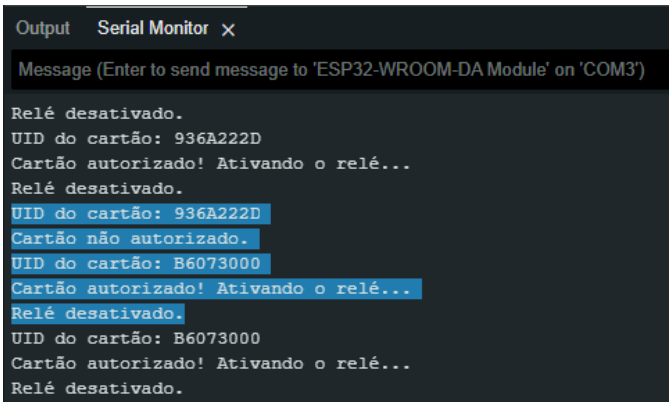
Figura 25 – Função loop do código utilizado para o teste de integração entre relé e RFID.

```
27  void loop() {
28
29    if (!rfid.PICC_IsNewCardPresent()) {  // Verifica se existe um novo cartão
30      return;
31    }
32
33    if (!rfid.PICC_ReadCardSerial()) {  // Tenta fazer a leitura do cartão
34      return;
35    }
36
37    // Obtem o UID do cartão lido
38    String content = "";
39    for (byte i = 0; i < rfid.uid.size; i++) {
40      content += String(rfid.uid.uidByte[i] < 0x10 ? "0" : "");
41      content += String(rfid.uid.uidByte[i], HEX);
42    }
43    content.toUpperCase();  // Converte as letras do UID que acabou de ser lido para letras maiúsculas
44
45    Serial.print("UID do cartão: ");
46    Serial.println(content);
47
48
49    if (content == card1 || content == card2) {  // Faz a verificação se o cartão é autorizado
50      Serial.println("Cartão autorizado! Ativando o relé...");
51      digitalWrite(RELAY_PIN, LOW);   // Liga o relé
52      delay(10000);                   // Estabelece um tempo de 10 segundos
53      digitalWrite(RELAY_PIN, HIGH);  // Desliga o relé
54      Serial.println("Relé desativado");
55
56    } else {
57      Serial.println("Cartão não autorizado.");
58    }
59
60    rfid.PICC_HaltA();  // Para de processar o cartão
61  }
```

Fonte: Autoria Própria

Após implementar o código e fazer o *upload* para a ESP32, foi possível realizar o teste de funcionamento da integração entre os dois módulos, aproximando um cartão cadastrado através do código e outro que não foi cadastrado. O resultado do teste pode ser visto na figura 26. Dessa forma, é possível concluir que os módulos estão funcionando como adequado e o resultado obtido foi satisfatório e suficiente para dar prosseguimento para os próximos testes.

Figura 26 – Saída do Monitor Serial do teste de integração entre relé e RFID.

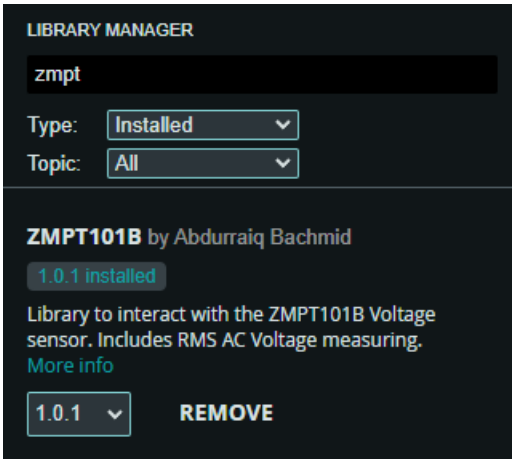


Fonte: Autoria Própria

3.2.3 Teste do Sensor de Tensão

Para realizar o teste de funcionamento do sensor de tensão ZMPT101B, optou-se por incluir a biblioteca "*ZMPT101B by Abdurraziq Bachmid*"na Arduino IDE. Através dessa biblioteca será possível obter os valores de tensão em RMS através da função "*getRmsVoltage*", sem a necessidade de cálculos adicionais para adequar o sinal analógico lido na entrada do pino da ESP32.

Figura 27 – Instalação da biblioteca ZMPT101B.



Fonte: Autoria Própria

Assim como será explicado nessa seção, o passo a passo para deixar o sensor pronto para o sensoriamento correto da tensão pode ser encontrado na página do GitHub da biblioteca, *Abdurraziq - ZMPT101B-arduino*, disponível em: <https://github.com/Abdurraziq/ZMPT101B-arduino>, acessado em: 24 set. 2024.

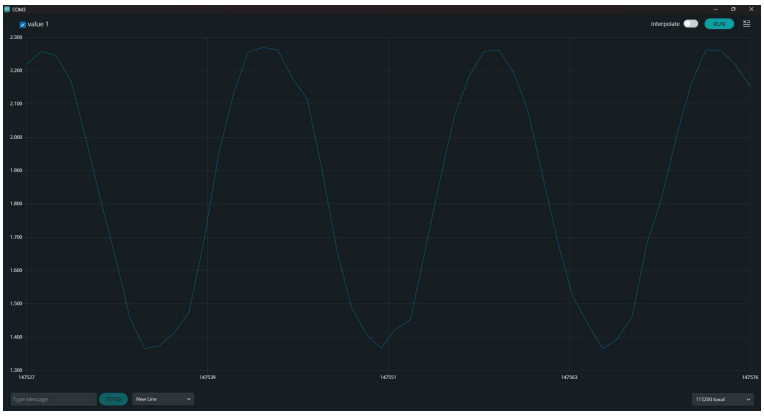
Inicialmente, por meio do *upload* do código disponível no exemplo 1, dentro da IDE clica-se em Ferramentas > *Serial Plotter*. Com esse procedimento seremos capazes de visualizar a forma de onda da fonte medida. É importante lembrar que o sensor deve estar devidamente alimentado e conectado a fonte na qual se deseja medir a tensão.

A idéia é que, girando o trimpot presente no módulo do sensor, a forma de onda se aproxime mais da onda vista na figura 29(a) e se afaste da onda vista na figura 29(b), objetivando que a onda não tenha distorções e nem tenha seus picos cortados.

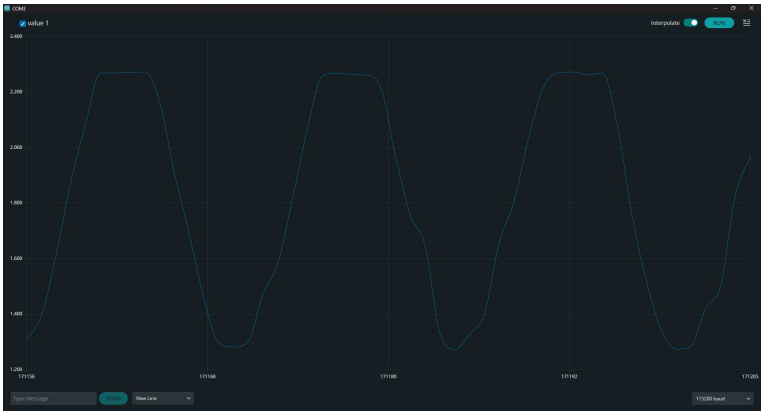
```
{
void setup() {
    Serial.begin(115200);
}

void loop() {
    Serial.println(analogRead(SENSOR_PIN)); //Inserir o pino utilizado
    delayMicroseconds(1000);
}
}
```

Exemplo 1 – *Código para visualizar forma de onda da tensão.*

Figura 28 – Análise da captação da senoide de tensão da rede.



(a) Imagem com sinal de tensão bom



(b) Imagem com sinal de tensão ruim

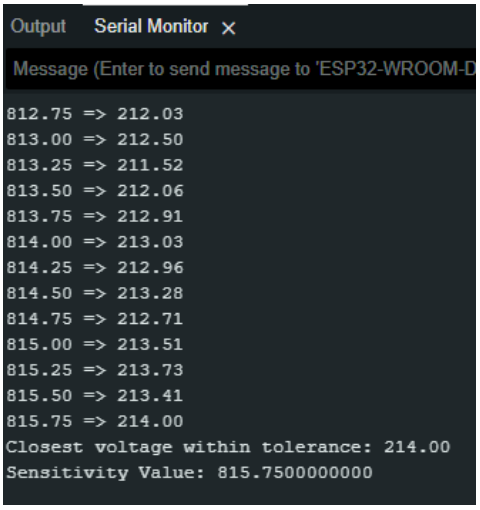
Fonte: Autoria Própria

O próximo passo consiste em utilizar o código *calibrate.ino* disponível na pasta *Exemplos/calibrate* da página do GitHub para obter a sensibilidade do sensor, repassando

o valor real da tensão que você objetiva monitorar para "*#define ACTUAL_VOLTAGE*".

Para obter o valor real é sugerido utilizar um multímetro. O resultado desse passo pode ser visto na figura 29, e o valor da tensão medido no momento do teste foi de 214V.

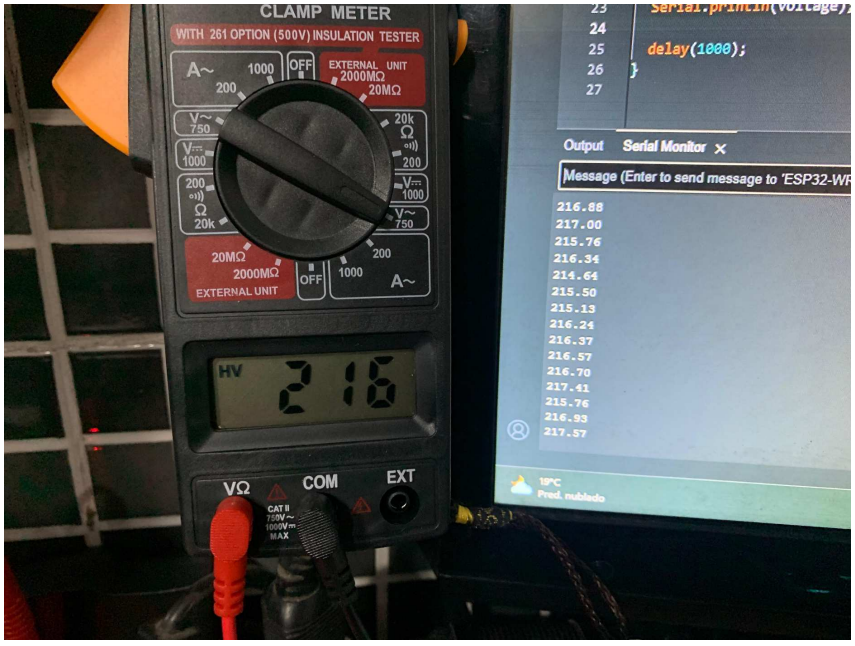
Figura 29 – Valor de sensibilidade da tensão no momento do teste.



Fonte: Autoria Própria

Por fim, fazendo o upload do código *simple_usage.ino* disponível na pasta *Exemplos/simple_usage*, pode-se obter o monitoramento da tensão. O resultado, que foi satisfatório, pode ser visto na figura 30.

Figura 30 – Comparativo entre tensão medida no multímetro e tensão monitorada pelo sensor.



Fonte: Autoria Própria

3.2.4 Teste do Sensor de Corrente

Dessa vez, não será necessário a utilização de nenhuma biblioteca para monitorar a corrente. Portanto, será feito o condicionamento do sinal captado pela entrada analógica do ESP32, como visto na seção 2.3, para que no final, tenha-se o valor da corrente RMS.

A aplicação de todas as informações vistas anteriormente, foram condensadas no código que pode ser visto na figura 31, para realizar o teste do sensor ACS712.

Figura 31 – Código para teste do sensor ACS712.

```
void loop() {

  currentSquaredSum = 0; // Reinicia a soma dos quadrados

  for (int i = 0; i < numSamples; i++) {

    sensorValue = analogRead(analogPin);
    voltage = sensorValue * (3.3 / 4095.0);
    voltage = voltage - zeroOffset;    // Eliminar ruído
    current = voltage / (sensitivity/1000); // Sensibilidade em V/A
    currentSquaredSum += current * current;    // Soma dos quadrados das correntes
    delayMicroseconds(sampleInterval); // 167 µs de atraso entre amostras para frequencia de 60Hz

  }

  // Calcular o valor RMS da corrente
  float currentRMS = sqrt(currentSquaredSum / numSamples);

  // Imprimir o valor de corrente RMS no monitor serial
  Serial.print("Corrente RMS: ");
  Serial.print(currentRMS);
  Serial.println(" A");

  delay(1000);
}
```

Fonte: Autoria Própria

Para fazer o teste se o sensor realmente está captando os valores corretos de tensão foi utilizado uma escova secadora conectada ao sistema teste e um amperímetro para fazer a comparação dos valores. Os resultados foram satisfatórios e podem ser vistos na figura 32.

Figura 32 – Resultado do teste do sensor ACS712 comparado com medição do amperímetro.

Fonte: Autoria Própria

3.2.5 Criação do Banco de Dados em Tempo Real no Firebase

O emprego do banco de dados (DB, do inglês *database*) em tempo real tem como propósito atualizar nossa interface com os dados provenientes da placa sempre que uma sessão de carregamento for encerrada. Sendo assim, a idéia é que a ESP32 colete os dados, envie esses dados para o servidor da Firebase para que assim a interface possa acessá-los e os apresentar de forma simples e organizada. A esquematização deste processo está presente na figura 33.

Figura 33 – Diagrama de comunicação ESP32-Firebase-Interface.

Fonte: Compilação do Autor[1]

Para dar início a construção do DB, é necessário possuir uma conta Google e acessar a página do Firebase. Logo após isso, adiciona-se um projeto em "Criar um Projeto", dá-se o nome e prossegue seguindo as instruções que o Firebase fornece.

Posteriormente, no painel do lado esquerdo da tela, clica-se em Criação > Realtime Database > Regras e definimos os campos "*write*" e "*read*" para "*true*".

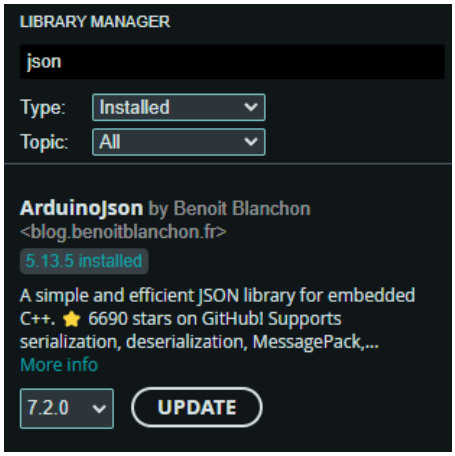
O Firebase fornece muitas outras ferramentas para compor o projeto, porém apenas o Realtime Database será utilizado. Sendo assim, o banco de dados está iniciado com as regras definidas para os testes e pronto para ser estruturado no formato Json. A estrutura pensada para o desenvolvimento do projeto se encontra no exemplo 2.

```
{
    "moradores": {
        "B6073000": {
            "apartamento": 201,
            "nome": "Tayrone"
        }
    }
    "sessions": {
        "sessao_1": {
            "UID_morador": "B6073000",
            "consumo_Kwh": 0.04,
            "custo": 0.03,
            "data_uso": "1/10/2024",
            "horarioEntrada": "17:35:33",
            "horarioSaida": "18:11:36",
            "morador": "José",
            "tarifa": 0.73,
            "tempoGasto_h": 0.6
        }
    }
}
```

Exemplo 2 – *Exemplo de JSON com dados de sessão de uso*

Portanto, agora que o lado do Firebase está configurado, o próximo passo é instalar as bibliotecas necessárias na IDE do Arduino. Dessa vez, será preciso instalar duas bibliotecas, a primeira será a "*ArduinoJson by Benoit Blanchon*", que se encontra disponível na própria IDE, logo sua instalação pode ser realizada normalmente.

Figura 34 – Instalação da biblioteca ArduinoJson.



Fonte: Autoria Própria

A segunda biblioteca precisará ser baixada através da página do GitHub, *ArtronShop - IOXhop_FirebaseESP32*, disponível em: <https://github.com/ArtronShop/IOXhop_FirebaseESP32>, acessado em: 01 out. 2024. Para incluir a segunda biblioteca na IDE, será necessário ir em Rascunho > Incluir Biblioteca > Adicionar Biblioteca .ZIP e adicionar o arquivo .zip baixado através do GitHub.

A biblioteca em questão possui quatro funções principais, *GET*, *SET*, *PUSH*, e *STREAM*. Porém do lado da IDE, só será empregado a função SET, repassando as variáveis importantes para o banco de dados. Um exemplo de implementação está apresentado no exemplo 3.

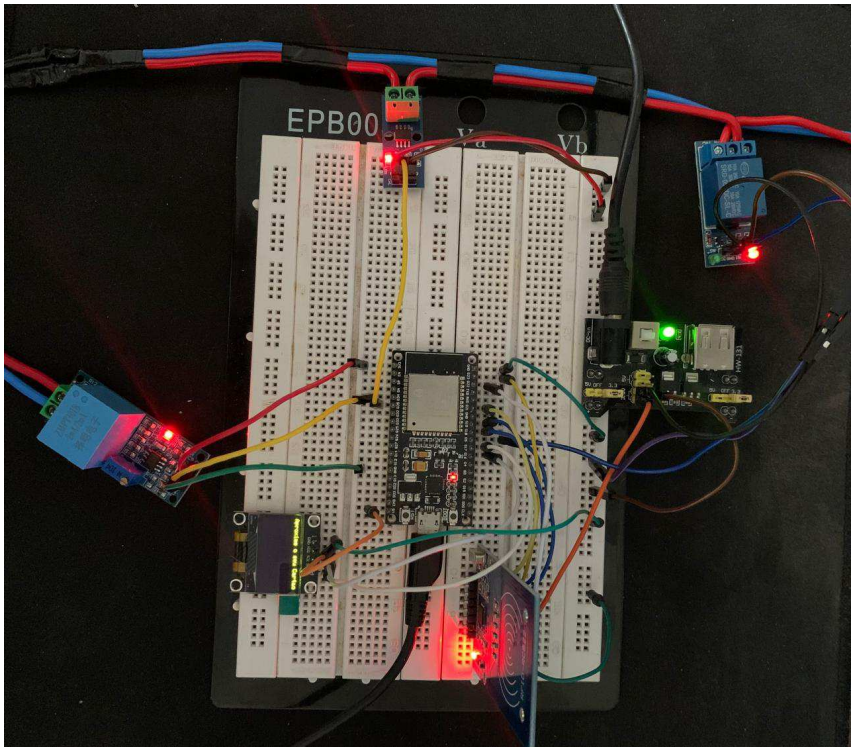
```
{
    Firebase.setString(defaultPath + "morador", morador);
    Firebase.setString(defaultPath + "UID_morador", content);
    Firebase.setFloat(defaultPath + "consumo_Kwh", totalEnergyConsumed);
    Firebase.setFloat(defaultPath + "custo", totalCost);
}
```

Exemplo 3 – *Exemplo de envio de dados para o DB*

3.2.6 Teste de integração

Logo após todos os testes serem realizados e todas ferramentas e ambientes estarem devidamente configurados, foi possível integrar todos os módulos à placa ESP32, criando um sistema em que todos os módulos conseguem se comunicar de forma sincronizada e eficiente, da mesma forma como se comportaram na fase de testes separados. Porém, agora em condições reais de operação. A montagem final do projeto se encontra na figura 35.

Figura 35 – Módulos integrados à placa ESP32.



Fonte: Autoria Própria

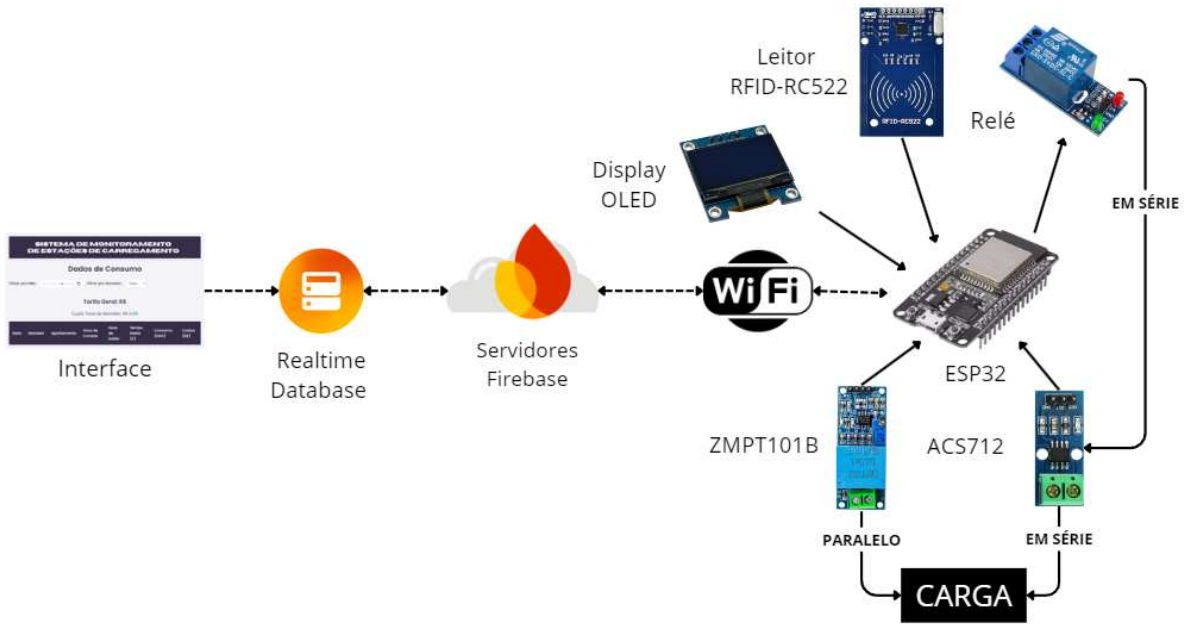
Além disso, foi adicionado uma tela OLED, que apresenta em tempo real todas informações necessárias para quem está acessando o sistema como usuário. Para que desse modo, a experiência do usuário seja voltada para transparência nos dados captados e equitativo para o propósito do projeto. Dessa forma, com a montagem devidamente concluída, foi factível fazer o *upload* do código, presente no Anexo A, para a ESP32, além de visualizar o comportamento da exibição do lado da interface com a inserção dos dados no DB.

4 Resultados e discussões

De modo geral, o projeto passou por duas fases importantes, uma primeira fase de pesquisa, anexando todo saber teórico que fundamentou o funcionamento e a proposta do projeto, e uma fase de desenvolvimento prático. Durante a fase de desenvolvimento, foram realizados testes que trouxeram como respaldo determinados meios e técnicas que mostraram a direção mais adequada para o projeto final. Sendo assim, é possível avaliar o resultado dos testes como necessários e eficazes no que se propuseram fazer.

Dois testes em específico, o de tensão e o de corrente, se mostraram serem passiveis de erros, que de certa forma são toleráveis, visto que estão lidando com sensoriamento de variáveis imprecisas e que estão a todo momento sofrendo interferências. De toda forma, cumpriram com a expectativa. A figura 36 apresenta a esquematização do fluxo de trabalho do projeto finalizado.

Figura 36 – Esquematização do fluxo de trabalho do projeto finalizado.



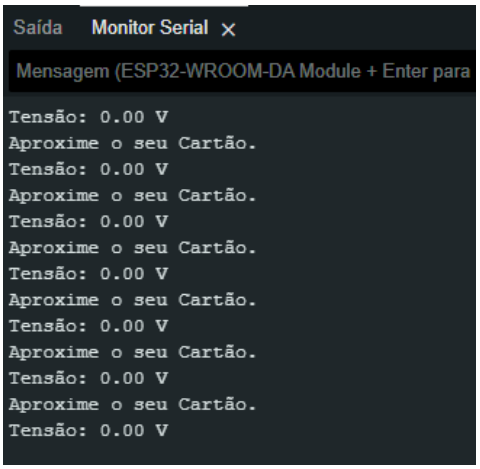
Fonte: Autoria Própria

O sistema possui dois estados específicos, o primeiro estado é o momento de espera pelo início de uma sessão de carregamento e a outra é exatamente o estado em que o sistema se encontra em uma sessão, capturando os dados de tensão e corrente e por conseguinte calculando todos os parâmetros necessários.

A diferença entre eles é definida pelo estado em que o relé se encontra. Por um

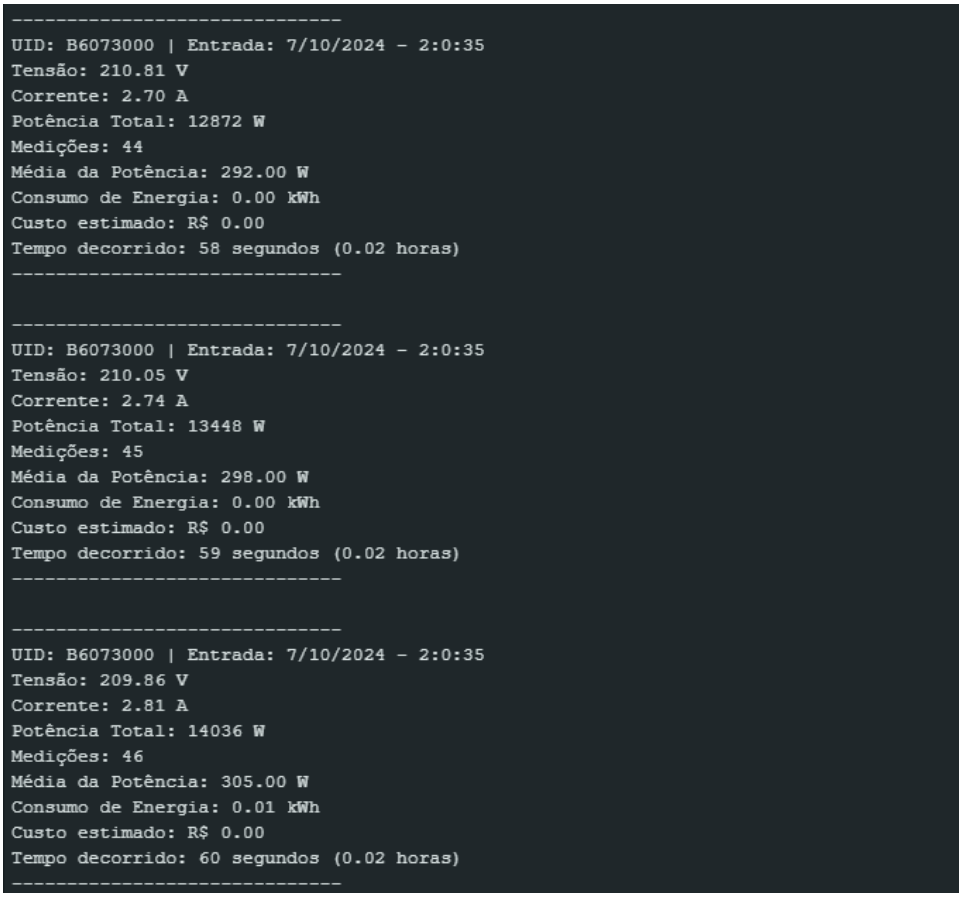
lado, se o relé estiver desligado, a tensão será zero como pode ser visto na figura 37. Por outro lado, se o relé estiver ligado, a sessão já foi iniciada e sistema está funcionando de acordo com os objetivos determinados anteriormente, assim como foi ser visto na figura 38.

Figura 37 – Captura da saída serial esperando pelo início da sessão.



Fonte: Autoria Própria

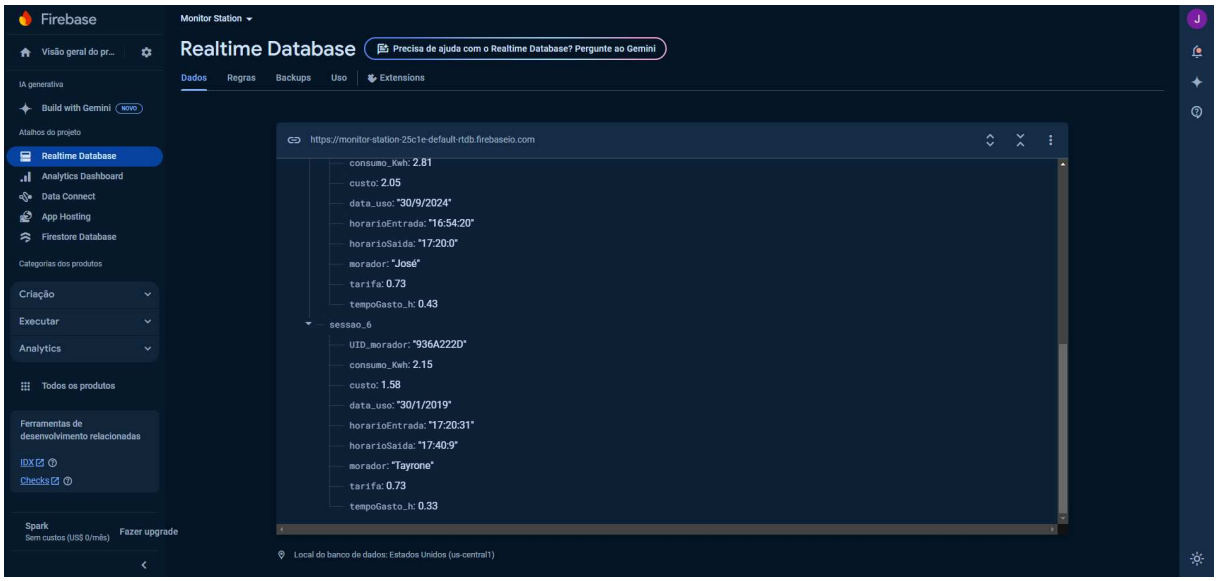
Figura 38 – Captura da saída serial durante sessão de carregamento.



Fonte: Autoria Própria

A estruturação do BD que se encontra na seção 3.2.5, mostrado na figura 39, foi mantida. Porém, mesmo que seja uma estrutura simples se mostrou ser eficaz e funcional conforme o objetivo do projeto, armazenando os dados de cada sessão e os dados referentes de cada morador do condomínio. Contudo, por mais que a estrutura "moradores" exista dentro do BD, não há uma forma uma forma fácil e intuitiva de cadastrar mais moradores (ou mais cartões RFID), a não ser interferindo diretamente no *Realtime Database* do *Firebase*.

Figura 39 – Captura de tela do Realtime Database com os dados das sessões armazenados.



Fonte: Autoria Própria

Em última análise, através da interface é possível visualizar todas as informações que o administrador do sistema precisa para tomar conhecimento e repassar os custos para cada morador, usuário da estação. A interface conta com informações relacionadas a data do uso da estação, quem a utilizou, em qual apartamento essa pessoa reside, o horário em que ela iniciou e encerrou o carregamento, o tempo gasto, a quantidade de energia consumida e os custos com isso.

Portanto, é uma interface que cumpre o seu papel de apresentar dados relevantes, visto que, quem a está utilizando possui a capacidade de filtrar o histórico das sessões com base no ano, mês e morador. Porém, assim como ocorre no BD, o cadastro de mais moradores para o sistema, ainda depende também de uma intervenção direta no arquivo HTML. A interface final pode ser visualizada através da figura 40.

Figura 40 – Captura de tela da interface web.

Fonte: Autoria Própria

Por fim, o código do projeto final está disponível na plataforma GitHub, acessível pelo seguinte link: https://github.com/thayroneo/Energy_Monitoring_with_ESP32.

5 Considerações finais

O objetivo principal deste trabalho foi desenvolver um sistema, integrando vários tipos de tecnologias e ferramentas, com o intuito de dar suporte aos condomínios na hora de fazer o levantamento de gastos com consumo de energia elétrica de estações de carregamento de EV. Embora, o sistema seja aplicável a inúmeras outras situações que envolvem o monitoramento de corrente e tensão, assim como a autenticação de usuários.

Se baseou em uma carga teórica relacionada a EV, assim como os modos de carregamento existentes e os requisitos e limitações para cada um deles. Para que assim, fosse possível entender como o projeto deveria ser construído, para se utilizar em uma possível aplicação em escala real, levando em consideração a rede monofásica de 220V encontrada na região Nordeste do Brasil, onde este trabalho foi desenvolvido.

Os materiais e as ferramentas foram bem escolhidos e cumpriram exatamente com a expectativa que lhes foi dada no início do trabalho.

Para possíveis trabalhos futuros, pode-se sugerir alguns complementos que tornarão o projeto mais completo, são eles:

- **Cadastramento simplificado** - Implementar, na interface web, uma aba de cadastro de mais moradores e de seus respectivos cartões.
- **Medições de Potências variadas** - Fazer com que o sistema seja capaz de medir potência tanto em estações monofásicas quanto trifásicas.
- **Geração de relatórios** - Implementar, do lado da interface, a possibilidade de gerar relatórios em PDF, com o intuito de compartilhar com os dados com os moradores.
- **Análise de dados** - Incluir ferramentas de análise dos dados capturados.
- **Atualização automática da tarifa** - Fazer com que a tarifa seja atualizada automaticamente de acordo com o mês e a bandeira tarifária vigente.

Referências Bibliográficas

ABNT. *NBR 5410:2004 - Instalações elétricas de baixa tensão*. Rio de Janeiro, 2004. Citado na página 8.

ANFAVEA. *Quais são os tipos de veículos eletrificados?* ANFAVEA, 2023. Acesso em: 04 out. 2024. Disponível em: <https://anfavea.com.br/site/quais-sao-os-tipos-de-veiculos-eletrificados/>. Citado na página 7.

Arduino. *Arduino IDE 1.x - Environment*. 2024. <https://docs.arduino.cc/software/ide-v1/tutorials/Environment/>. Acesso em: 5 out. 2024. Citado na página 19.

Associação Brasileira de Engenharia Automotiva. *Cartilha de Eletromobilidade*. Associação Brasileira de Engenharia Automotiva, 2023. Disponível em: <https://aea.org.br/inicio/wp-content/uploads/2023/12/cartilha_eletromobilidade.pdf>. Citado 4 vezes nas páginas 5, 6, 7 e 9.

BEZERRA, L. B. *Carros elétricos: princípios, tendências e efeitos da popularização no Brasil*. 53 f. p., 2021. Trabalho de Conclusão de Curso (Graduação em Ciência e Tecnologia) – Universidade Federal Rural do Semi-Arido, Caraúbas, 2021. Orientador: Prof. Dr. Rudson de Souza Lima. Disponível em: <https://repositorio.ufersa.edu.br/server/api/core/bitstreams/555ea247-9a6a-4f0f-8061-55783f26f66e/content>. Citado na página 6.

CAETANO, J. V. A. *Carregadores de Veículos Elétricos e Seu Impacto na Qualidade da Energia Elétrica*. 63 f. p., 2021. Trabalho de Conclusão de Curso (Graduação em Engenharia Eletrica) – Universidade Federal de Uberlandia, Uberlandia, 2021. Orientador: Paulo Henrique Oliveira Rezende. Disponível em: <https://repositorio.ufu.br/bitstream/123456789/32110/1/CarregadoresVeiculosEletricos.pdf>. Citado na página 8.

Canal VE. *Saiba por que os carros elétricos têm freio regenerativo KERS*. 2023. Acesso em: 04 out. 2024. Disponível em: <https://canalve.com.br/saiba-por-que-os-carros-eletricos-tem-freio-regenerativo-kers/>. Citado na página 5.

CHARGERS, D. *Charging modes*. 2024. Acesso em: 04 out. 2024. Disponível em: <https://deltrixchargers.com/about-emobility/charging-modes/>. Citado 4 vezes nas páginas 8, 9, 10 e 11.

CHAVES, A. F. et al. In: *3º Anuário Brasileiro de Mobilidade Elétrica*. Brasília: Plataforma Nacional de Mobilidade Elétrica, 2023. p. 16. Acesso em: 04 out. 2024. Disponível em: <https://pnme.org.br/wp-content/uploads/2023/12/3o-Anuario-Brasileiro-de-Mobilidade-Eletrica.pdf>. Citado na página 8.

ELECTROMAPS. *Modos de carregamento de veículos elétricos*. 2024. Acesso em: 04 out. 2024. Disponível em: <https://www.electromaps.com/pt/blog/modos-carregamento-veiculos-eletricos>. Citado na página 10.

Eletrogate. *Módulo Relé 1 Canal 5V*. 2024. Acesso em: 04 out. 2024. Disponível em: <https://www.eletrogate.com/modulo-rele-1-canal-5v>. Citado na página 15.

ENGINEERS, L. M. *ESP32 Basics: ADC (Analog to Digital Converter)*. 2024. Acesso em: 07 out. 2024. Disponível em: <https://lastminuteengineers.com/esp32-basics-adc/>. Citado na página 11.

ESPRESSIF. *ESP32-WROOM-32: Datasheet*. 2024. Acesso em: 04 out. 2024. Disponível em: <https://www.espressif.com/sites/default/files/documentation/esp32-wroom-da_datasheet_en.pdf>. Citado na página 13.

EV Connect. *4 Types of EV and How to Charge Them*. 2024. Acesso em: 04 out. 2024. Disponível em: <https://www.evconnect.com/blog/4-types-of-ev-and-how-to-charge-them>. Citado na página 4.

FERREIRA, A. G. L. et al. Veículos elétricos em redes de baixa tensão: impactos na qualidade da distribuição de energia. In: *IX Simpósio Brasileiro de Sistemas Elétricos - SBSE 2022*. Sociedade Brasileira de Automática (SBA), 2022. p. 631–639. ISSN: 2177-6164. Disponível em: <https://www.sba.org.br/open_journal_systems/index.php/sbse/article/view/2964/2495>. Citado na página 10.

FROMER, C. *Com vendas de junho, mercado bate em 80 mil eletrificados no 1º semestre e 300 mil em circulação*. 2024. <https://abve.org.br/80-mil-eletrificados-so-no-primeiro-semestre/>. Acesso em: 25 jul. 2024. Citado na página 1.

Google. *Firebase Realtime Database Documentation*. 2024. Acessado em: 05 out. 2024. Disponível em: <https://firebase.google.com/docs/database?hl=pt-br>. Citado na página 20.

HERCOG, D. et al. Design and implementation of esp32-based iot devices. *Sensors*, v. 23, n. 15, 2023. ISSN 1424-8220. Disponível em: <https://www.mdpi.com/1424-8220/23/15/6739>. Citado na página 13.

INSTIPER. *ESP32*. 2024. Acesso em: 04 out. 2024. Disponível em: <https://robotics.instiperjogja.ac.id/post/ESP32>. Citado na página 13.

International Energy Agency. *Global EV Outlook 2024*. Paris: IEA, 2024. <https://www.iea.org/reports/global-ev-outlook-2024>. Licence: CC BY 4.0, p. 17. Citado na página 1.

IPEC Carregadores. *Carregador Portátil para Carro Elétrico*. 2024. Acesso em: 04 out. 2024. Disponível em: <https://www.ipecarregadores.com.br/carregador-portatil-carro-eletrico>. Citado na página 9.

JORDAO, E.; KAKUTA, A. *Oito meses que mudaram o mercado de eletromobilidade*. 2023. Acesso em: 01 out. 2024. Disponível em: <https://abve.org.br/oito-meses-que-mudaram-o-mercado-de-eletromobilidade/>. Citado na página 4.

KURUNDKAR, S. et al. Advance security system. In: . [S.l.: s.n.], 2021. p. 1–4. Citado na página 18.

LI, L. et al. The application of hall sensors acs712 in the protection circuit of controller for humanoid robots. In: *2010 International Conference on Computer Application and System Modeling (ICCASM 2010)*. [S.l.: s.n.], 2010. v. 12, p. V12–101–V12–103. Citado na página 17.

LISTON, R. J. *Desenvolvimeno de um dispositivo para testes de carregadores para veiculos eletricos*. 79 f. p., 2024. Trabalho de Conclusão de Curso (Graduação em Engenharia Eletrica) – Universidade Federal de Santa Catarina, Florianopolis, 2024. Orientador: Mateus Nava Mezaroba, Msc. Disponível em: <https://repositorio.ufsc.br/handle/123456789/256980>. Citado 3 vezes nas páginas 8, 9 e 10.

MakerHero. *Guia Completo sobre o Relé: O Que é e Como Utilizar em Projetos*. 2024. Acesso em: 04 out. 2024. Disponível em: <https://www.makerhero.com/guia/componentes-eletronicos/rele/>. Citado na página 14.

MasterWalker Shop. *Sensor de Corrente AC e DC - ACS712 20A*. 2024. Acesso em: 04 out. 2024. Disponível em: <https://www.masterwalkershop.com.br/sensor-de-corrente-ac-e-dc-acs712-20a>. Citado na página 17.

Mckinsey & Company. *O futuro da mobilidade no Brasil*. 2023. Acesso em: 01 out. 2024. Disponível em: <https://www.mckinsey.com.br/our-insights/all-insights/o-futuro-da-mobilidade-no-brasil>. Citado na página 1.

Microsoft. *Visual Studio Code Documentation*. 2024. Acessado em: 05 out. 2024. Disponível em: <https://code.visualstudio.com/docs>. Citado na página 21.

NEOCHARGE. *Glossário*. 2023. Acesso em: 04 out. 2024. Disponível em: <https://www.neocharge.com.br/tudo-sobre/carro-eletrico/glossario>. Citado na página 8.

NeoCharge. *Tipos de Veículos Elétricos*. 2024. Acesso em: 04 out. 2024. Disponível em: <https://www.neocharge.com.br/tudo-sobre/carro-eletrico/tipos-veiculos-eletricos>. Citado na página 4.

NITI Aayog. *Types of Electric Vehicles*. 2024. E-AMRIT – Accelerated e-Mobility Revolution for India's Transportation. Acesso em: 04 out. 2024. Disponível em: <https://e-amrit.niti.gov.in/types-of-electric-vehicles>. Citado na página 6.

OZORIO, L.; OTHERS. Perspectivas para o mercado brasileiro de veículos elétricos. *Agência CanalEnergia*, 2021. Citado na página 1.

PROESI. *Display LCD/OLED I2C 0.96 Branco I2C*. 2024. Acessado em: 22 out. 2024. Disponível em: <https://www.proesi.com.br/display-lcd-oled-i2c-0-96-branco-i2c>. Citado na página 19.

Proesi. *Módulo Sensor de Tensão AC 0 a 250VAC ZMPT101B*. 2024. Acesso em: 04 out. 2024. Disponível em: <https://www.proesi.com.br/modulo-sensor-de-tens-o-ac-0-a-250vac-zmpt101b>. Citado na página 16.

Rytronics. *RFID Reader Writer RC522 SPI S50 with RFID Card and Tag*. 2024. Acesso em: 04 out. 2024. Disponível em: <https://www.rytronics.in/product/rfid-reader-writer-rc522-spi-s50-with-rfid-card-and-tag/>. Citado na página 18.

SILíCIO, V. de. *ESP32 - ESP-WROOM-32 NodeMCU*. 2024. Acesso em: 06 out. 2024. Disponível em: <https://www.vidadesilicio.com.br/produto/esp32-esp-wroom-32-nodemcu/>. Nenhuma citação no texto.

Smart Componentes. *Módulo Sensor de Tensão AC ZMPT101B 0-250V Voltímetro Arduino*. 2024. Acesso em: 04 out. 2024. Disponível em: <https://www.smartcomponentes.com.br/MLB-2050936045-modulo-sensor-tenso-ac-zmpt101b-0-250v-voltimetro-arduino-_JM>. Citado na página 16.

SparkFun Electronics. *ACS712 Low Current Sensor Hookup Guide*. 2023. Acesso em: 05 out. 2024. Disponível em: <https://learn.sparkfun.com/tutorials/acs712-low-current-sensor-hookup-guide/all?print=1>. Citado na página 17.

STA Eletrônica. *Como utilizar o módulo relé com Arduino*. 2024. Acesso em: 04 out. 2024. Disponível em: <https://www.sta-eletronica.com.br/artigos/arduinos/como-utilizar-o-modulo-rele-com-arduino>. Citado na página 15.

ANEXO A – Código Implementado

Neste apêndice, é apresentado o código base utilizado para o funcionamento do projeto.

```
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>
#include <ZMPT101B.h>
#include <SPI.h>
#include <MFRC522.h>
#include <IOXhop_FirebaseESP32.h>
#include <ArduinoJson.h>
#include <WiFi.h>
#include <time.h>

#define SCREEN_WIDTH 128
#define SCREEN_HEIGHT 64
#define OLED_RESET -1

// Definindo pinos do OLED
#define I2C_SDA 17
#define I2C_SCL 16
Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, OLED_RESET);
    //Insancia objeto display

// Definindo pinos do RFID, rel  e do Sensor de corrente
#define CURRENT_PIN 35          // Pino ADC conectado ao OUT do ACS712-20A
#define SS_PIN 21               // SDA do RFID
#define RST_PIN 22              // RST do RFID
#define RELAY_PIN 5             // Pino onde o rel  est  conectado
MFRC522 rfid(SS_PIN, RST_PIN);  // Instancia o objeto rfid

// Sensor de tens o ZMPT101B
#define SENSITIVITY 785.75f
ZMPT101B voltageSensor(34, 60.0);  // (Pino, Frequencia da Rede)

#define FIREBASE_HOST ""  // Link do DB
#define FIREBASE_AUTH ""  // Chave de Autenica  o do DB

// Nome e senha do WiFi
const char* wifi_ssid = "";
const char* wifi_pass = "";

// UID dos cart es autorizados
```

```
String card1 = "B6073000";
String card2 = "936A222D";

bool relayState = false;  // Estado do rel  (true para aberto e false para fechado)

// Definida conforme tarifa na cidade de Campina Grande do m s 09/2024
float energyCostPerKwh = 0.731980;  // Tarifa de energia em R$/kWh

// Vari veis
unsigned long relayStartTime = 0;      // Armazena o tempo de in cio do rel
unsigned long elapsedTime = 0;         //Tempo da sessao em segundos
float elapsedHours = 0.0;              //Tempo da sessao em horas
float averagePower = 0.0;              // Poencia media
float totalEnergyConsumed = 0.0;       // Energia acumulada em kWh
unsigned long totalPowerSum = 0.0;     // Soma das potencias para o c lculo da m dia
unsigned long powerMeasurements = 0;  // N mero de medi  es realizadas
float totalCost = 0.0;                 // Custo total
String content = "";
int countSessionsID = 0;               //Individualizar cada dessao em ordem numerica

// Vari veis para armazenar os dois hor rios
String entryDate_DMY, exitDate_DMY;
String entryTime_HMS, exitTime_HMS;
int entryDay, entryMonth, entryYear, exitDay, exitMonth, exitYear;
time_t entryTime_stamp, exitTime_stamp;  // Para realizar a diferen a entre os hor rios

void setup() {
  Serial.begin(115200);

  // Inicialize o display
  Wire.begin(I2C_SDA, I2C_SCL);
  if (!display.begin(SSD1306_SWITCHCAPVCC, 0x3C)) {
    Serial.println(F("SSD1306 allocation failed"));
    for (;;)
      ; // N o prossiga, loop infinito
  }

  display.clearDisplay();
  display.setTextSize(1);
  display.setTextColor(SSD1306_WHITE);
  display.setCursor(0, 0);
  display.display();
```

```
82
83    // Inicializa o sensor de tens o
84    voltageSensor.setSensitivity(SENSITIVITY);
85
86    // Inicializa a comunica  o SPI para o RFID
87    SPI.begin();
88    rfid.PCD_Init();
89
90    //Inicialia a conex o com o WiFi
91    WiFi.begin(wifi_ssid, wifi_pass);
92    int i = 0;
93    while (WiFi.status() != WL_CONNECTED) {
94      delay(500);
95      Serial.print(".");
96      if (i > 20) {
97        Serial.println("Conex o WiFi n o foi bem-sucedida.");
98        ESP.restart();
99      }
100     i++;
101   }
102   Serial.println("WiFi conectado");
103
104   Firebase.begin(FIREBASE_HOST, FIREBASE_AUTH);  // Inicializa o BD
105
106   // Configura o NTP
107   configTime(-10800, 0, "pool.ntp.org");  // (Fuso hor rio do Brasil,
        Hor rio de Verao, Servidor NTP)
108
109   // Configura o pino do rele como sa da e garante que ele inicie
        desligado
110   pinMode(RELAY_PIN, OUTPUT);
111   digitalWrite(RELAY_PIN, HIGH);
112   relayState = false;
113
114   zeroOffset = analogRead(CURRENT_PIN) * (3.3 / 4095);  // Valor de tens o
        de referencia
115   Serial.println("-> Offset: " + String(zeroOffset));
116 }
117
118 void loop() {
119
120   // Vari vel para armazenar o UID do cart o que ativou o sistema
121   static String activeCardUID = "";  // Mant m o UID do cart o que ativou
        o rel
122
123   // Verifica se h  um novo cart o RFID
124   if (rfid.PICC_IsNewCardPresent() && rfid.PICC_ReadCardSerial()) {
```

```
125
126     // Obt m o UID do cart o lido
127     content = "";
128     // Trecho retirado da biblioteca
129     for (byte i = 0; i < rfid.uid.size; i++) {
130       content += String(rfid.uid.uidByte[i] < 0x10 ? "0" : "");
131       content += String(rfid.uid.uidByte[i], HEX);
132     }
133     content.toUpperCase();  // Converte o UID para mai sculas
134
135     // Verifica se o cart o    autorizado
136     if (content == card1 || content == card2) {
137
138       // Se o rel  estiver desligado (sistema inativo)
139       if (!relayState) {
140
141         // Armazena o UID do cart o que ativou o rel
142         activeCardUID = content;
143         countSessionsID++;
144
145         Serial.println("Cart o autorizado! Ativando o rel  ...");
146         display.clearDisplay();
147         display.print(F("Cartao autorizado!"));
148         display.display();
149         delay(1000);
150
151         // Ativa o rel  -> Atualiza o estado do rel  para Fechado
152         digitalWrite(RELAY_PIN, HIGH);
153         relayState = true;
154
155         // Registra o dia e a hora de entrada
156         getTimeInfo(entryDate_DMY, entryTime_HMS, entryDay, entryMonth,
    entryYear, entryTime_stamp);
157         Serial.println("Entrada: " + String(entryDate_DMY) + " | " + String
    (entryTime_HMS));
158
159         // Registra o tempo de in cio
160         relayStartTime = millis();  // S  atualiza o tempo se o rel
    estava desligado
161         Serial.println("Rel  ativado.");
162
163       } else if (content == activeCardUID) {  // Verifica se    o mesmo
    cart o que ativou o rel
164
165         // Caso o rel  j  esteja ligado e o mesmo cart o que ativou
    tentar desativar
166         Serial.println("Cart o autorizado!");
```

```
167         Serial.println("Desativando o rel  ...");
168
169         display.clearDisplay();
170         display.setCursor(0, 0);
171         display.print(F("Cartao autorizado!"));
172         display.setCursor(0, 10);
173         display.print(F("Encerrando sessao..."));
174         display.display();
175         delay(1000);
176
177         getTimeInfo(exitDate_DMY, exitTime_HMS, exitDay, exitMonth,
    exitYear, exitTime_stamp);
178         double spentTime = difftime(exitTime_stamp, entryTime_stamp);
179         float spentTime_h = spentTime / 3600;
180
181         Serial.println("Sa da: " + String(exitTime_HMS));
182         Serial.println("Tempo gasto: " + String(spentTime) + " Segundos ("
    + String(spentTime_h) + " Horas)");
183
184         sendDataToFirebase(countSessionsID, content, totalEnergyConsumed,
    totalCost, entryDate_DMY, entryTime_HMS, exitTime_HMS, energyCostPerKwh,
     spentTime);
185
186         // Desativa o rel  -> Atualiza o estado do rel  para Aberto
187         digitalWrite(RELAY_PIN, LOW);
188         relayState = false;
189
190         activeCardUID = "";  // Limpa o UID do cart o ativo
191         Serial.println("Rel  desativado.");
192
193         powerMeasurements = 0;
194         totalPowerSum = 0;
195
196       } else {
197         // Caso o cart o n o seja o mesmo que ativou o rel , o sistema
    rejeita a desativa  o
198         Serial.println("Cart o n o autorizado para desativar o sistema!")
    ;
199         display.clearDisplay();
200         display.print(F("Nao autorizado para desligar."));
201         display.display();
202         delay(1000);
203       }
204
205     } else {
206       // Cart o n o autorizado
207       Serial.println("Cart o n o autorizado!");
```

```
208         display.clearDisplay();
209         display.setCursor(0, 0);
210         display.print(F("Cartao nao autorizado!"));
211         display.display();
212         delay(1000);
213       }
214
215       // Para o processamento do cart o
216       rfid.PICC_HaltA();
217     }
218
219     // Se o rel  estiver ativado, monitora a tens o
220     if (relayState) {
221       float voltage = voltageSensor.getRmsVoltage();
222       if (voltage < 198) {
223         voltage = 0;
224       }
225       calculateAndPrintCost(voltage);
226
227     } else {
228       Serial.println("Aproxime o seu Cart o.");
229       display.clearDisplay();
230       display.setCursor(0, 0);
231       display.print(F("Aproxime o seu Cartao"));
232       display.display();
233
234       float voltage = voltageSensor.getRmsVoltage(); //Garante tens o zero
        quando o rele esta desligado
235       if (voltage < 198) {
236         voltage = 0;
237       }
238       Serial.println("Tens o: " + String(voltage) + " V");
239     }
240
241     delay(1000);
242 }
243
244 // Fun  o para calcular o consumo de energia e o custo ao longo do tempo
245 void calculateAndPrintCost(float voltage) {
246
247     // Limpa o display antes de atualizar os dados
248     display.clearDisplay();
249
250     float current = calculateRMSCurrent();
251
252     if (voltage == 0) {
253       averagePower = 0.0;
```

```
254   } else {
255     float power = voltage * current;  // Pot ncia em watts
256     totalPowerSum += power;           // Acumula a pot ncia total
257     powerMeasurements++;              // Incrementa o n mero de medi  es
258
259     // Calcula a pot ncia m dia
260     averagePower = totalPowerSum / powerMeasurements;
261   }
262
263   // Calcula o tempo decorrido em segundos e horas
264   unsigned long elapsedTime = (millis() - relayStartTime) / 1000;  //
       Milisegundos para Segunos
265   float elapsedHours = elapsedTime / 3600.0;                        //
       Converte tempo em segundos para horas
266
267   // Calcula a energia consumida em kWh
268   float energyConsumed = (averagePower / 1000.0) * elapsedHours;  //
       Energia consumida em kWh
269   totalEnergyConsumed = energyConsumed;                           //
       Acumula a energia total consumida
270   totalCost = totalEnergyConsumed * energyCostPerKwh;             //
       Calcula o custo total
271
272   // Exibe os valores no Monitor Serial
273   Serial.println("-----------------------------------");
274   Serial.println("UID: " + String(content) + " | Entrada: " + String(
       entryDate_DMY) + " - " + String(entryTime_HMS));
275   Serial.println("Tens o: " + String(voltage) + " V");
276   Serial.println("Corrente: " + String(current) + " A");
277   Serial.println("Pot ncia Total: " + String(totalPowerSum) + " W");
278   Serial.println("Medi  es: " + String(powerMeasurements));
279   Serial.println("M dia da Pot ncia: " + String(averagePower) + " W");
280   Serial.println("Consumo de Energia: " + String(totalEnergyConsumed) + "
       kWh");
281   Serial.println("Custo estimado: R$ " + String(totalCost));
282   Serial.println("Tempo decorrido: " + String(elapsedTime) + " segundos ("
       + String(elapsedHours) + " horas)");
283   Serial.println("-----------------------------------");
284   Serial.println(" ");
285
286   // Exibir no display
287   display.setCursor(0, 0);
288   display.print(F("Tensao: "));
289   display.print(voltage);
290   display.print(F(" V"));
291
292   display.setCursor(0, 10);
```

```
293   display.print(F("Corrente: "));
294   display.print(current);
295   display.print(F(" A"));
296
297   display.setCursor(0, 20);
298   display.print(F("Energia: "));
299   display.print(totalEnergyConsumed);
300   display.print(F(" KWh"));
301
302   display.setCursor(0, 30);
303   display.print(F("Pot Med: "));
304   display.print(averagePower);
305   display.print(F(" W"));
306
307   display.setCursor(0, 40);
308   display.print(F("Custo: R$ "));
309   display.print(totalCost);
310
311   display.setCursor(0, 50);
312   display.print(F("Tempo: "));
313   display.print(elapsedTime);
314   display.print(F(" s ("));
315   display.print(elapsedHours);
316   display.print(F(" h)"));
317
318   // Atualiza o display
319   display.display();
320 }
321
322 // Fun  o para enviar os dados para o BD
323 void sendDataToFirebase(int countSessionsID, String content, float
      totalEnergyConsumed, float totalCost, String entryDateString, String
      entryTimeString, String exitTimeString, float energyCostPerKwh, double
      spentTime) {
324
325   String defaultPath = "/sessions/sessao_" + String(countSessionsID) + "/";
326   String morador = Firebase.getString("/moradores/" + String(content) + "/
      nome");
327
328   // Obtem horario em horas
329   float spentTime_h = spentTime / 3600;
330
331   Firebase.setString(defaultPath + "morador", morador);
332   Firebase.setString(defaultPath + "UID_morador", content);
333   Firebase.setFloat(defaultPath + "consumo_Kwh", totalEnergyConsumed);
334   Firebase.setFloat(defaultPath + "custo", totalCost);
335   Firebase.setString(defaultPath + "data_uso", entryDateString);
```

```
336   Firebase.setString(defaultPath + "horarioEntrada", entryTimeString);
337   Firebase.setString(defaultPath + "horarioSaida", exitTimeString);
338   Firebase.setFloat(defaultPath + "tarifa", energyCostPerKwh);
339   Firebase.setFloat(defaultPath + "tempoGasto_h", spentTime_h);
340 }
341
342 //Funcao para obter horario e data
343 void getTimeInfo(String& dateFormat_DMY, String& time_HMS, int& day, int&
      month, int& year, time_t& nowTime) {
344
345   time_t rawtime;
346   time(&rawtime);
347   struct tm timeInfo;
348   localtime_r(&rawtime, &timeInfo);
349
350   dateFormat_DMY = String(timeInfo.tm_mday) + "/" + String(timeInfo.tm_mon
       + 1) + "/" + String(timeInfo.tm_year + 1900);
351   time_HMS = String(timeInfo.tm_hour) + ":" + String(timeInfo.tm_min) + ":"
        + String(timeInfo.tm_sec);
352
353   day = timeInfo.tm_mday;
354   month = (timeInfo.tm_mon + 1);
355   year = (timeInfo.tm_year + 1900);
356
357   nowTime = rawtime;
358 }
359
360 //Funcao para calcular corrente RMS
361 float calculateRMSCurrent() {
362
363   float current = 0.0;
364   float sumCurrentSquares = 0.0;
365   float sensitivity = 100.0;  // Sensibiblidade do ACS712-20A (100 [mV/A])
366   float analogValue = 0.0;
367   float analogVoltValue = 0.0;
368   const int numCurrentMeasurements = 1000;
369
370   for (int i = 0; i < numCurrentMeasurements; i++) {
371
372     analogValue = analogRead(CURRENT_PIN);
373     analogVoltValue = analogValue * (3.3 / 4095);
374     analogVoltValue = analogVoltValue - zeroOffset;
375     current = analogVoltValue / (sensitivity / 1000);
376     sumCurrentSquares += current * current;
377
378     delayMicroseconds(167);  // Periodo da onda (1/60Hz) em milissegundos
379   }
```

```
380
381   float RMSCurrent = sqrt(sumCurrentSquares / numCurrentMeasurements);
382   return RMSCurrent;
383 }
```

Codigos/tcc_codigo.ino