UNIVERSIDADE FEDERAL DOS VALES DO JEQUITINHONHA E MUCURI
Sistemas de Informação
Eduardo Augusto Balsamão Campos

DESENVOLVIMENTO DE UM DISPOSITIVO PARA CONTROLE E MONITORAMENTO DO CONSUMO ENERGÉTICO DE APARELHOS ELÉTRICOS DOMÉSTICOS

Diamantina
2023

Eduardo Augusto Balsamão Campos

DESENVOLVIMENTO DE UM DISPOSITIVO PARA CONTROLE E MONITORAMENTO DO CONSUMO ENERGÉTICO DE APARELHOS ELÉTRICOS DOMÉSTICOS

Trabalho de Conclusão de Curso apresentado a banca avaliadora do curso de Sistemas de Informação, como parte dos requisitos necessários à obtenção do título de bacharel em sistemas de informação.

Orientador: Rafael Santin

Diamantina
2023

MINISTÉRIO DA EDUCAÇÃO
UNIVERSIDADE FEDERAL DOS VALES DO JEQUITINHONHA E MUCURI

FOLHA DE APROVAÇÃO

Eduardo Augusto Balsamão Campos

DESENVOLVIMENTO DE UM DISPOSITIVO PARA CONTROLE E MONITORAMENTO DO CONSUMO ENERGÉTICO DE APARELHOS ELÉTRICOS DOMÉSTICOS

Trabalho de Conclusão de Curso apresentado ao Departamento de Computação como requisito parcial para obtenção do grau de Bacharel em Sistemas de Informação pela Universidade Federal dos Vales do Jequitinhonha e Mucuri.

Aprovada em 08/02/2023

BANCA EXAMINADORA

Prof Rafael Santin (orientador)
Faculdade de Ciências Exatas - UFVJM

Prof Marcelo Ferreira Rego
Faculdade de Ciências Exatas - UFVJM

Profª Cinthya Rocha Tameirão
Faculdade de Ciências Exatas - UFVJM

Documento assinado eletronicamente por **Rafael Santin**, **Servidor (a)**, em 14/02/2023, às 17:03, conforme horário oficial de Brasília, com fundamento no art. 6º, § 1º, do Decreto nº 8.539, de 8 de outubro de 2015.

Documento assinado eletronicamente por **Marcelo Ferreira Rego**, **Servidor (a)**, em 14/02/2023, às 17:15, conforme horário oficial de Brasília, com fundamento no art. 6º, § 1º, do Decreto nº 8.539, de 8 de outubro de 2015.

Documento assinado eletronicamente por **Cinthya Rocha Tameirão**, **Servidor (a)**, em 14/02/2023, às 17:56, conforme horário oficial de Brasília, com fundamento no art. 6º, § 1º, do Decreto nº 8.539, de 8 de outubro de 2015.

A autenticidade deste documento pode ser conferida no site https://sei.ufvjm.edu.br/sei/controlador_externo.php?acao=documento_conferir&id_orgao_acesso_externo=0, informando o código verificador **0978881** e o código CRC **78A49560**.

Referência: Processo nº 23086.000729/2023-83 SEI nº 0978881

Dedicatória

Dedico este projeto à minha família e amigos que sempre estiveram presentes em todos os momentos de minha formação. Dedico também aos professores que foram fudamentais para a construção da minha vida profissional.

Resumo

Cada vez mais, nota-se o crescimento dos dispositivos domésticos conectados à Internet e consequentemente o número e a variedade de dados é grande. O presente trabalho apresenta um dispositivo que democratiza tais dados, trazendo sustentabilidade energética e o bem-estar do usuário. Utilizando conceitos de ***Internet of Things*** (IOT) o protótipo é capaz de medir as grandezas energéticas do aparelho conectado e fornece-los de forma fácil, através de um sistema WEB. O objetivo é que o usuário acompanhe o gasto energético de seus eletrodomésticos e possa ligà-lo ou desligà-lo a distância, além disso, será possível definir um limite diário de gasto que quando atingido desliga o aparelho. Graças ao banco de dados em nuvem é possível acionar a carga de qualquer lugar basta acesso à internet. Os testes realizados comprovaram a efetividade do dispositivo em monitorar e controlar o cusmo de energia, possibilitando o seu uso para a gestão energética de aparelhos domésticos.

Palavras-chave: Internet of Things, Sistema Web, Nuvem, Gestão energética, Automação residencial.

Abstract

Increasingly, it's noted the growth of home devices connected to the internet and consequently the number and variation of data is huge. This paper presents a device that democratizes this data, bringing energy sustainability and user health care. Using the concept of Internet of Things (IOT) the prototype is able to measure the energy magnitude of the connected device and provide them in an easier way, through a WEB system. The objective is that the user follows the energy consumption of your appliances and be able to turn them on or turn them off remotely, besides this it will be possible to define a daily limit of consumption that turns off the device when reached. Due to the cloud database it's possible to activate the charge from anywhere, it just needs to have access to the internet. The tests performed comproved the effectiveness of the device in monitoring and controlling the cost of energy, allowing your use to the energy management of home devices.

Keywords: Internet of Things, Web system, Cloud, Energy management, Home automation.

Lista de ilustrações

Lista de tabelas

Sumário

1 Introdução

O consumo de energia vem crescendo no Brasil nos últimos anos, a projeção segundo o Sistema ONS indica que entre 2019 e 2024 o consumo nas áreas residenciais e comerciais aumentem, respectivamente, 3.9% e 4,1% ao ano. (SISTEMA ONS, 2020).

O aumento do consumo de energia elétrica no país veio acompanhado do aumento na tarifação da energia, devido à crise hídrica. Por esse motivo, a Agência Nacional de Energia Elétrica (ANEEL) elevou o custo da energia acrescentando R$14,20 a cada 100 quilowatt hora, consumidos. Essa sobretaxa na conta de energia foi denominada de bandeira de escassez hídrica e ficou em vigência de setembro de 2021 a abril de 2022 (ANEEL, 2021).

Os sistemas inteligentes conectados à internet estão sendo cada vez mais utilizados por coletarem informações através de sensores e fornecerem meios para que o usuário interaja com esses objetos, podendo controlá-los remotamente (PEREIRA, 2017).

Assim, a proposta desse trabalho é empregar os benefícios da Internet das coisas (IoT) para o desenvolvimento de um dispositivo de monitoramento e controle do consumo energético dos aparelhos elétricos de uma edificação, proporcionando um consumo mais consciente por parte dos usuários e maior entendimento dos gastos por aparelho que contabilizados em um único valor na fatura de energia.

1.1 Objetivo

Propõe-se com esse trabalho a elaboração de um dispositivo de monitoramento elétrico e acionamento remoto utilizando conceitos de Internet das coisas, a visualização dos dados será feita através de um sistema web hospedado em nuvem, isso permite que o usuário interaja com o sistema por qualquer dispositivo conectado a internet como computadores, smartphones e tablets.

1.2 Objetivos Específicos

- Desenvolver um dispositivo que tenha uma interface entre os aparelhos elétricos e a rede elétrica e permita capturar informações de consumo.
- Desenvolver um sistema web que permita a interação humana com o dispositivo.
- Estabelecer uma comunicação enviando as informações do dispositivo proposto para um sistema web. O sistema web deve receber dados energéticos do aparelho e enviar comandos de acionamento para o dispositivo.
- Realizar testes de acionamento a distância e confirmar se os dados coletados condizem com os dados reais

2 Revisão Bibliográfica

O presente capítulo tem a finalidade de apresentar a revisão bibliográfica relacionada aos conceitos teóricos como o de automação residencial e também aos equipamentos e softwares essenciais para o entendimento do sistema proposto.

2.1 Automação Residencial

A automação residencial é a aplicação de sistemas integrados visando satisfazer as tarefas no meio doméstico como segurança, comunicação, gestão energética e conforto. Tais sistemas devem ter a capacidade de seguir comandos estabelecidos pelo usuário com a possibilidade de mudanças conforme seus interesses (MURATORI, 2016).

No Brasil o mercado de automação é recente e teve seu início nas indústrias. Segundo Silveira et al. (2015). O número de empresas que atuam no segmento tem aumentado nos últimos anos trazendo maior oferta e baixa nos preços, porém, ainda falta mão de obra qualificada tanto na produção quanto na instalação desses equipamentos.

Os sistemas inteligentes estão cada vez mais presentes nas residências devido à flexibilidade de suas funções. As tarefas realizadas por esses aparelhos reduzem o esforço humano, economizam energia e minimizam a preocupação com a segurança.

2.2 Internet Das Coisas

O termo Internet das coisas (do inglês *Internet of things* - IoT) teve origem em uma apresentação do pesquisador britânico Kevin Ashton em 1999. Essa tecnologia pode ser entendida como objetos que se conectam à internet a fim de coletar e compartilhar informações e interagir de maneira inteligente com pessoas ou com outros objetos, permitindo a otimização e até mesmo a automação de tarefas (PEREIRA; CARVALHO, 2017).

Segundo (SANTOS ET AL., 2016). A IoT é uma combinação de tecnologias unidas que possibilitam a interação entre o mundo físico e virtual, essas tecnologias são:

Tecnologias de Identificação: tecnologias que permitem identificar os objetos para conectá-los à Internet como RFID ou endereçamento IP.

Sensores/Atuadores: sensores captam a informação e encaminham esses dados para serem processados, já os atuadores manipulam o ambiente de acordo com comandos recebidos.

Comunicação: tecnologias que conectam os objetos entre si, WiFi e Bluetooth são alguns exemplos.

Computação: unidade de processamento que executa os algoritmos locais, utilizam-se das informações coletadas pelos sensores e controlam os atuadores.

Serviços: dizem respeito aos serviços que a IoT pode prover como por exemplo: Serviços de Identificação, que retornam dados como temperatura local, coordenadas ge-

ográficas e instante da coleta e Serviços de Inteligência que usam os dados para tomar decisões de modo conveniente para cada cenário.

Semântica: remete à habilidade de extrair conhecimento e fazer descobertas, a partir dos dados coletados e então prover determinado serviço com uso eficiente dos recursos existentes na IoT.

Esses objetos podem disponibilizar as informações coletadas para análise humana mas também podem ser usadas de forma automática para que os dispositivos tomem ações específicas com base nos dados coletados.

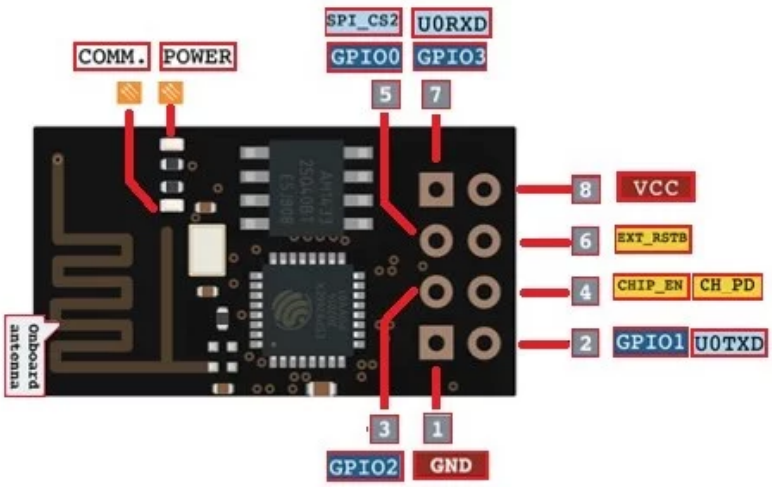
Em 2012 especialistas na área previram que levaria de 5 a 10 anos para que a tecnologia IOT fosse adotada pelo mercado e que em 2016 era vivenciado o maior pico de expectativas sobre a tecnologia até então (SANTOS ET AL., 2016).

2.3 ESP8266

O ESP8266 é uma série de microcontroladores com Wi-Fi integrado. Eles são compactos, baratos e energeticamente eficientes, tornando-os adequados para projetos IOT. Os módulos podem ser programados em linguagem LUA ou usando o ambiente de desenvolvimento do Arduino.

A série ESP8266 é produzida pela empresa chinesa Espressif Systems e possui várias placas disponíveis (ESP8266-01, ESP8266-05, ESP8266-07, ESP8266-201 e ESP8266-12E). O módulo ESP8266-01 é bastante empregado em projetos de IoT por possuir uma boa capacidade de processamento que supre as necessidades da maioria das aplicações e ser compacto. A Figura 1, mostra a referência dos pinos com suas devidas funções.

Figura 1 – Pinout da ESP8266-01



Fonte: Blog Eletrogate. Disponível em: <https://blog.eletrogate.com>

Os pinos 1 e 8 são respectivamente o aterramento e a fonte de energia. Já 2, 3, 5 e

7 são os GPIOs (em inglês *General Purpose Input/Output),* pinos programáveis de entrada e saída de dados. Os pinos 4 e 6 são os de controle sendo responsáveis pela reinicialização da placa por uma via externa e por ligar a placa respectivamente.

O chip está disponível no mercado desde 2014 e ainda existem poucas publicações acadêmicas sobre ele. A maioria dos trabalhos publicados são envolvendo internet das coisas. O trabalho de Garcia (2020) utiliza o modulo para desenvolver um telhado verde IoT autossustentável, que armazena água da chuva e utiliza de sensores de umidade para irrigar as plantas presentes no telhado.

O ESP8266 tem a capacidade de coletar dados de variados tipos de sensores, sua capacidade de conexão com internet possibilita enviar tais dados para um servidor hospedado em nuvem. Além disso, o chip consegue enviar sinais elétricos pelas GPIOs que servem como comandos para acionadores.

2.4 Monitoramento energético

Os sistemas de monitoramento e gerenciamento de energia permitem que os usuários entendam e controlem seu consumo de energia elétrica. Os dispositivos que medem energia podem ser divididos em duas classes diferentes: (DO NASCIMENTO, 2020)

- Eletromecânico: Os tradicionais relógios medidores presentes nas residências, que utilizam um disco de metal que se movem ao passar eletricidade nas bobinas.
- Eletrônico: Medidores digitais, compostos de sensores que capturam os dados energéticos.

Os medidores digitais estão presentes em artigos que abordam o monitoramento e a gestão de redes elétricas. Alguns trabalhos abordam sensores individuais para cada grandeza como é o caso de Andrade (2016) e Oliveira (2019) que usam sensores de corrente.

Andrade (2021) e Nascimento (2020) utilizaram o módulo PZEM (Figura 2), um medidor capaz de mensurar mais de uma gradeza de uma só vez. Produzidos pela chinesa Peacefair o sensor consegue medir as grandezas de tensão e corrente, variáveis cruciais para a gestão energética. Além disso, ele também calcula e fornece a frequência, a potência ativa e o fator de potência. O Anexo 1 contem uma tabela que mostra todos os dados que o módulo fornece.

Figura 2 – Módulo PZEM-004T

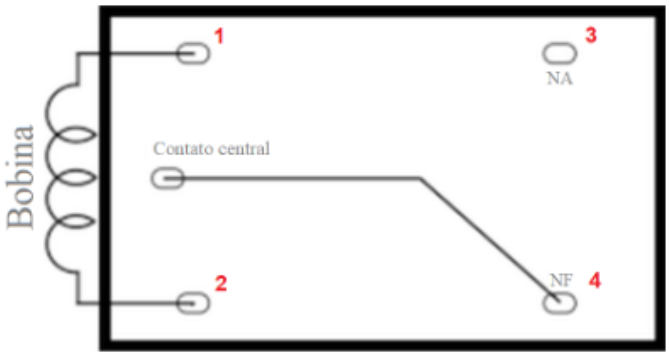
Fonte: Recicomp, Disponível em: <https://www.recicomp.com.br/produtos/modulo-wattimetro-voltimetro-amperimetro-100a-pzem-004t>

2.5 Relés

O relé é um componente elétrico que funciona como um interruptor, através da aplicação de uma corrente na bobina presente dentro do relé cria-se um campo magnético que atrai a alavanca responsável pela mudança de chaves. Isso permite que o módulo relé integre circuitos de alta e baixa tensão. (SANTOS 2019).

A Figura 3 mostra um exemplo do circuito interno. Os pinos da bobina (1 e 2) são alimentados com uma corrente de 5V criando um campo magnético que atrai a chave. O contato normalmente aberto (NA pino 3) é fechado permitindo a passagem de corrente, quando a corrente na bobina é interrompida a chave volta a sua posição original consequentemente interrompendo a passagem de energia.

Figura 3 – Circuito interno Relé.



Fonte: Mundo Projetado. Disponível em: <https://mundoprojetado.com.br/rele-o-que-e-e-como-funciona></https s:>

2.6 Arduino

O Arduino é uma plataforma de microcontroladores de código aberto com seu próprio ambiente de desenvolvimento. O Arduino foi criado com o intuito de ensinar Design de Interação, que se preocupa em criar experiencias significativas entre humanos e objetos. Utilizando conceitos de computação física esses microcontroladores podem criar objetos inteligentes que se comunicam com humanos através de sensores e atuadores com um comportamento predefinido por um software. (BANZI, 2011)

Um microcontrolador é um pequeno computador em uma placa de circuito integrado. Ele contém um processador, memória RAM para guardar os dados, memória flash para armazenar o software e os pinos de entrada e saída que ligam o microcontrolador aos sensores e atuadores. (MONK, 2013)

A IDE Arduino é um software de código aberto que foi projetado inicialmente para programar as placas Arduino, mas tem suporte para diversos outros tipos de microcontroladores, incluindo os da família ESP. A IDE conta com diversas bibliotecas produzidas pela comunidade que auxiliam muito na produção dos softwares.

Além disso, esse ambiente de desenvolvimento oferece um editor de código que exibe as informações sobre a compilação do programa, espaço de armazenamento na memória do sistema e outros estados do programa.

Para transferir um programa escrito no ambiente de desenvolvimento para o hardware, primeiro é necessário especificar o modelo da placa em questão e a porta serial do computador a qual o dispositivo foi inserido.

Por fim o programa conta um monitor serial onde é possível observar as saídas que o código produz com a função “Serial.print”, com isso é possível identificar os dados produzidos por sensores, a atuação dos controladores e os possíveis erros.

2.7 Banco de dados

O crescimento da Internet das Coisas (IoT) tem gerado um grande volume de dados a cada instante, exigindo uso do banco de dados para armazenar, gerenciar e acessar de forma eficiente os dados coletados pelos dispositivos IoT. Além disso, a constate atualização dos dados coletados necessitam que determinadas aplicações usem banco de dados em tempo de real para o gerenciamento eficiente dessas informações com altas taxas de atualizações.

Assim, a utilização de banco de dados em tempo real, como o Firebase, é bastante comum em aplicações da Internet das Coisas, permitindo que os desenvolvedores criem aplicativos IoT escaláveis.

O Firebase é uma plataforma criada pelo Google que oferece diversas ferramentas para produzir aplicativos de alta qualidade, seu objetivo é acelerar o desenvolvimento integrando, de maneira fácil e rápida, recursos baseados em nuvem para dispositivos móveis e web. (SILVA, 2018).

O Firebase fornece também ferramentas para a gestão de negócios baseados em aplicativos, como testes A/B que realizam previsões e análise de dados que promovem a compreensão sobre o uso do aplicativo e o envolvimento do usuário. A maioria dessas ferramentas são gratuitas possibilitando que um projeto comece a gerar receita sem gastos iniciais. (FIREBASE, 2022).

2.7.1 Realtime Database

O Realtime Database é um banco de dados NoSQL hospedado na nuvem e sincronizado em tempo real com os clientes conectados. Os dados são armazenados como uma árvore de objeto JSON e representado graficamente. (WSC SILVA, 2018).

Os dados armazenados são mantidos em um cache local e permite, mesmo offline, que alguns eventos em tempo real continuem sendo acionados, quando o cliente restabelecer a conexão as alterações locais são sincronizadas com atualizações remotas que ocorreram enquanto o cliente estava offline. (FIREBASE, 2022).

O Firebase proporciona um conjunto de regras baseadas em expressão, chamadas regras de segurança, que definem como os dados deverão ser acessados e por quem. Esse recurso funciona bem com o Authentication, pois assim é possível restringir a visualização dos dados apenas para usuários logados no sistema. (FIREBASE, 2022).

2.7.2 Authentication

O Firebase Authentication proporciona serviços de autenticação backend e bibliotecas de interface de usuário prontas e fáceis de usar. Ele permite que o desenvolvedor escolha o tipo de autenticação do seu aplicativo, como, por exemplo, email e senha, nú-

mero de celular ou provedores de identidade como Google, Facebook, Twitter entre outros. (FIREBASE, 2022).

A ferramenta ainda disponibiliza métodos para confirmação de usuário e redefinição de senha via e-mail que pode ser personalizado pelo desenvolvedor promovendo assim a padronização para marcas e produtos.

2.7.3 Hosting

O Firebase Hosting é uma ferramenta que oferece hospedagem rápida e segura para aplicativos web. Com ele é possível hospedar sites com conteúdos estáticos e dinâmicos sem precisar configurar um servidor complexo. O conteúdo é disponibilizado globalmente por meio de uma conexão SSL do servidor de borda mais próximo. (FIREBASE, 2022).

O Firebase garante a segurança fornecendo automaticamente certificados SSL para todos os domínios. Por fim o serviço disponibiliza subdomínios gratuitos nos domínios “web.app” e “firebase.com” porém é possível pagar para implementar um domínio próprio.

2.8 Potência Elétrica

Potência elétrica é uma medida física que determina quanto trabalho é realizado por uma unidade de tempo. Em circuitos elétricos determina a quantidade de energia que o circuito converte em outras formas de energia, também por unidade de tempo. Sua unidade de medida é o, Watts (W).

A informação da potência foi dada como importante para a visualização no sistema, pois ela define quanta energia o aparelho consome e qual é a sua capacidade. O seu cálculo pode ser realizado pela seguinte fórmula.

$$P = VI \tag{2.1}$$

P – Potência em watts (W)

V – Tensão em volts (V)

I – Corrente elétrica em amperes (A)

2.9 Energia Elétrica

Energia elétrica é a capacidade de trabalho de uma corrente elétrica. Ela pode ser gerada através de outras energias, por exemplo, usinas hidrelétricas que transformam energia mecânica em elétrica. Se tratando de consumo a unidade utilizada é o kilowatt hora (KWh), ela é usada para a tarifação da energia elétrica e seu cálculo é dado pela seguinte fórmula.

$$E = P\Delta t/1000 \tag{2.2}$$

E – Energia elétrica em quilowatt-hora (kWh)

P – Potência em watts (W)

Δt = Variação do tempo em horas (H)

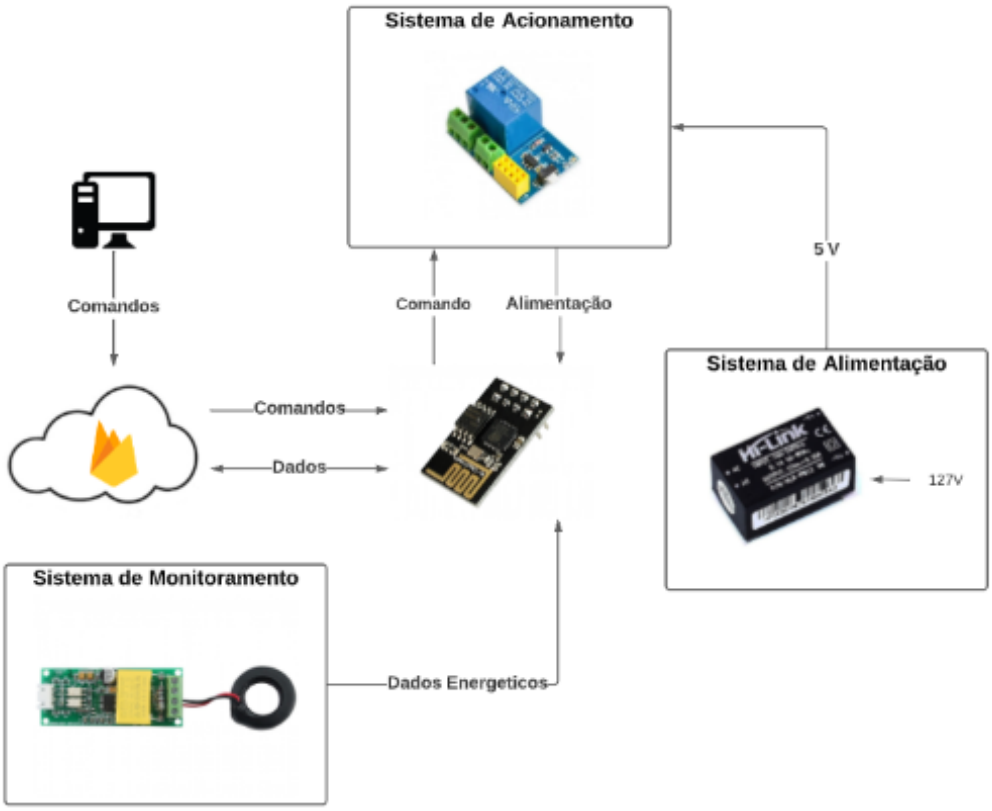
3 Materiais e Métodos

O objetivo deste capítulo é apresentar os procedimentos utilizados para atingir o objetivo do projeto. Será apresentada a arquitetura do sistema desenvolvido, explicando como é feita a aquisição e transmissão de dados. Além disso, é mostrado o desenvolvimento do sistema web, seu diagrama de casos de uso, decisões de design e a integração do microcontrolador com a aplicação web.

3.1 Arquitetura do Sistema

O sistema proposto é baseado na Internet das Coisas. Sendo composto por um sensor para medir as grandezas energéticas (P-ZEM), um microcontrolador (ESP8266) capaz de enviar os dados via internet e um aplicativo que permite a interação com os dados e o aparelho. A Figura 4 ilustra a arquitetura do sistema.

Figura 4 – Arquitetura do sistema



Fonte: Do autor

O sistema de alimentação recebe 127 volts e transforma em 5 volts para alimentar o modulo relé com a ESP8266. Para ligar o aparelho a distância o usuário dispara um comando pelo sistema web, esse comando é enviado para o banco de dados que depois é lido pelo microcontrolador que controla o relé. Os dados energéticos são lidos pelo sistema

de monitoramento, enviados para o banco de dados pela ESP8266 e lidos pelo usuário no site.

O sensor PZEM004 foi escolhido por ter uma interface aberta e bem documentada, o módulo é fácil de ser integrado aos circuitos. A leitura das medições é realizada pelo ESP8266 e pode ser feita analisando cada endereço de memória de forma binária, ou usando bibliotecas de livre acesso (ANDRADE, 2021).

Tanto o sensor quanto a ESP8266 são alimentados por uma fonte de 5V e se comunicam através da interface TLL (Transistor-Transistor Logic) (NASCIMENTO, 2020). Isso torna o PZEM-004 a escolha ideal para o projeto.

Para realizar os comandos de liga e desliga nos dispositivos de alta tensão presentes no projeto, foi utilizado um módulo relé projetado para o chip ESP8266-01. A decisão de usar este dispositivo foi tomada com objetivo de reduzir a complexidade do circuito final, uma vez que a corrente emitida pelo chip ESP é insuficiente para acionar um relé convencional por si só (ANDRADE, 2019).

O aplicativo web foi projetado para funcionar em computadores pessoais e mobile, e permite que o usuário monitore o gasto por dia e por mês do aparelho e gere relatórios em forma de gráficos e tabelas. Além disso, é possível interagir ligando e desligando a alimentação dos aparelhos conectados.

A rápida sincronização de informações promovida pelo Firebase é um ponto importante do sistema desenvolvido. Isso permite que múltiplos usuários comandem o mesmo aparelho simultaneamente, graças a isso foi possível criar um sistema de compartilhamento de dispositivos.

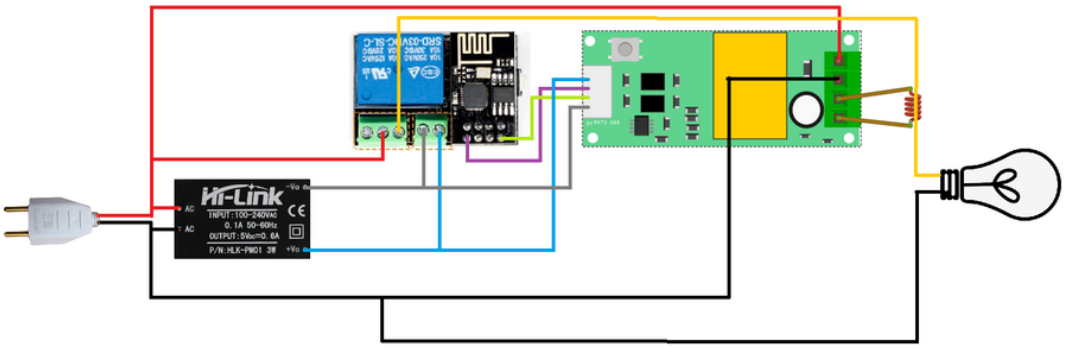
3.2 Montagem do Protótipo

O circuito projetado foi montado utilizando uma uma placa para desenvolvimento de protótipos (protoboard), e foi usada para coletar as primeiras amostras sobre o consumo energético. As partes principais que compõem o circuito são a fonte de 5V do tipo Hi-Link, o relé integrado ao microcontrolador ESP8266 e o medidor PZEM-004T.

O sensor precisa ser alimentado com uma tensão de referência, a fonte de 5V trabalha com corrente alternada em uma faixa de entrada parecida com a do medidor (100V a 240V para o HI-LINK e 80V a 260V para o PZEM-004T), logo os dois serão ligados a mesma fonte. O PZEM possui ainda duas entradas para o medidor de corrente não invasivo, uma bobina, que envolve a fase ou o neutro do cabo de energia ao qual o aparelho está ligado.

A saída do HI-LINK é uma corrente contínua de 5V suficiente para alimentar o módulo relé com a ESP8266 e o módulo medidor. A imagem abaixo mostra como foi montado o circuito.

Figura 5 – Circuito



Fonte: Do autor

O módulo medidor tem 4 conectores Rx, Tx, 5V e GND. Ele se comunica com o microcontrolador utilizando a interface TLL UART (Universal Asynchronous Receiver/Transmiter) de forma paralela, ou seja, o RX de um dispositivo se conecta ao TX de outro e vice-versa. (ANDRADE, 2021)

3.3 Programando o MCU

Para a programação da ESP8266 foi utilizado o IDE do Arduino com a linguagem de programação C++. Foram usadas bibliotecas externas para facilitar a implementação do algoritmo. Tais bibliotecas contêm as funções específicas para o projeto como, facilitar a conexão WiFi (Wifi Manager), retornar valores de data como ano, mês, dia, horas, minutos e segundos com um curto intervalo entre os envios (NTPClient), destinar os resultados para o banco de dados (FirebaseESP8266) e, além disso, uma biblioteca com métodos próprios para o medidor PZEM (PZEM004Tv30).

O programa funciona em duas etapas: o setup e o loop, antes disso são definidas as variáveis globais que serão usadas ao longo do código, como, por exemplo, as credenciais para a conexão com o banco de dados.

O Arduíno possui duas funções chamadas de setup e loop que não necessitam de chamada ao decorrer do programa. Quando o código é compilado, a função setup é chamada apenas uma vez. Apos a conclusão do código presente no setup a função loop é chamada repetidamente até que o microcontrolador seja reiniciado ou desligado. (CHAVIER, 2017)

A função setup, primeiramente, define qual será a taxa de transferência em bits por segundo (baud rate) para transmissão serial, depois são chamadas as funções iniciais das bibliotecas. O fluxo normal de execução do sistema busca estabelecer a conexão WiFi utilizando métodos da biblioteca WifiManager, caso a conexão se concretize são acionados os métodos que precisam de internet para funcionar, como é o caso do Firebase e do NTPClient.

Por último é estabelecido quais pinos do MCU serão de saída de dados (output), para a mudança de estado do relé, e quais serão os pinos de entrada de dados (input), para receber os dados vindos do PZEM. O Código 3.1 a seguir mostra como foi escrita a função setup.

Código 3.1 – Função setup

```
1 void setup()
2 {
3   Serial.begin(115200); //Inicia a Conexão Serial
4   WiFiManager wifiManager; // Inicia o WiFiManager
5   //wifiManager.resetSettings(); //Reseta as credenciais de WiFi
6   wifiManager.autoConnect("Grupo Connect"); //Inicia o
       autoConnect com o nome Grupo Conect
7
8   Firebase.begin(FIREBASE_HOST, FIREBASE_AUTH); //Inicia a conexã
       o com o Firebase
9   ntpClient.begin(); //Inicia a variavel ntp
10
11  //Tempo Médio de Greenwich brasileiro em segundo (-3 = -10800)
12  ntpClient.setTimeOffset(-10800);
13
14  pinMode(rele,OUTPUT); // Seta que o "rele" é de saida
15  pzem = PZEM004Tv30(pzemSWSerial); //Seta pinos do pzem
16 }
```

Na função loop é onde realmente o código funciona, nele são definidas quais funções serão realizadas de tempo em tempo. Esse bloco será dividido em 4 para facilitar o entendimento.

Bloco 1: Nesse bloco acontece a atualização das variáveis de tempo, para isso foi utilizado 4 métodos do ntpClient (update, getEpochTime, getHours e getDay) e uma struct nativa do C++ (struct tm). Tais variáveis serão responsáveis por definir os intervalos entre as capturas das informações energéticas. O Código 3.2 mostra quais variáveis foram capturadas.

Código 3.2 – Bloco 1 do loop

```
1 //Time updates
2   ntpClient.update();
3   unsigned long epochTime = ntpClient.getEpochTime();
4   struct tm *ptm = gmtime ((time_t *)&epochTime);
5   int hours = ntpClient.getHours();
6   int monthDay = ptm->tm_mday;
7   int currentYear = ptm->tm_year+1900;
8   String weekDay = weekDays[ntpClient.getDay()];
9   int currentMonth = ptm->tm_mon+1;
10  String currentMonthName = months[currentMonth-1];
```

Bloco 2: Neste bloco são lidas as informações colhidas pelo módulo medidor, para isso foi utilizado 5 métodos do PZEM cada um para ler um tipo de informação (voltage, current, power, frequency e pf), em seguida essas informações são armazenadas no banco de dados utilizando um método do Firebase (setFloat). No Código 3.3, as linhas 2 a 7

são responsáveis por capturar as informações do medidor e as linhas 8 a 11 enviam tais informações ao banco de dados.

Código 3.3 – Bloco 2 do loop

```
1   // Read the data from the sensor
2   float voltage = pzem.voltage();
3   float current = pzem.current();
4   float power = pzem.power();
5   float energy = pzem.energy();
6   float frequency = pzem.frequency();
7   float pf = pzem.pf();
8   Firebase.setFloat(firebaseData, "/dispositivos/gc002/energy/
       voltage", voltage);
9   Firebase.setFloat(firebaseData, "/dispositivos/gc002/energy/
       current", current);
10  Firebase.setFloat(firebaseData, "/dispositivos/gc002/energy/
       power", power);
11  Firebase.setFloat(firebaseData, "/dispositivos/gc002/energy/
       frequency", frequency);
12  Firebase.setFloat(firebaseData, "/dispositivos/gc002/energy/
       powerfactor", pf);
```

Bloco 3: Nesse bloco é tratado como o dado de consumo (energy) será armazenado no banco de dados. Aqui é definido que a leitura de consumo será armazenada de hora em hora. O dado então é escrito no banco de dados em uma sequência de ano, mês, dia e hora, redefinindo o dado de consumo toda vez que o dia muda, assim é possível visualizar quanto (em KW/H) o aparelho consome por dia. Para isso foram utilizados 5 métodos do Firebase (setString, setFloat, getInt, setInt e intData) e um método do PZEM (resetEnergy).

Código 3.4 – Bloco 3 do loop

```
1   Firebase.setString(firebaseData, "/dispositivos/gc002/energy_
       historic/" + String(currentYear) + "/" + currentMonthName +
       "/" + String(monthDay) + "/mes" , currentMonthName);
2   Firebase.setString(firebaseData, "/dispositivos/gc002/energy_
       historic/" + String(currentYear) + "/" + currentMonthName +
       "/" + String(monthDay) + "/dia" , weekDay);
3   Firebase.setFloat(firebaseData,
4      "/dispositivos/gc002/energy_historic/" + String(currentYear)
           + "/" + currentMonthName + "/" + String(monthDay) + "/
          energy/" + String(hours), energy);
5
6   if (Firebase.getInt(firebaseData, "/dispositivos/gc002/dia_
       sistema")) {  //On successful Read operation, function
       returns 1
7
8     dia_sistema = firebaseData.intData();
9     if(dia_sistema != monthDay){
10      pzem.resetEnergy();
11      Firebase.setInt(firebaseData, "/dispositivos/gc002/dia_
           sistema" , monthDay);
12    }
13
14  } else {
```

```
15      Serial.println(firebaseData.errorReason());
16    }
```

Bloco 4: Nesse bloco é realizado o acionamento do relé tanto manual quanto por definição de limite. É realizada a leitura do dado “status” no banco de dados, se o valor é zero o relé desliga o aparelho se for um o relé é ativado ligando o aparelho. Para isso foram utilizados 4 métodos do Firebase (getString, setString, stringData, dataTypeEnum).

Código 3.5 – Bloco 4 do loop

```
1  //Acionamento "manual"
2  if (Firebase.getString(firebaseData, "/dispositivos/gc002/status
       ")) {  //On successful Read operation, function returns 1
3    val = firebaseData.stringData();
4    if(val == "0"){
5      digitalWrite(rele, LOW); // Desliga o rele
6    }
7    if (val == "1"){
8      digitalWrite(rele, HIGH); // Liga o rele
9    }
10 } else {
11   Serial.println(firebaseData.errorReason());
12 }
13
14 //Acionamento "por limite"
15 if (Firebase.getString(firebaseData, "/dispositivos/gc002/energy_
     historic/" + String(currentYear) + "/" + currentMonthName + "/
     " + String(monthDay) + "/limite")) {  //On successful Read
     operation, function returns 1
16   //Se o retorno for != de nulo
17   if(firebaseData.dataTypeEnum() != 1) {
18     limite = firebaseData.stringData();
19     if(limite.toFloat() <= energy){
20       Firebase.setString(firebaseData, "/dispositivos/gc002/
           status", "0");
21     }
22   }
23 }
```

3.4 Configurações do Firebase

O presente projeto usa o Firebase como ferramenta de autenticação e banco de dados. O Firebase foi utilizado pelo fato dessa ferramenta ser fácil de integrar tanto no código do MCU quanto no código do sistema web. Neste capítulo será apresentado como foi feita a integração dos dois códigos, a configuração de autenticação e de banco de dados.

3.5 Configuração do Firebase no projeto

Para criar um projeto no Firebase é obrigatorio o uso de uma conta Google, acessar o console, definir o nome do projeto e criá-lo. Na sequência é necessária a vinculação do projeto criado ao seu aplicativo sendo ele web ou mobile.

O Firebase suporta projetos com o sistema operacional IOS (do inglês iPhone Operating System), Android, Web e Unity (plataforma de desenvolvimento de jogos). Cada formato necessita de configurações particulares, no caso do WebApp é necessário a instalação do kit de desenvolvimento ou SDK (Software Development Kit) que por sua vez, é feito através do gerenciador de pacotes NPM (do inglês Node Package Manager).

Será gerado então um arquivo de configuração em formato JavaScript (ou TypeScript), basta adicioná-lo ao projeto e usufruir de seus métodos. O processo todo não demora e demanda a escrita de poucas linhas de código o que agilizou o desenvolvimento.

Por fim, para que o ESP8266 consiga reconhecer qual banco de dados usar é preciso fornecer uma chave de acesso e senha. Essa chave e senha podem ser encontradas em: “Configurações do projeto” -> “Contas de serviço” -> “Chaves secretas do banco de dados”.

3.6 Configurações de Autenticação

Para configurar o Firebase Authentication devemos acessar o projeto pelo console do Firebase e escolher quais serão os métodos de login, podendo ser provedores nativos como email e telefone ou provedores externos como Google, Facebook, Twitter, Microsoft entre outros.

É possível configurar se é necessário a ativação da conta via e-mail ou SMS e definir layouts de e-mail específicos para ativação, redefinição de senha e alteração de endereço de e-mail. Também é possível gerenciar o limite de novas inscrições feitas no aplicativo usando o mesmo endereço de IP por hora, isso protege o sistema de ataques.

3.7 Configurações do Banco de Dados

O Realtime Database não necessita de muitas configurações para ser utilizado. Quando se inicia um novo projeto o desenvolvedor deve escolher em qual local será hospedado o banco de dados, as opções são América do Norte, Europa e Ásia. Por uma questão de proximidade e otimização do projeto foi escolhido hospedar o servidor na América do Norte.

Após isso é preciso definir as regras de segurança do banco de dados, nesta etapa foi inicialmente escolhido o modo teste que permite que qualquer pessoa com a referência do seu banco de dados acesse, edite e exclua todos os dados nele por 30 dias.

4 Modelagem

Neste capítulo, serão apresentados os requisitos funcionais do sistema proposto neste trabalho, ou seja, suas funcionalidades, tais requisitos serão fundamentais para a criação das interfaces e dos layouts. Serão apresentados também os diagramas de caso de uso que facilitam o entendimento da solução na totalidade.

4.1 Requisitos

Os requisitos foram identificados considerando os objetivo principais do projeto: monitorar o gasto energético e acionar o dispositivo a distância. Também foi considerado a interação entre o dispositivo físico e o usuário, por isso existem requisitos como autenticação de conta e compartilhamento de informações. Diante disso os requisitos levantados são os seguintes:

- Criar uma conta e fazer login.
  - O usuário deve ter uma conta para utilizar o sistema, o cadastro é simplificado exigindo apenas e-mail e senha. Isso permitirá que o dispositivo físico possa ser registrado em várias contas, ou seja, monitorado por várias pessoas.
- Adicionar um dispositivo em sua conta.
  - O usuário poderá cadastrar quantos dispositivos quiser, cada dispositivo tem um código padrão que deverá ser informado ao sistema. Deverá ser informado também um nome para exibição e uma cor para torná-lo diferente e facilitar a identificação.
  - Ao adicionar um dispositivo o sistema verificará se o mesmo está sendo registrado pela primeira vez ou não. Caso nunca tenha sido registrado, o primeiro usuário será considerado dono e poderá definir uma senha, caso contrário o sistema solicita a senha de acesso que foi criada pelo dono do dispositivo.
- Ligar e desligar o dispositivo a distância.
  - O sistema deve possibilitar o usuário ligar e desligar o aparelho de qualquer lugar do mundo a partir de um clique, desde que ele esteja conectado a internet.
- Mostrar os dados de energia em tempo real.

  - O sistema deve fornecer os seguintes dados em tempo real: corrente (A) , frequência (Hz), potência (W) e tensão (V).

- Mostrar o histórico de gastos.

  - O sistema deve fornecer ao usuário o histórico diário mostrando o gasto de energia em KW/H de hora em hora, das 00:00 às 23:59.
  - O sistema deve fornecer ao usuário o histórico mensal mostrando o gasto de energia em KW/H a cada dia do mês.
  - O sistema deve permitir que o usuário mude a data de ambos históricos, permitindo monitorar o gasto de dias e meses passados.

- Definir limite de gastos.

  - O usuário poderá definir um teto de gastos diário baseado em KW/H. O sistema deve informar o gasto máximo calculando o consumo total em 24 horas, esse cálculo deve ser feito utilizando a informação de potência recebida em tempo real.

- Alterar o tema da aplicação.

  - O usuário poderá escolher entre o tema claro e o tema escuro da aplicação.

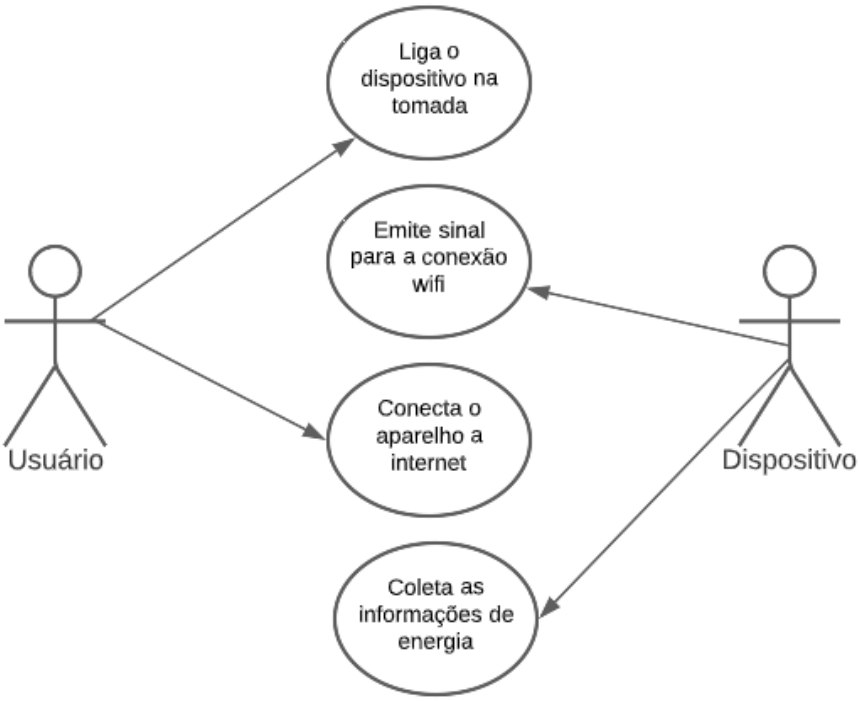
4.2 Diagrama de Casos de Uso

Após o levantamento dos requisitos, foram elaborados dois diagramas de casos de uso, visando facilitar a compreensão do funcionamento do sistema proposto. Os diagramas ajudam a modelar o fluxo básico de eventos e também a listar quais interações o usuário poderá fazer com o sistema.

Foram criados dois diagramas diferentes, um para apresentar as funcionalidades do sistema web e outro para apresentar interações do usuário com o dispositivo físico.

Na Figura 6, são apresentadas todas as iterações inerentes ao usuário utilizando o dispositivo físico.

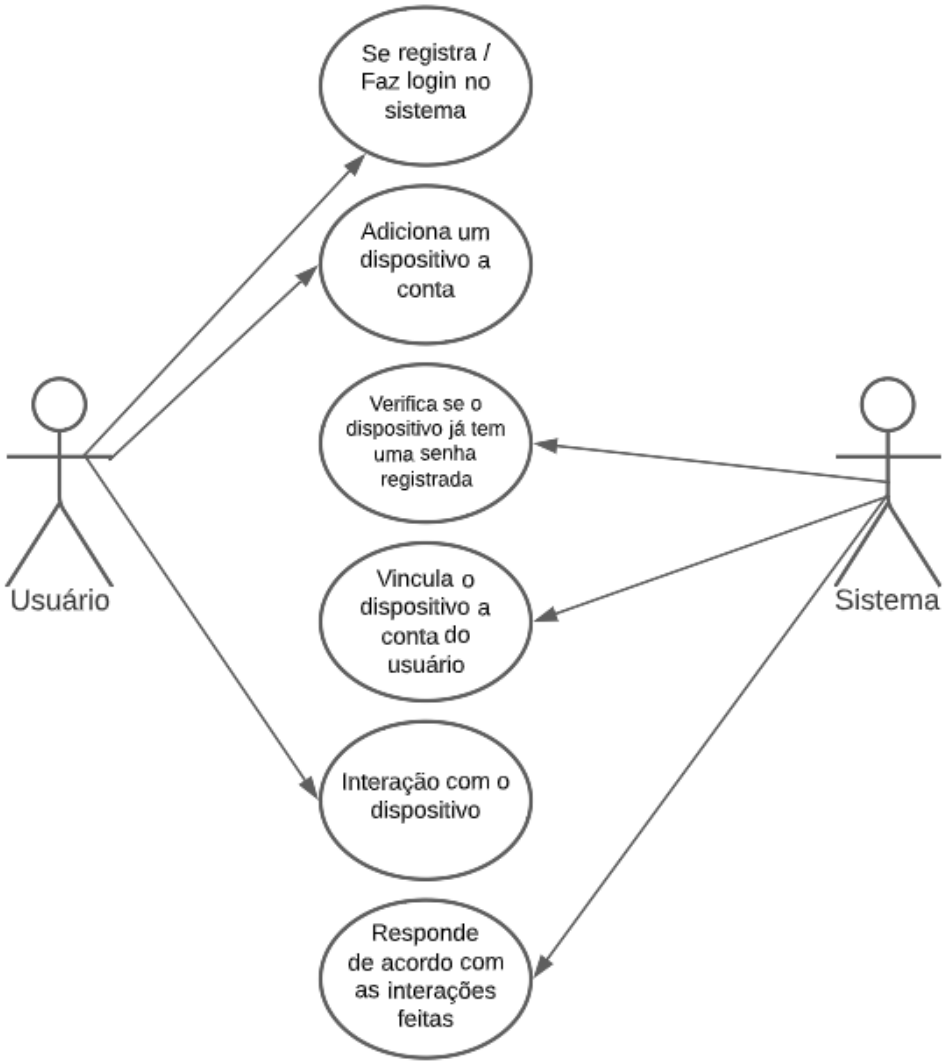
Figura 6 – Diagrama de casos de uso dispositivo



Fonte: Do autor

Na Figura 7, são apresentadas todas as funcionalidades inerentes ao usuário utilizando o sistema web.

Figura 7 – Diagrama de casos de uso sistema web



Fonte: Do autor

4.3 Banco de Dados

O banco de dados oferecido pelo Firebase armazena dados em documentos do tipo JSON e isso facilita a leitura dos dados pelo aplicativo. As informações foram divididas em duas ramificações principais: “dispositivos” e “users”.

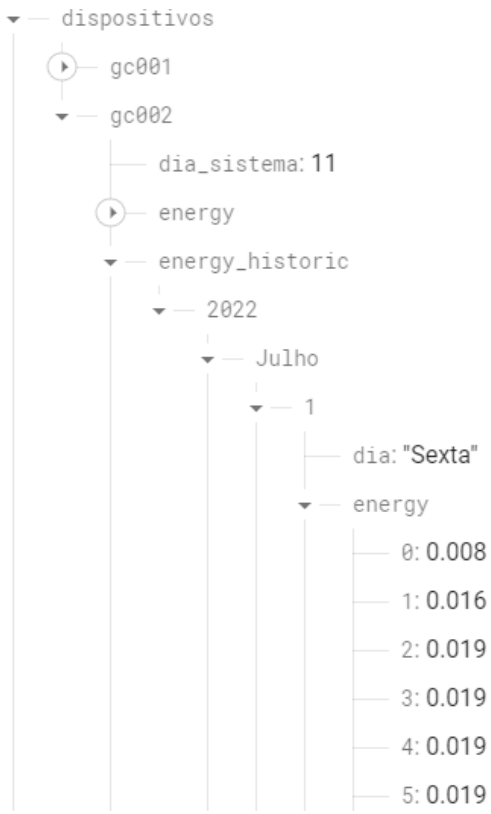
A chave “dispositivos” contém todos os dispositivos registrados no banco de dados. Cada aparelho físico conta com um código único de identificação usado para vinculá-lo à conta do usuário, esse código é a chave de cada dispositivo no banco de dados. Essas chaves únicas armazenam informações que chegam da MCU, ou seja, os dados energéticos e o histórico de consumo.

Além dos dados energéticos e de consumo, são armazenados também o status, que serve para verificar o estado do aparelho (ligado ou desligado), o dia no sistema é usado para conferir quando é preciso reiniciar o dado de consumo, e a senha caso o dispositivo já

tenha sido registrado por outro usuário.

Os dados de histórico de consumo foram divididos em ano, mês e dia. Ao final o dado de consumo é armazenado de hora em hora acumulando das 00:00 as 23:59 do dia. A Figura 8 ilustra o formato final da chave “dispositivos”.

Figura 8 – Chave dos dispositivos



Fonte: Do autor

A chave de usuários contém todos os usuários que criaram conta no sistema, nesta seção é armazenado o e-mail utilizado para efetuar o cadastro e os dispositivos vinculados à conta. A chave de dispositivos vinculados serve apenas para armazenar informações que podem ser definidas pelos usuários como: nome do dispositivo no sistema, cor e local.

Desta forma é possível que cada usuário customize da forma que preferir como esse dispositivo será exibido na tela sem forçar esse padrão para usuários que vincularem o mesmo dispositivo no futuro. A Figura 9 ilustra o formato final da chave “users”:

Figura 9 – Chave dos usuários

Fonte: Do autor

4.4 Decisões de Design

A partir dos diagramas e dos fluxogramas criados deu-se início a criação da interface gráfica. Primeiramente foram desenvolvidos esboços das telas usando a ferramenta Figma. Tais esboços são comumente chamados de wireframes.

4.5 Wireframes

Para realizar a prototipação foi utilizado uma ferramenta gratuita chamada Figma. Essa ferramenta auxilia na criação de telas, que podem interagir entre si, e ser compartilhadas com outras pessoas para a criação em conjunto. A prototipação é uma parte essencial para a criação do sistema web, pois esse rascunho permite saber quantos componentes serão criados e como serão organizados em tela.

A Figura 10 abaixo mostra o wireframe da tela de login e da tela principal que servirão de exemplo, os componentes estão em escalas de cinza, uma vez que a neste momento a paleta de cores ainda não tinha sido definida. O resultado final dessas telas será apresentado na seção Testes e Resultados no final do trabalho.

Figura 10 – Wireframe das telas

Fonte: Do autor

4.6 Paleta de Cores e Temas

As principais cores escolhidas para o sistema foram o azul e o amarelo, por serem cores complementares e terem um contraste agradável entre si. O sistema também oferece dois tipos de temas: claro e escuro, isso foi criado pensando na experiência do usuário com o intuito de deixar a interface mais amigável à visão.

A paleta conta com duas cores, primária e secundária, que invertem de papel caso o tema seja alterado, a Figura 11 abaixo exemplifica o que foi feito.

Figura 11 – Paleta de cores

Fonte: Do autor

4.7 Implementação das telas

Para facilitar o desenvolvimento da interface gráfica do usuário, foi utilizado a biblioteca Material UI. Essa é uma biblioteca gratuita que implementa os componentes do Material Design, uma linguagem de design desenvolvida pela Google, para o React.

Os componentes do Material UI são autossuficientes e tem estilos default que são suaves mesmo sem estilização. Porém, para customizar de forma eficiente e individual foi utilizado a biblioteca Styled-Components, para a criação de componentes em arquivos únicos, mas que por sua vez, serão utilizados várias vezes ao longo do código. Como o Styled Components permite escrever CSS no JavaScript é possível passar atributos como propriedades para o componente, mudando o seu comportamento conforme o que foi passado.

O uso de tais bibliotecas é justificado pela agilidade e simplicidade no desenvolvimento do sistema, uma vez que, os componentes estão prontos necessitando apenas de estilização conforme o que faz sentido esteticamente.

4.8 Responsividade

O sistema foi projetado para poder oferecer acesso remoto de qualquer dispositivo, seja ele mobile através de interações com o dedo ou desktop utilizando mouse e teclado. Por isso é de grande importância adaptar o sistema para que as telas possam ser exibidas de forma confortável independente da tela.

A biblioteca Material UI oferece pontos de quebra (breakpoints em inglês) baseados na largura da tela do dispositivo, esses são pontos que definem o limite de uma tela. No projeto foram utilizados os pontos de quebra padrão da biblioteca que estão detalhados no Anexo 2.

Com isso é possível realizar operações de condição determinando o que aparece na tela conforme o ponto de quebra definido. Juntamente com isso foi utilizado conceitos de flexbox do CSS para mudar a direção dos componentes na tela, quanto menor a largura da tela mais horizontal o conteúdo deve ser apresentado.

5 Testes e Resultados

5.1 Telas

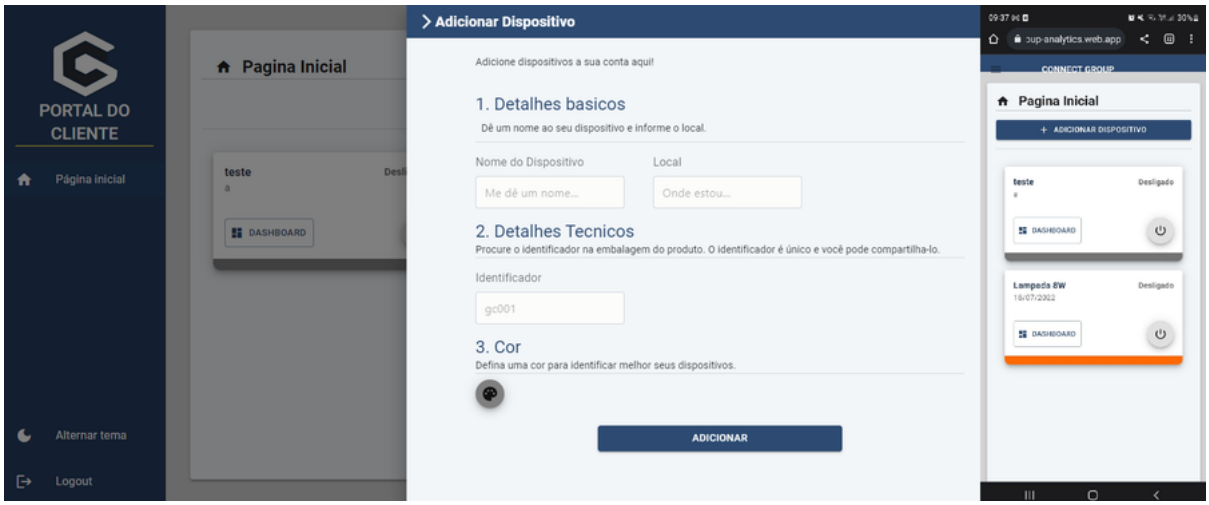
O presente capítulo irá apresentar as telas do sistema em suas duas versões desktop e mobile. A Figura 12 apresenta o resultado da tela de Login. Nela o usuário deve informar seu e-mail e senha para poder usar as funcionalidades do sistema. Além disso, poderá se cadastrar clicando em “CRIAR UMA CONTA”, caso o faça abrirá uma janela onde o cliente deve fornecer um e-mail e uma senha para se cadastrar.

Figura 12 – Tela de Login

Fonte: Do autor

Após o credenciamento o cliente será direcionado a tela principal do aplicativo que inicialmente estará vazia. Para adicionar um novo dispositivo o cliente deve clicar no botão “ADICIONAR DISPOSITIVO” no canto superior direito da tela. Assim abrirá um menu lateral onde o cliente preencherá os dados do produto.

Figura 13 – Tela Inicial

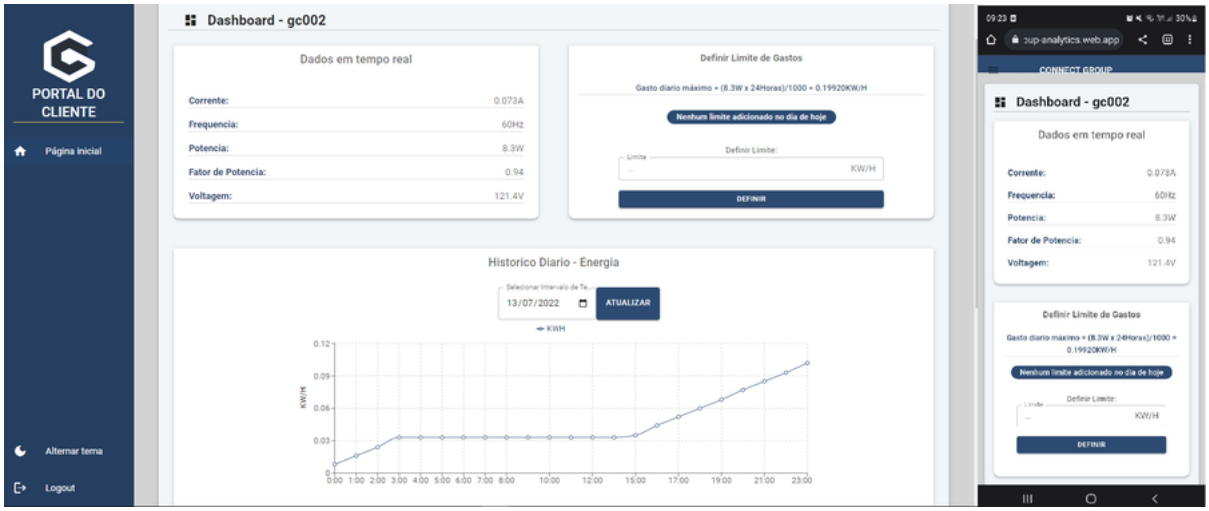


Fonte: Do autor

Como descrito nos requisitos, caso o produto esteja sendo registrado pela primeira vez o usuário deve criar uma senha, caso já tenha sido registrado deve-se fornecer a senha do dispositivo. Após isso o dispositivo será vinculado a conta e será possível visualizá-lo na página principal.

Ao clicar em “DASHBOARD” o usuário será direcionado para tela onde estão os dados energéticos do dispositivo. Nessa tela é possível ver os dados em tempo real, o histórico diário e mensal de consumo e também definir um limite de gastos diário.

Figura 14 – Tela de Dashboard



Fonte: Do autor

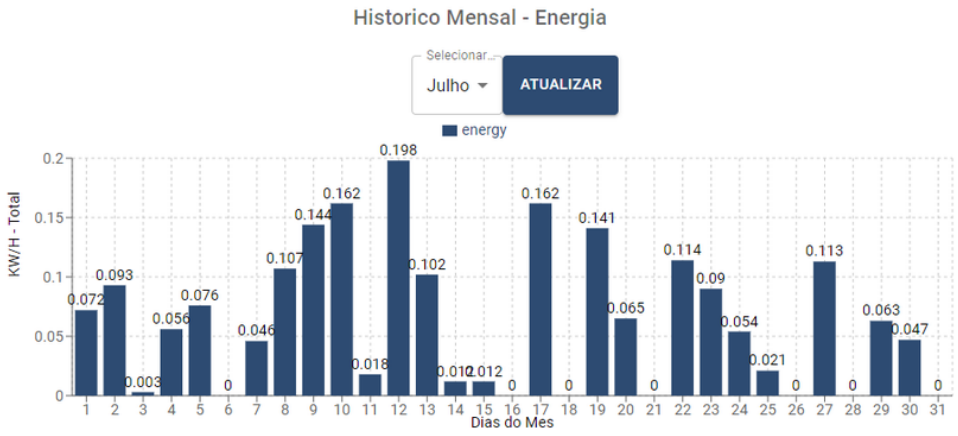
5.2 Medições realizadas

Para os testes foi usado uma lâmpada de 8W e seu consumo foi medido entre 01/07/2022 a 31/07/2022. A taxa de atualização é feita de hora em hora graças a isso

é possível ver à que hora do dia a lâmpada fica ligada. Também é possível distinguir o consumo entre todos os dias do mês.

A Figura 15 a seguir mostra o gráfico do consumo em todo mês de julho. É possível notar que no dia 12/07/2022 a lâmpada ficou ligada o dia inteiro, pois, como sua potência é de 8W e ela ficou ligada 24 horas o cálculo do consumo é: $\frac{8X24}{1000} = 0.192$

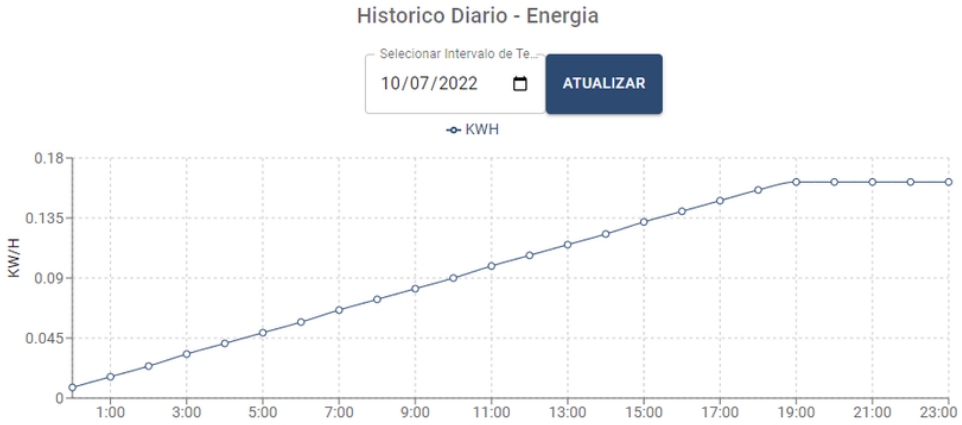
Figura 15 – Gráfico mensal de consumo



Fonte: Do autor

A Figura 16 mostra o consumo específico do dia 10/07/2022. Com esse gráfico é possível analisar que a lâmpada ficou acesa de 00:00 as 19:00, pois, a linha sobe até chegar às 19 horas e então se estabiliza e mantém um valor constante, ou seja, sem registrar gastos, logo está desligada.

Figura 16 – Gráfico diário de consumo



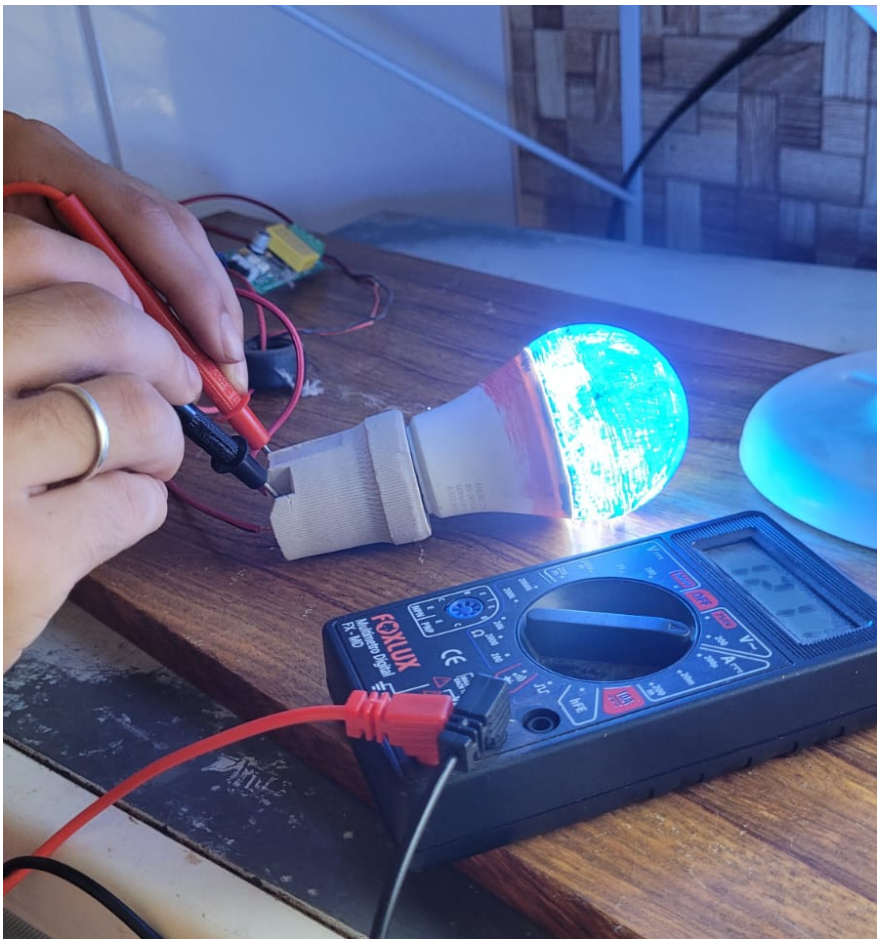
Fonte: Do autor

5.3 Testes

Para verificar se os dados de energia coletados condizem com os dados reais foi realizada a medição do circuito utilizando um multímetro. Foram verificadas 2 grandezas a

corrente e a tensão. Para medição da tensão, os cabos do multímetro foram posicionados nos conectores do bocal da lâmpada, como mostra a Figura 17.

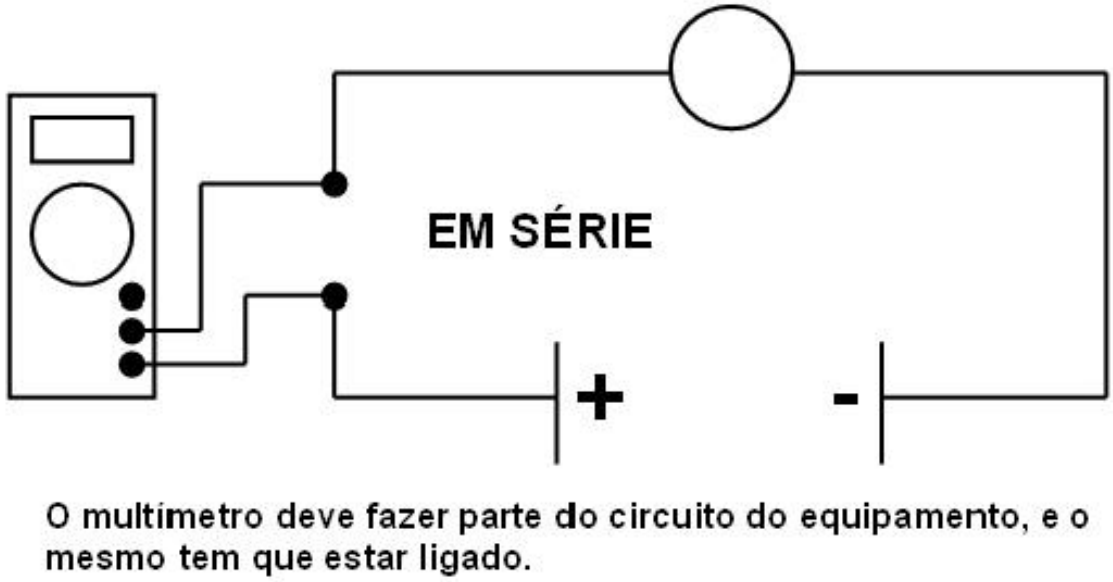
Figura 17 – Medição da tensão



Fonte: Do Autor

Para medir a corrente foi necessário abrir o circuito e posicionar o multímetro entre as conexões do fio, como exemplifica a Figura18.

Figura 18 – Medição da corrente



Fonte: Hardware Livre USP

Os resultados obtidos pelo multímetro foram verificados coerentes com os dados apresentados pelo sistema, ocorrendo algumas perdas. A Tabela 2 faz uma comparação das medições coletadas.

Tabela 1 – Comparação de medidas registradas

Grandezas	Multímetro	Pzem-004T
Corrente	0.07A	0.072A
Tensão	121.1 V	122.5 V

Fonte: Do autor

As demais grandezas energéticas não foram medidas devido à incapacidade do multímetro em questão, porém a lampada usada para teste possui um indicador de 8W o que condiz com o dado capturado pelo dispositivo.

6 Conclusão

Ao longo da realização do trabalho percebe-se que os conceitos de IoT (*Internet of Things*) podem oferecer grande ajuda na área de sustentabilidade como visto no texto de Carvalho (2020), que propõe a criação de telhados verdes utilizando a tecnologia da placa ESP8266 e sensores de umidade.

O dispositivo e o sistema proposto no trabalho oferecem ao usuário definir um limite de gastos diários de um determinado aparelho elétrico e, além disso, permite o acionamento e desligamento à distância. Essas duas funcionalidades oferecem eficiência energética e uma economia de energia para a máquina a qual o dispositivo foi acoplado.

O trabalho obteve êxito, o medidor de energia funciona e é acessível em questões financeiras. Foi possível também testá-lo e verificar duas das grandezas coletadas utilizando um medidor multímetro. Pelo fato do circuito elétrico não ser o foco do trabalho, é possível melhorá-lo integrando todos os componentes em uma placa impressa e soldando-os.

O banco de dados do Firebase se mostrou uma ferramenta importantíssima para a realização do trabalho. A quantidade de dados e a frequência de dados coletados possibilitam o uso gratuito da ferramenta. Caso seja proposto uma mudança onde os dados sejam coletados de minuto a minuto ou de segundo a segundo o Firebase seria um gargalo para o projeto, pois, existe um limite de downloads e uploads de informação na versão gratuita do Firebase.

Os resultados obtidos no experimento com a lâmpada foram satisfatórios, os dados coletados batem com a descrição do produto na embalagem e condizem com o tempo de testes.

7 Referências

ANDRADE, G. H. E. de. **Desenvolvimento de um sistema de monitoramento energético controlado pela internet.** 2016 — Universidade Tecnológica Federal do Paraná.

ANDRADE, L. H. de O. **Analisador de eficiência energética conectado à internet.** 2021. Monografia (Engenharia Elétrica) — Universidade Federal de Uberlândia.

BANZI, M.; SHILOH, M. **Primeiros Passos com o Arduino.** Novatec, 2011.

CHAVIER, L. F. **Programação para Arduino - Primeiros Passos.** [S.l.], 2013.

NASCIMENTO, B. do. Medidor de grandezas elétricas com acesso remoto. **Revista eletrônica de engenharia elétrica e engenharia mecânica**, v. 2, n. 1, p. 42 – 53, 2020.

FIREBASE GOOGLE. **Documentação do Firebase.** 2022. Disponível em: https://firebase.google.com/docs?hl=pt-br. Acesso em: 10 de maio de 2022.

GARCIA, Marcus Valério Rocha et al. IoT EcoHome: Internet das coisas e Sustentabilidade. **Brasil Para Todos-Revista Internacional**, v. 8, n. 1, p. 27-32, 2020.

SILVA, L. **Básico de Eletrônica – Parte II.** 2002. Disponível em: https://electraenergy.com.br/faq-tire-suas-duvidas-sobre-a-bandeira-da-escassez-hidrica/. Acesso em: Maio de 2022.

MONK, Simon. **Programação com Arduino: começando com Sketches.** Bookman Editora, 2013.

MUNDO PROJETADO. **Relé – O que é e como funciona.** 2017. Disponível em: https://mundoprojetado.com.br/rele-o-que-e-e-como-funciona/. Acesso em: Agosto de 2022.

MURATORI, José R.; BÓ, P. H. D. **Automação residencial: Histórico, definições e conceitos.** 2016.

OLIVEIRA, Isabella Ferreira de. **Desenvolvimento de um sistema de automação residencial baseado em IoT para controle e monitoramento de dispositivos elétricos.** 2019. 70 f. Monografia (Graduação em Engenharia de Controle e Automação) - Escola de Minas, Universidade Federal de Ouro Preto, Ouro Preto, 2019.

PEREIRA, CEP; CARVALHO, F. **A internet das coisas (iot): Cenário e perspectivas no brasil e aplicações práticas.** VII SRST–Seminário de Redes e Telecomunicações, 2017.

SANTOS, Bruno P. et al. **Internet das coisas: da teoria à prática.** Minicursos SBRC-Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuıdos, v. 31, p. 16, 2016.

SANTOS, Jean Willian; LARA JUNIOR, Renato Capelin de. **Sistema de automatização residencial de baixo custo controlado pelo microcontrolador esp32 e monitorado via smartphone.** 2019. Trabalho de Conclusão de Curso. Universidade Tecnológica Federal do Paraná.

SILVA, W. C. S. da. **Aplicações móveis nativas com react native e firebase: um estudo de caso.** Monografia (Ciência da Computação) — Universidade Federal do Maranhão.

SILVEIRA, M. A. da; SANTOS, L. A. dos; ROSÁRIO, D. B. do. Automação Residencial – Um grande negócio. In: UNIFACEAR (Ed.). **Revista Eletrônica Multidisciplinar.** 2015.

SISTEMA-ONS. **Previsão de carga para o Planejamento Anual da Operação Energética.** 2002. Disponível em: https://electraenergy.com.br/ faq-tire-suas-duvidas-sobre-a-bandeira-da-escassez-hidrica/. Acesso em: Fevereiro de 2023.

Apêndices

APÊNDICE A – Código Arduino

Código A.1 – Código compilado no chip ESP8266

```
#include  //Inclui a biblioteca do WiFi da ESP8266
#include   //Inclui a biblioteca do FireBase https://github.com/mobizt/Firebase-ESP8266/
#include
#include
#include
#include   //Inclui a biblioteca do WifiManager https://github.com/tzapu/WiFiManager
#include
#include  //Biblioteca NTPClient para trabalhar com tempo

#define FIREBASE_HOST "connect-group-analytics-default-rtdb.firebaseio.com"
#define FIREBASE_AUTH "uiJ8R81kbHJGTdWJtYrFZ3fnowF7wkuqstYHQlUE"

WiFiUDP udp; //Socket UDP que a lib do NTP utiliza para recuperar dados sobre o horário

int rele = 0;     //Saida padrão da ESP01 - Modulo relé 5v- GPIO 0
String val, valLigar, valDesligar, limite;      //Variavel para conferir o estado no firebase
int dia_sistema;
//Week Days
String weekDays[7]={"Domingo", "Segunda", "Terça", "Quarta", "Quinta", "Sexta", "Sabado"};

//Month names
String months[12]={"Janeiro", "Fevereiro", "Março", "Abril", "Maio", "Junho", "Julho", "Agosto", "Setembro", "Outubro", "Novembro", "Dezembro"};

SoftwareSerial pzemSWSerial(3, 1);
PZEM004Tv30 pzem;

FirebaseData firebaseData;

NTPClient ntpClient(udp, "br.pool.ntp.org");

void setup()
{
  Serial.begin(115200); //Inicia a Conexão Serial
  WiFiManager wifiManager; // Inicia o WiFiManager
  //wifiManager.resetSettings(); //Reseta as credenciais de WiFi
  wifiManager.autoConnect("Grupo Connect"); //Inicia o autoConnect com o nome Grupo Conect

  Firebase.begin(FIREBASE_HOST, FIREBASE_AUTH); //Inicia a conexão com o Firebase
  ntpClient.begin(); //Inicia a variavel ntp
  ntpClient.setTimeOffset(-10800);

```

```
42   pinMode(rele,OUTPUT); // Seta que o "rele" é de saida
43   pzem = PZEM004Tv30(pzemSWSerial); //Seta pinos do pzem
44 }
45
46 void loop() {
47
48   //Time updates
49   ntpClient.update();
50   unsigned long epochTime = ntpClient.getEpochTime();
51   struct tm *ptm = gmtime ((time_t *)&epochTime);
52   int hours = ntpClient.getHours();
53   int monthDay = ptm->tm_mday;
54   int currentYear = ptm->tm_year+1900;
55   String weekDay = weekDays[ntpClient.getDay()];
56   int currentMonth = ptm->tm_mon+1;
57   String currentMonthName = months[currentMonth-1];
58
59   // Read the data from the sensor
60   float voltage = pzem.voltage();
61   float current = pzem.current();
62   float power = pzem.power();
63   float energy = pzem.energy();
64   float frequency = pzem.frequency();
65   float pf = pzem.pf();
66   Firebase.setFloat(firebaseData, "/dispositivos/gc002/energy/
        voltage", voltage);
67   Firebase.setFloat(firebaseData, "/dispositivos/gc002/energy/
        current", current);
68   Firebase.setFloat(firebaseData, "/dispositivos/gc002/energy/
        power", power);
69   Firebase.setFloat(firebaseData, "/dispositivos/gc002/energy/
        frequency", frequency);
70   Firebase.setFloat(firebaseData, "/dispositivos/gc002/energy/
        powerfactor", pf);
71
72   Firebase.setString(firebaseData, "/dispositivos/gc002/energy_
        historic/" + String(currentYear) + "/" + currentMonthName +
        "/" + String(monthDay) + "/mes" , currentMonthName);
73   Firebase.setString(firebaseData, "/dispositivos/gc002/energy_
        historic/" + String(currentYear) + "/" + currentMonthName +
        "/" + String(monthDay) + "/dia" , weekDay);
74   Firebase.setFloat(firebaseData,
75      "/dispositivos/gc002/energy_historic/" + String(currentYear)
            + "/" + currentMonthName + "/" + String(monthDay) + "/
          energy/" + String(hours), energy);
76
77   if (Firebase.getInt(firebaseData, "/dispositivos/gc002/dia_
        sistema")) {  //On successful Read operation, function
        returns 1
78
79     dia_sistema = firebaseData.intData();
80     if(dia_sistema != monthDay){
81       pzem.resetEnergy();
82       Firebase.setInt(firebaseData, "/dispositivos/gc002/dia_
           sistema" , monthDay);
83     }
84
```

```
85    } else {
86      Serial.println(firebaseData.errorReason());
87    }
88
89    if (Firebase.getString(firebaseData, "/dispositivos/gc002/
          status")) {  //On successful Read operation, function
          returns 1
90
91      val = firebaseData.stringData();
92      if(val == "0"){
93        digitalWrite(rele, LOW); // Desliga o rele
94        //Serial.println(val);
95      }
96      if (val == "1"){
97        digitalWrite(rele, HIGH); // Liga o rele
98        //Serial.println(val);
99      }
100
101   } else {
102     Serial.println(firebaseData.errorReason());
103   }
104
105     if (Firebase.getString(firebaseData, "/dispositivos/gc002/
            energy_historic/" + String(currentYear) + "/" +
            currentMonthName + "/" + String(monthDay) + "/limite")) {
             //On successful Read operation, function returns 1
106     //Se o retorno for != de nulo
107     if(firebaseData.dataTypeEnum() != 1) {
108       limite = firebaseData.stringData();
109       if(limite.toFloat() <= energy){
110         Firebase.setString(firebaseData, "/dispositivos/gc002/
                status", "0");
111       }
112     }
113   }
114
115   delay(150);
116 }
```

Anexos

ANEXO A – Faixa de medição PZEM-004

PZEM-004		
Grandezas	Faixa de Medição	Precisão
Tensão de Alimentação [V]	3,3 a 5,5	-
Tensão [V]	80 a 260	0,5%
Corrente [A]	0 a 100	0,5%
Potência ativa [W]	0 a 23×10^3	0,5%
Fator de potência [Φ]	0 a 1	1%
Frequência [Hz]	45 a 65	0,5%
Energia ativa [kWh]	0 a 9999,00	0,5%

Fonte: GitHub. Disponível em: <https://github.com/mandulaj/PZEM-004T-v30>

ANEXO B – Pontos de quebra padrão

- **xs: extra-small:** Muito pequeno, largura de 0px;
- **sm: small:** Pequeno, largura de 600px;
- **md: medium:** Médio, largura de 900px;
- **lg: large:** Grande, largura de 1200px;
- **xl: extra-large:** Muito grande, largura de 1536px;