DESENVOLVIMENTO DE UMA SOLUÇÃO DE *HARDWARE* E *SOFTWARE* PARA ANÁLISE DE CONSUMO DE ENERGIA RESIDENCIAL.

LORENZO CHIARELLO WEFFORT[1], ANDRÉ LUIS OLIVETE[2]

[1] Graduando em Bacharelado em Ciência da Computação, IFSP, Câmpus Presidente Epitácio, lorenzo.weffort@aluno.ifsp.edu.br.
[2] Docente, IFSP, Câmpus Presidente Epitácio, olivete@ifsp.edu.br.
Área de conhecimento: 1.03.00.00-7 Ciência da Computação

RESUMO:

O presente trabalho apresenta o desenvolvimento de um solução para monitoramento do consumo de energia elétrica que permite a coleta dos dados, o armazenamento em nuvem e a análise com técnicas de Inteligência Artificial (IA) para detecção de anomalias. O sistema proposto monitora continuamente o nível de corrente elétrica, enviando os dados coletados para um servidor na nuvem. Esses dados posteriormente serão utilizados para fazer análise com algoritmos de IA com o objetivo de identificar padrões de consumo e possíveis anomalias, indicando problemas como sobrecargas ou condições operacionais não ideais. Os dados são acessíveis em tempo real através de uma interface *web* desenvolvida para uma fácil visualização das informações coletadas.

PALAVRAS-CHAVE: monitoramento de energia elétrica; microcontroladores; protocolo mqtt; séries temporais; detecção de anomalias

DEVELOPMENT OF A HARDWARE AND SOFTWARE SOLUTION FOR ANALYZING RESIDENTIAL ENERGY CONSUMPTION.

ABSTRACT: This paper presents the development of an IoT device for monitoring electricity consumption, data collection, cloud storage, and analysis with Artificial Intelligence (AI) techniques for anomaly detection. The proposed device continuously monitors the level of electrical current, sending the collected data to a cloud server. This data is then analyzed by AI algorithms that identify consumption patterns and possible anomalies, indicating problems such as overloads or non-ideal operating conditions. The data is accessible in real time through a *web* interface developed for easy viewing of the collected information.

KEYWORDS: electrical energy monitoring; microcontrollers; non-invasive current sensors; mqtt protocol; time series; anomaly detection

INTRODUÇÃO

Nos últimos anos, a Internet das Coisas (IoT) tem transformado diversos setores da indústria, proporcionando novas oportunidades para monitoramento e controle remoto de sistemas e dispositivos. Entre as muitas aplicações da IoT, o monitoramento de energia tem ganhado destaque, especialmente com o aumento da demanda por eficiência energética e a necessidade de detecção precoce de anomalias em sistemas elétricos.

Este trabalho propõe o desenvolvimento de um dispositivo IoT para coletar dados sobre o nível de corrente elétrica e armazená-los na nuvem e de um sistema *web* e *mobile* para o monitoramento dos dados coletados, que utiliza técnicas de Inteligência Artificial para análise e detecção de anomalias.

O dispositivo desenvolvido neste projeto será capaz de monitorar continuamente o nível de corrente elétrica em um sistema, enviando os dados coletados para um servidor MQTT em nuvem, que estará alimentando um servidor responsável por armazenar os dados enviados ao servidor MQTT.

Uma vez armazenados, os dados são submetidos a algoritmos de IA que analisam os padrões de consumo e identificam possíveis anomalias. Essas anomalias podem indicar problemas como sobrecargas ou condições operacionais não ideais. Como os dados são coletados em sequência, com período de tempo pré-definido entre as coletas, sendo definida como uma série temporal, permitindo utilizar as principais técnicas para análise de séries temporais.

Similar a esse trabalho proposto, Martins (2020) propõe a utilização de aprendizado de máquina usando modelos LSTM-RNNS na detecção de anomalias de séries temporais. Em Campos (2008) é apresentado um estudo de previsão de séries temporais não estacionárias de consumo de energia elétrica abrangendo várias metodologias de modelagem.

Além da análise automatizada, os dados coletados são disponibilizados em uma interface *web*, permitindo que os usuários visualizem em tempo real o comportamento do sistema elétrico monitorado. A aplicação também oferece funcionalidades para a gestão dos dados armazenados e possibilita a visualização de gráficos que auxiliam na interpretação dos dados coletados.

Assim, este trabalho contribui para a área de monitoramento inteligente de sistemas elétricos, apresentando uma solução que não apenas coleta e armazena dados de corrente elétrica, mas também utiliza análises avançadas para melhorar a eficiência e a segurança dos sistemas monitorados.

MATERIAL E MÉTODOS

O desenvolvimento do projeto segue um conjunto de atividades definidas em cronograma próprio, iniciando por um estudo sobre as alternativas de comunicação entre o dispositivo proposto e o sistema de armazenamento dos dados coletados, onde foi definido o protocolo MQTT (*Message Queuing Telemetry Transport*) para essa comunicação, por ser um protocolo leve e que permite a troca de mensagens utilizando filas (Perez, 2012). É um protocolo de mensagens que se baseia em se inscrever e publicar mensagens em tópicos de um *broker*, que é o servidor responsável por filtrar as mensagens desses tópicos onde cada cliente pode se inscrever em um ou mais tópicos.

Um estudo foi realizado para entender as características técnicas e funcionais do MQTT, incluindo suas vantagens em termos de eficiência e baixo consumo de banda, uma vez que é projetado para ser leve e suportar uma gama de linguagens e frameworks como Java, Javascript, Python, Arduino, etc. Isso permite que, desde aplicação *web* ou aplicação desktop, até microcontroladores possam enviar e receber mensagens pelo protocolo MQTT, o que é ideal para esse projeto.

Quando uma mensagem é enviada, ela é destinada a um tópico específico, e todos os clientes inscritos no tópico de destino podem receber a mensagem. Tópicos podem conter diferentes subtópicos, indicado pela barra (“/”), como por exemplo: “casa/quarto/ar_condicionado”. Também é possível utilizar curingas, que fornecem um mecanismo para se inscrever em vários tópicos simultaneamente. Há dois tipos de curingas: de nível único representado por “+” e de vários níveis, representado por “#”.

O passo seguinte consistiu no estudo aprofundado dos microcontroladores ESP32 e ESP8266, que são amplamente utilizados em projetos de automação devido à sua capacidade de conexão WiFi e flexibilidade de uso. As arquiteturas do *hardware* dos microcontroladores foram estudadas, incluindo seus pinos de entrada e saída, capacidade de memória e modos de energia. Experimentos práticos foram conduzidos para programar esses dispositivos utilizando a IDE do Arduino, verificando a funcionalidade básica como leitura de sensores e controle de atuadores.

Para a medição de corrente elétrica, foi realizado um levantamento sobre os possíveis sensores, e com base nesse estudo, foi definido o sensor SCT-013 devido à sua característica não invasiva, o que facilita a instalação sem a necessidade de modificar a fiação elétrica existente (Lima, 2017). O estudo envolveu a compreensão do princípio de funcionamento deste tipo de sensor, que é baseado no efeito Hall e permite a medição de corrente alternada sem contato direto.

O banco de dados será utilizado para armazenar os dados enviados para o *broker* MQTT e recebidos pelo servidor da aplicação, permitindo consultas para a visualização e análise desses dados. Foi utilizado o SQLite na aplicação Python Flask, que oferece uma solução leve e portátil, ideal para desenvolvimento rápido e de fácil implantação. A figura 1 mostra o modelo de dados, detalhando suas relações e atributos de cada tabela, onde a tabela **local** armazenará dados referentes ao local onde o dispositivo de monitoramento está instalado, a tabela **dispositivo** será responsável por armazenar o dispositivo, pois poderão ser instalados vários dispositivos em um mesmo local permitindo monitorar diversos circuitos de energia dentro do local, na tabela **sensor** ficarão armazenados informações sobre o circuito que está sendo monitorado pelos sensores conectados aos dispositivos e a tabela de **medição** permitirá armazenar todos os dados enviados pelos dispositivos.

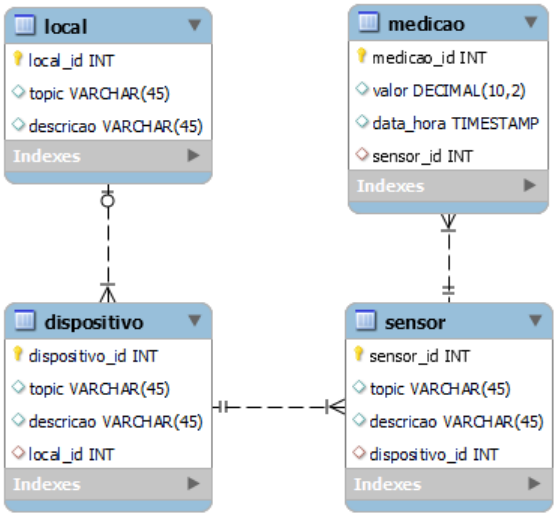

FIGURA 1. Modelagem do banco de dados.

O servidor responsável por receber os dados do *broker*, armazená-los em um banco de dados e posteriormente disponibilizar para os clientes está sendo desenvolvido em Python com o *framework* Flask, um *microframework* conhecido por sua leveza e flexibilidade. Esse servidor permitirá alimentar o sistema com os locais onde os dispositivos serão instalados, juntamente com seus sensores, e disponibilizará dados para o monitoramento dos circuitos. Atualmente foi implementado um sistema *web* convencional que permite realizar todas essas ações, possibilitando realizar as consultas nos dados do banco e visualizar através de gráficos utilizando a biblioteca Matplotlib.

Depois da coleta dos dados feito pela aplicação temos a chamada série temporal, uma sequência de dados numéricos em ordem sucessiva, ocorrendo em intervalos uniformes. Então, faremos a utilização de redes neurais para detecção de anomalias em séries temporais, que é uma abordagem poderosa capaz de capturar padrões complexos e dependências temporais nos dados.

Primeiro é feito um pré-processamento dos dados para garantir que estejam em um formato adequado para treinamento da rede neural. Para a detecção de anomalias em séries temporais, as Redes Neurais Recorrentes (RNNs) e suas variantes, como as *Long Short-Term Memory* (LSTM), são frequentemente utilizadas devido à sua capacidade de capturar dependências temporais de longo prazo nos dados (IBM, 2024).

O modelo será treinado utilizando a série temporal normalizada e segmentada. Durante o treinamento, o modelo aprende a prever o próximo valor na série temporal com base nos valores anteriores. Após o treinamento, o modelo pode ser usado para prever os próximos pontos na série temporal. Anomalias são detectadas ao comparar as previsões do modelo com os valores observados.

O desempenho do modelo de detecção de anomalias deve ser avaliado após a previsão utilizando métricas apropriadas, como taxa de falsos positivos, taxa de falsos negativos e precisão para

garantir que os resultados sejam coerentes. Posteriormente, será feita a implementação em um ambiente onde são monitorados os dados recebidos em tempo real e, após essa implementação, continuará a ser monitorado o desempenho do modelo para verificar sua eficácia.

RESULTADOS E DISCUSSÃO

Com os testes em *hardware* real utilizando o ESP8266, foi possível realizar postagens para o servidor MQTT EMQX na *web*, enviando mensagens para os tópicos configurados. Isso envolveu a configuração do ESP8266 para se conectar ao servidor EMQX e publicar mensagens nos tópicos específicos. Os testes foram bem-sucedidos, permitindo validar a comunicação entre o microcontrolador e o servidor MQTT, comprovando a viabilidade da implementação no ambiente real.

A arquitetura do sistema proposto foi definida para integrar todos os componentes do sistema de forma coesa. A arquitetura definida inclui o ESP8266 com sensores de corrente elétrica conectados, o servidor EMQX para a centralização das mensagens MQTT, a aplicação utilizando API REST para armazenamento e gestão dos dados com o banco de dados, do qual armazena as mensagens, e para exibir para os dispositivos.

A figura 2 apresenta a arquitetura do sistema proposto, integrado com todas as partes mencionadas, onde há vários sensores de corrente coletando os valores da rede em locais diferentes, esses sensores estão conectados a microcontroladores, no caso do ESP8266 e ESP32, podem conectar até quatro sensores em um único microcontrolador, do qual processa os valores recebidos dos sensores e então enviam mensagens para um tópico especificado do MQTT, após a mensagem ser filtrada pelo *broker* EMQX para o tópico de destino, entra a API REST, que tem a função de se comunicar com o *broker* para resgatar os dados recebidos a guardá-los no banco de dados, além de resgatar desse armazenamento quando requisitado para exibir para os dispositivos com um *software* responsável por fazer a requisição dessas informações, que podem ser um celular ou computador/notebook.

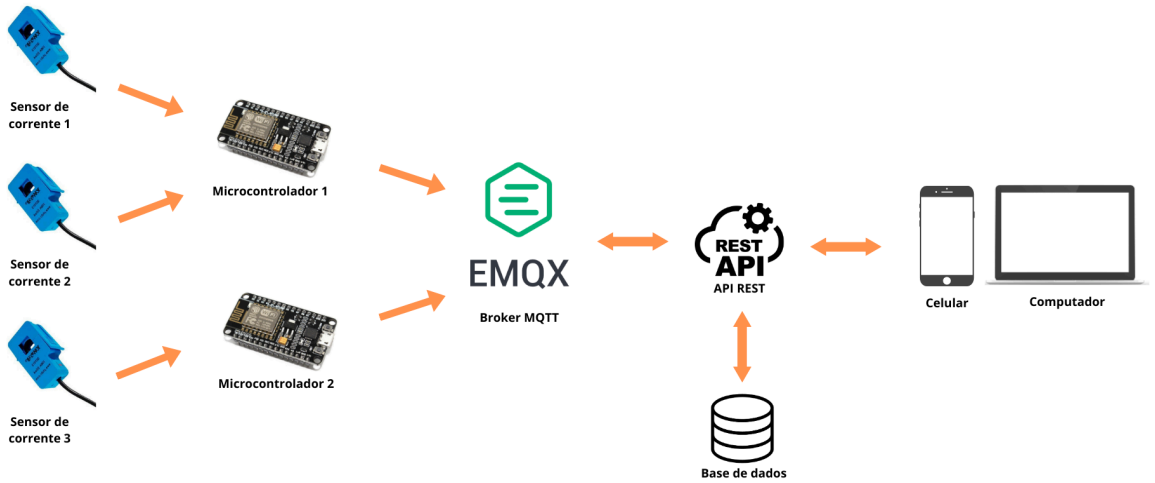

FIGURA 2. Arquitetura do sistema integrado.

Decorrente disso, foi implementado um sistema *web* utilizando Python com o *framework* Flask para proporcionar uma interface de cliente, onde será possível fazer a consulta e manipulação dos dados coletados, além da filtragem por tópicos e datas para facilitar o monitoramento e uma visualização de dados através de gráficos, como séries temporais, fornecendo uma visão clara das informações coletadas pelo sistema.

Com o objetivo de configurar o dispositivo em qualquer rede de computadores WiFi, foi realizado um estudo para utilizar a memória não volátil do dispositivo para armazenar as informações como SSID da rede, senha de acesso, além dos tópicos utilizados pelo dispositivo. Para a definição desses parâmetros importantes para o funcionamento do sistema como um todo, foi implementado um servidor *web* que provê uma página de configuração, que pode ser acessada de um celular ou computador e entrar com essas informações.

A figura 3 detalha esse fluxo de conexão do microcontrolador, onde inicialmente o *software* interno procura por uma rede WiFi para conectar o microcontrolador, caso não tenha um um WiFi configurado nele, ou não a encontre, é criado um ponto de acesso local, que fica visível a qualquer dispositivo que possa se conectar a um rede WiFi normal, por exemplo um celular, que então se conecta ao ponto de acesso do microcontrolador e acessa em qualquer navegador de internet o IP dessa rede, assim ficará disponível uma tela de configuração básica onde o usuário pode inserir uma rede WiFi com acesso a internet e sua senha, além de um tópico, e o microcontrolador salva essas informações em sua memória interna e reinicia, então o processo retorna ao estado inicial, mas dessa vez com as informações para se conectar a uma rede válida inserida pelo usuário, quando a conexão é bem sucedida o microcontrolador começa a enviar as informações lidas pelos sensores de corrente elétrica no tópico especificado, o que se repete até ser desligado ou desconectado da rede.

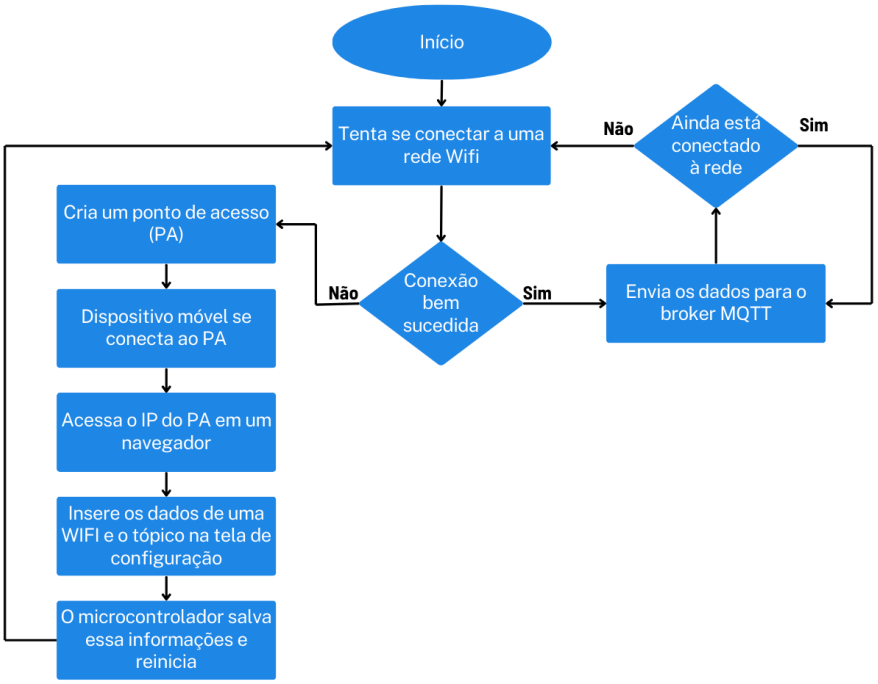

FIGURA 3. Fluxo de conexão do microcontrolador.

O sistema *web* para acessar os dados contém uma visualização por meio de gráfico, os valores da corrente elétrica são coletados e junto é salvo a hora que eles foram gerados, com isso é possível traçar um gráfico com os valores utilizados no eixo Y, ao longo da data e hora, utilizados no eixo X. Dessa forma fica fácil a visualização e entendimento do uso para analisar horários de pico, por exemplo, que é apresentada pela figura 4.

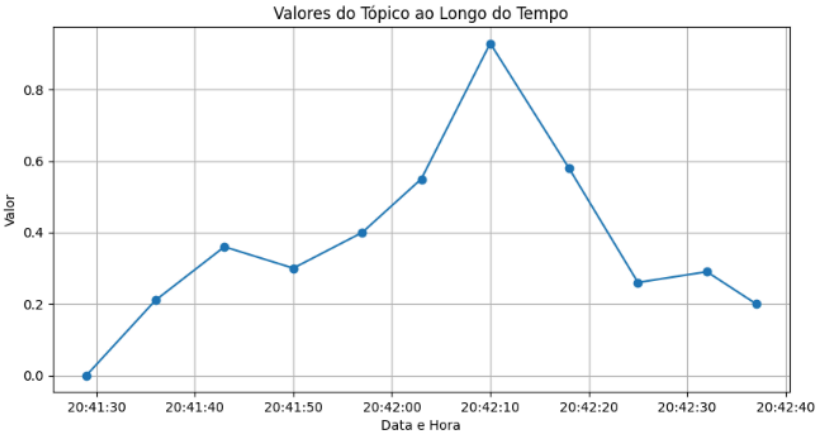

FIGURA 4. Gráficos dos valores ao longo do tempo.

CONCLUSÕES PARCIAIS

Foram realizados diversos testes básicos iniciais para fazer a análise do comportamento dos componentes de *hardware* dispostos, certificando que seriam viáveis para a implementação. Também foram realizados testes com linguagens de programação diferentes e foram verificadas as mais adequadas para o nosso caso testando suas capacidades, limitações e facilidade de implementação.

Após os testes iniciais, foram realizadas diversas etapas essenciais para a elaboração do projeto até o momento, incluindo a junção dos componentes para o teste de utilização do dispositivo como um todo, junto a implementação dos *softwares* selecionados. Um dos *softwares* desenvolvidos foi para o microcontrolador ESP8266 se comunicar em rede e enviar dados de testes para um *broker* MQTT, uma vez visto viável essa comunicação, foi desenvolvido outro *software*, dessa vez utilizando Python junto com Flask e bibliotecas adequadas, para a coleta, armazenamento e organização desses dados para sua visualização.

Para as próximas fases do projeto será acoplado o componente eletrônico para realizar a medidas de corrente elétrica e utilizar dados reais para a visualização e, posteriormente, sua análise utilizando redes neurais artificiais, realizando testes utilizando diferentes técnicas disponíveis, para assim ser possível fazer a detecção de anomalias desses dados analisados.

Os próximos passos para esse projeto incluem o aprimoramento dos *softwares* desenvolvidos, utilização do *hardware* de medição de corrente elétrica e utilização de inteligência artificial para detecção de anomalias.

CONTRIBUIÇÕES DOS AUTORES

Lorenzo Chiarello Weffort contribuiu com a escrita, desenvolvimento do trabalho e realização dos testes. André Luís Olivete auxiliou na revisão, orientou o desenvolvimento e forneceu suporte técnico ao longo do projeto.

Ambos os autores contribuíram para a revisão final e aprovaram a versão submetida.

AGRADECIMENTOS

Agradecemos ao Instituto Federal de São Paulo, Câmpus Presidente Epitácio, por fornecer um ambiente propício para o desenvolvimento acadêmico e prático.

Agradecemos ao professor André Luis Olivete por sua orientação e contribuições ao longo do processo.

REFERÊNCIAS

CAMPOS, Roger. **Previsão de séries temporais com aplicações a séries de consumo de energia elétrica**, 2008. Disponível em: https://repositorio.ufmg.br/handle/1843/BUOS-8CTETD. Acesso em: 24 mai. 2024.

IBM. **O que é uma rede neural**, 2024. Disponível em: https://www.ibm.com/br-pt/topics/neural-networks, Acesso em: 23 mai. 2024.

LIMA, Jully. **Validação do sensor de corrente SCT-013**, 2017. Disponível em: https://repositorio.ufopa.edu.br/jspui/handle/123456789/1440. Acesso em: 05 jul. 2024.

MARTINS, Bruno. **Detecção de anomalias em séries temporais variáveis com LSTM-TNNS**, 2020. Disponível em: https://www.hitecnologia.com.br/o-que-e-o-protocolo-mqtt/. Acesso em: 24 mai. 2024

PEREZ, Cristiano. **Arquiteturas para aplicações realtime utilizando MQTT**, 2017. Disponível em: <https://blog.elo7.dev/arquiteturas-para-aplica%C3%A7%C3%B5es-realtime-utilizando-mqtt/>. Acesso em: 6 jun. 2024.