# Interface para monitoramento elétrico com Mult-K 05 no contexto de Smart Campus

**Fabiano Lucas Jung[1], Dr. Marcelo Trindade Rebonatto[1]**

[1]Curso de Engenharia de Computação– Universidade de Passo Fundo (UPF)
Av. Brasil Leste, 285 - São José, Passo Fundo - RS, 99052-900

{158796,rebonatto}@upf.br

***Abstract.*** *This work describes the project and development of a monitoring electric system using a commercial electrical data meter. Tools used for theoretical foundation and construction of each part of the application are described, as well as the way in which each part was made. As result, was obtained a monitoring platform, with data acquisition time starting from 10 seconds and a web visualization interface.*

***Resumo.*** *Este trabalho descreve o projeto e desenvolvimento de um sistema de monitoramento elétrico a partir de um medidor de grandezas elétricas comercial. São descritas ferramentas utilizadas para embasamento teórico e construção de cada parte da aplicação, assim como a forma em que cada parte foi feita. Como resultados, obteve-se uma plataforma de monitoramento, com tempos de coleta a partir de 10 segundos e interface web de visualização.*

## 1. Introdução

Elementos de *Smart City[Barth de Lima et al. 2020; Cunha et al. 2016]*, *Smart Building[Snoonian 2003]* e *Smart Campus[Min-Allah and Alrashed 2020; Rebonatto et al. 2020]* estão cada vez mais presentes no cotidiano. Esses temas buscam de alguma forma melhorar o que já existe, ou criar funcionalidades inovadoras que se estabeleçam e sirvam as pessoas em seu respectivo ambiente. No caso de *Smart City*, nas cidades, no caso de *Smart Campus*, em um campus universitário ou, no caso de *Smart Building*, em prédios.

Um tipo de recurso inteligente que pode ser inserido em todos os escopos citados é uma gestão do sistema elétrico eficiente, ou, em uma descrição mais próxima dos assuntos abordados, *Smart Grid[Liu et al. 2015; Morvaj et al. 2011]*. Para esse ser desenvolvido é preciso de uma análise das medidas elétricas do local, o que traz a necessidade de um equipamento medidor com armazenamento das informações. Em prédios é comum existir um medidor geral, mas dificilmente ele armazena os dados, o que traz a necessidade de um sistema paralelo ao medidor para fazer essa função. Em residências, esse tipo de medidor normalmente não é utilizado, pois são de alto custo e fornecem uma quantidade de dados onde, muitas vezes, a maioria das medidas não é utilizada. Nesse caso, uma das alternativas seria o desenvolvimento de um sistema medidor[Mella 2019] em conjunto com um serviço que processe e mostre esses dados em um ambiente de cidades inteligentes[Rizzardi 2020]. Dentro dessa mesma perspectiva, foi constatado, em uma conversa com responsável pelos sistemas elétricos da UPF, a necessidade de monitoramento de prédios individuais da universidade através de um acesso externo aos medidores.

Tendo esses problemas em vista, este trabalho tem por objetivo desenvolver um

sistema que lê dados de um medidor comercial de um prédio, armazena em um banco de dados e mostra os dados em uma aplicação web com *dashboard*.

Na sequência do texto é feita uma abordagem sobre medidas elétricas, seguindo para as tecnologias utilizadas no desenvolvimento, é então mostrada a arquitetura do projeto, solução implementada, conclusões e considerações finais.

## 2. Conceitos relacionados ao monitoramento elétrico

Para se efetuar um monitoramento elétrico é necessário entender o que cada medida representa e como elas se relacionam em instalações elétricas de corrente contínua e corrente alternada. Por isso, essa seção descreve de forma geral o que cada grandeza utilizado no trabalho representa.

### 2.1. Tensão Elétrica

Também referida como diferença de potencial, é uma força exercida para movimentar elétrons e gerar com isso uma corrente elétrica. Em um circuito de corrente contínua pode ser medida instantaneamente, já que não ocorre grande variação da mesma em relação ao tempo, dessa forma, segundo a primeira lei de Ohm (equação 1). Em um circuito de corrente alternada ela varia invertendo a polaridade em relação ao tempo, e por isso utiliza-se a medida de valor eficaz ou tensão raiz quadrada média (*Root Mean Square* - RMS) para se obter dados instantâneos do sistema.

$$V = R \cdot I \tag{1}$$

A Tensão RMS [“Glossary Definition for vrms” 2021; Sadiku and Alexander 2013]é uma medida utilizada principalmente em sistemas de corrente alternada, onde a tensão pode ser representada por uma onda senoidal. A tensão RMS representa o valor eficaz de tensão e é calculado, em ondas senoidais, a partir da medição de valores de pico da onda (Vp). A equação pode ser representada pela equação 2. As medições de tensão utilizadas no trabalho são medições RMS.

$$VRMS = \frac{Vp}{\sqrt{2}} \tag{2}$$

### 2.2. Corrente Elétrica

Medição que representa a movimentação de elétrons em um circuito elétrico, um fluxo de cargas elétricas[Sadiku and Alexander 2013]. É diretamente afetada pela tensão elétrica (V) e resistência elétrica (R) e pode ser medida instantaneamente em sistemas de corrente contínua, pode ser calculada pela equação 3. Porém, assim como a tensão, em sistemas de corrente alternada também existe a medida de valor eficaz, a Corrente RMS.

$$I = \frac{V}{R} \tag{3}$$

A Corrente RMS possui as mesmas características da tensão RMS, porém, é calculada a partir da corrente de pico da onda senoidal (Ip), e não da tensão de pico. Pode ser expressa pela equação 4.

$$IRMS = \frac{Ip}{\sqrt{2}} \quad (4)$$

Em circuitos de corrente contínua não ocorre inversão do sentido da corrente elétrica ou inversão da polaridade da tensão de alimentação. Este circuitos são geralmente encontrados em eletrônicos, principalmente digitais.

Em circuitos de corrente alternada ocorre a inversão do sentido da corrente elétrica. A tensão varia em um formato de onda, normalmente, senoidal em uma frequência geralmente fixa. A transmissão da energia elétrica residencial e industrial é feita nesse modelo e as medições realizadas no projeto se enquadram nesta classificação.

## 2.3. Resistência Elétrica

É uma medida que quantifica uma restrição ao movimento dos elétrons, limitando a corrente elétrica. Em circuitos de corrente contínua possui valor fixo, uma vez que a resistência (R) é linear. Nesse caso o valor depende da constante do material (p), das medidas de comprimento (l) e área da sessão transversal do componente resistivo (A) [Sadiku and Alexander 2013]. Com esses valores é gerada a equação 5.

$$R = p \cdot \frac{l}{A} \quad (5)$$

Em circuitos em corrente alternada a resistência possui mais elementos, pois além da resistência imposta por componentes resistivos lineares, ainda existe a resistência não linear causada pelo efeito de capacitores (capacitância) e indutores (indutância). Este efeito é chamado de Reatância (X) e é dividido em Reatância Capacitiva e Reatância Indutiva. Convencionalmente o total das resistências de um sistema em corrente alternada chama-se Impedância (Z), como mostra a equação 6.

$$Z = R + jX \quad (6)$$

A Reatância Capacitiva representa a impedância gerada pelo campo elétrico de capacitores. Quanto menor a frequência, maior a reatância capacitiva. A Reatância Indutiva representa a impedância gerada pelo campo magnético de indutores. Quanto maior a frequência, maior a reatância indutiva.

## 2.4. Potência Elétrica

Relaciona a corrente elétrica e a tensão elétrica em uma medida que representa a velocidade com que se consome ou se absorve energia[Sadiku and Alexander 2013]. Em corrente contínua o valor é linear e é único, mas em corrente alternada além de não ser linear ela possui três classificações: a potência ativa, potência reativa e potência aparente.

A Potência Ativa representa a parte da potência elétrica que efetivamente é convertida em trabalho no circuito elétrico. Em um circuito puramente resistivo (sem capacitores e indutores) ela é máxima enquanto que a potência reativa é zero.

A Potência Reativa é a gerada por indutores e capacitores no circuito. Não gera trabalho no circuito e geralmente se tenta reduzi-la. Em circuitos indutivos ela é positiva e em circuitos capacitivos ela é negativa.

A Potência Aparente é a total do circuito, resultado da combinação da potência ativa e da potência reativa por meio de uma relação chamada de triângulo de potência (Figura 1).

Na figura, a potência aparente é representada por S, a potência reativa é representada por Q e a potência ativa é representada por P, dessa forma gerando a equação 7.

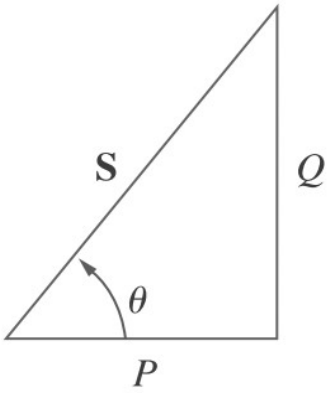


**Figura 1: Triângulo de Potência [Sadiku and Alexander 2013]**

$$|S|=\sqrt{P^2+Q^2} \tag{7}$$

O Fator de Potência (Fp) é a fração entre a potência ativa e a potência aparente. Esta medida representa o quão eficientemente a potência é utilizada no circuito. Utilizando a Figura 1 como referência, pode ser representado como a equação 8.

$$Fp=\frac{P}{S} \tag{8}$$

## 2.5. Outras medidas elétricas utilizadas no trabalho

No trabalho de monitoramento elétrico de prédios em ambientes de S*mart Campus*, outras medidas específicas podem ser monitoradas. Entre elas se encontram as demandas, as energias e o valor de THD.

A Demanda Ativa é a média da potência ativa por intervalo de tempo. A Demanda Aparente é a média da potência aparente por intervalo de tempo[Kron 2018].

A Energia Ativa Positiva é o consumo de energia acumulado. A energia gasta pela potência ativa em um determinado período de tempo. A Energia Reativa Positiva é o consumo de energia gerado pela parte indutiva da potência reativa em um determinado período de tempo. A Energia Reativa Negativa é o consumo de energia gerado pela parte capacitiva da potência reativa em um determinado período de tempo.

A distorção harmônica total representa o quão distorcida a onda está em relação a frequência fundamental. O medidor calcula com base na equação 9, onde V1 é a magnitude da fundamental e Vi é a magnitude da harmônica de ordem i [Kron 2018].

$$THD = 100 \times \frac{\sqrt{\sum_{i=2}^{31} Vi^2}}{V1} \quad (9)$$

## 3. Tecnologias aplicadas ao desenvolvimento desse projeto

Para a realização do projeto uma abordagem de tópicos relacionados se faz necessária, pois implicam diretamente em conceitos utilizados no desenvolvimento do mesmo. Foram utilizados fundamentos de Internet das Coisas (*Internet of Things* - IoT), de Cidades Inteligentes, de NodeJS, do medidor utilizado, do protocolo Modbus-RTU e do funcionamento da serial RS485.

### 3.1. IoT

A Internet das Coisas vem se tornado um termo comum em anos recentes e é um dos principais motivos que causam o crescente número de dados na Internet, o *BigData[Oussous et al. 2018]*. O termo IoT pode ser dividido em três fragmentos principais[BNDES 2018]C: o recebimento ou envio de dados através de um dispositivo de IoT, uma conexão de rede externa ao sensor e o processamento autônomo desses dados para a utilização final.

Uma das ideias quando se fala em internet das coisas é a automação residencial, pois em uma casa há diversos elementos que podem ser automatizados visando a melhora da qualidade de vida e economia de energia. Pode-se notificar o proprietário quando algum alimento está próximo de acabar na geladeira, ligar/desligar o ar-condicionado dependendo da localização do celular do dono, desligar luzes ou equipamentos eletrônicos quando não estão sendo usados por algum tempo, monitoramento do consumo elétrico, ou ainda gerenciar a casa utilizando um aplicativo de celular[Sales Rocha and Anhesine 2020; Wanzeler et al. 2016]. Enfim, basta ideias para uma potencial implementação de IoT. Mas existem muitas outras coisas que podem ser automatizadas em diversos setores da indústria, transporte, medicina, turismo, muitas delas utilizando *Radio Frequency Identification* (RFID) ou *Near Field Communication* (NFC) para substituir o papel[Atzori et al. 2010]. Dessas podemos citar a utilização do RFID para fichas médicas, ou ainda para peças em uma linha de produção industrial onde a leitura resulta em um evento com registro em banco de dados, NFC em mapas turísticos para direcionar para sites de estabelecimentos como hotéis, lojas.

Outros tipos aplicações IoT são as que utilizam uma variedade de sensores a fim de otimizar um sistema. Um paciente hospitalar que não necessite estar acamado, por exemplo, poderia ter um conjunto de sensores menores que enviassem dados em tempo real para um sistema onde o médico/enfermeira possuem acesso, assim tendo maior cuidado ao paciente. Uma rodovia poderia ter sensores para medir a situação do tráfego e o mesmo ser disponibilizado para acesso em um serviço web onde aplicativos de GPS possam otimizar as rotas dos usuários a fim de evitar engarrafamentos, o que a longo prazo poderia baixar os custos de transporte[Atzori et al. 2010; MINISTÉRIO DA ECONOMIA 2020].

Este projeto, assim como algumas das aplicações citadas, busca fazer uma captação de dados para uma futura otimização. Ele utiliza um dispositivo IoT para fazer

a leitura de um medidor e a transmite para um serviço em rede para então ser consumido por uma aplicação de visualização.

### 3.2. Cidades Inteligentes

O termo Smart Cities[Barth de Lima et al. 2020; Cunha et al. 2016] cresceu junto com o IoT, pois cidades inteligentes podem utilizar elementos da internet das coisas para solucionar problemas em um escopo de cidade. Também existem variações a cerca do tema, como *Smart Campus[Min-Allah and Alrashed 2020; Rebonatto et al. 2020], Smart Buildings[Snoonian 2003], Smart Grids* onde cada uma possui o IoT integrado para diferentes propósitos em estruturas diferentes, mas com um princípio igual, o melhoramento de um ambiente, e/ou sistema. Nesse sentido, o projeto se enquadra dentro dos todos subtemas citados, pois é desenvolvido em uma universidade, ao mesmo tempo dentro de prédios onde o sistema extrai as informações de um sistema elétrico que pode ser utilizado para a aplicação de um *Smart Grid[Liu et al. 2015; Morvaj et al. 2011]*.

O termo cidades inteligentes proporciona discussões para o melhoramento de setores de cidades[BNDES 2018], dentre eles principalmente mobilidade, segurança e eficiência energética. Todos eles de alguma forma impactam no desenvolvimento da cidade, do ambiente e da qualidade de vida das pessoas que nela vivem. A mobilidade urbana afeta no tempo de deslocamento de pessoas/veículos na cidade, o que impacta em custos de transporte, consumo de combustível e consequentemente aumento de gases poluentes no ambiente. Uma das formas que a mobilidade poderia ser melhorada seria aplicando semáforos inteligentes[Osorio-Comparán et al. 2017] que monitoram a situação do tráfego atual para a tomada de decisões, assim evitando veículos parados quando não há tráfego concorrente na intersecção da via. Ainda é possível verificar pelo som, ou algum outro tipo de informação, quando um veículo de emergência (ambulância, bombeiros) está próximo, dessa forma podendo liberar o tráfego ou tomar a decisão correta. Na segurança é possível utilizar sensores e câmeras para um monitoramento centralizado da cidade, ainda sendo possível integrar algoritmos de aprendizado de máquina a fim de detectar crimes e atividades suspeitas. Já a eficiência energética pode ser obtida através de monitoramento de dados da rede elétrica em conjunto com a aplicação de conceitos de *Smart Grid*.

Dentre os conceitos descritos, o que mais se relaciona com *Smart Campus* e *Smart Building* é a gestão energética. O trabalho vigente direciona justamente nessa área, pois é focado ao monitoramento da rede elétrica, no caso, a prédios de um campus de universitário.

### 3.3. NodeJS

NodeJS[“About | Node.js” 2021] é um software de código aberto que permite a execução de códigos JavaScript[“JavaScript | MDN” 2021] fora do navegador. Ele é baseado em eventos assíncronos em *single thread* onde a cada ocorrência de evento uma função de *callback* é executada. O NodeJS utiliza o gerenciador de pacotes Node Package Manager (NPM)[“npm” [S.d.]] que contém ferramentas para a criação de vários tipos de aplicações.

NodeJS foi utilizado no projeto para o desenvolvimento da API e aplicação de visualização.

### 3.4. Medidor Multi-K05

O medidor Kron Mult-K 05 é um instrumento microprocessado desenvolvido para fazer leituras de parâmetros elétricos[Kron 2020a] em terminais de corrente alternada de alta, média ou baixa tensão.

O aparelho pode medir sistemas monofásicos, bifásicos ou trifásicos e as medições são feitas através de sinais de tensão e corrente que seriam dados 'puros', utilizados para calcular as demais grandezas através de fórmulas matemáticas específicas para cada tipo de medição[Kron 2018].

No equipamento, são apresentadas duas formas principais para os dados serem visualizados, uma delas é diretamente pelo *display* do equipamento, a outra é pela comunicação serial que é, por padrão, do tipo RS-485. A interface serial do medidor possui algumas características adicionais configuráveis. O protocolo padrão do equipamento[Kron 2020b] é o Modbus-RTU. O *baudrate* (taxa de transmissão) pode ser configurado nos modos 9600, 19200, 38400 ou 57600 bps. Pode ser configurado o bit paridade para nenhum, ímpar ou par. Um ou dois bits de parada. Endereços de 1 até 247. O número de bits de início (*start bit*) é fixo, assim como o número de bits de dados que é fixo em oito (8).

### 3.5. Padrão RS-485

O modelo de comunicação serial RS-485[Kugelstadt 2008] foi desenvolvido em 1983 como uma forma de melhorar as limitações da RS-232[Maxim Integrated Products 2021]. O padrão desenvolvido suporta comunicação de distâncias de até 1200m com um alto nível de imunidade a ruídos utilizando a diferença de potencial entre dois cabos para fazer a comunicação, desse modo, não precisando obrigatoriamente conectar os GNDs dos equipamentos na rede. A velocidade dessa rede depende principalmente da distância entre os dispositivos conectados podendo variar de 10 Mbps a distâncias de aproximadamente 10m até 100kbps a distâncias de até 1200m. O número máximo de dispositivos conectados na rede em distâncias de aproximadamente 1000m sem uso de repetidores é de 32, porém em distâncias menores podem-se conectar até 256 (ou ainda mais em alguns casos) em redes dos tipos *Full-Duplex* (4 fios) ou *Half-Duplex* (2 fios). Na indústria o modo mais comumente utilizado em redes com serial RS-485 é o *Half-Duplex*, modo que o equipamento estudado também utiliza.

A RS-485 possui uma amplitude de tensão máxima de -7V a +12V, o que faz necessário o uso de um conversor para ligar a uma placa de prototipação. No mercado existem vários tipos de conversores com diferentes preços e diferentes comunicações alvo como RS-232, *USB*, RS-422[Soltero et al. 2010], Ethernet, entre outros. O cabo de comunicação a ser utilizado possui algumas características recomentadas para o uso: cabo par trançado com bitola do tipo AWG-24 e impedância característica de 120Ω. A recomendação de cabo provavelmente é para aplicações mais robustas e que exigem mais da rede de comunicação, por isso a fim de reduzir o custo do projeto, serão testados cabos mais simples.

### 3.6. Protocolo Modbus-RTU

O protocolo *Modbus Remote Terminal Unit* (Modbus-RTU)[Modicon 1999] estabelece uma formato onde os endereços e os valores são utilizados na forma binária para se comunicar entre dispositivos dentro de um padrão de funcionamento físico. O sistema físico pode ser de vários formatos, os mais comuns onde esse protocolo é utilizado são RS-232, RS-485, RS-422. O Modbus-RTU estabelece um *frame* de bits onde cada byte

ou grupos de bytes são representados como um parâmetro para a comunicação. O protocolo utiliza a ideia de *master-slave*, onde um *master*, no caso a placa de prototipação, faz solicitações ou comandos para o *slave*, no caso o medidor, que apenas responde ou atua conforme as mensagens/comandos recebidas.

Uma mensagem do *master* que solicita dados para leitura, no modelo padrão, se denomina *Read Input Registers*. Existem variações desse modelo conforme a aplicação, mas em geral possui os campos: endereço do *slave*, código da função, endereço inicial de registro, quantidade de registros a serem lidos e o CRC (Figura 2).

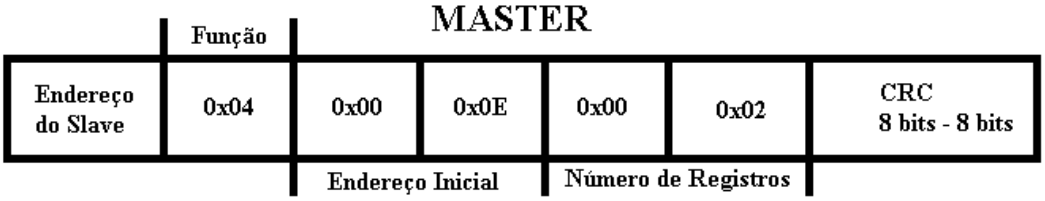


**Figura 2: Mensagem enviada para o medidor**

O endereço do *slave* é um valor configurável que possui tamanho de 1 byte, o código da função é um valor de 1 byte que indica o que o *slave* precisa fazer, o endereço inicial possui 2 bytes e indica qual o registro inicial do *slave* a ser lido, quantia de registros possui 2 bytes indica a quantidade de leituras a serem feitas e o CRC possui 2 bytes e é um valor calculado com base no conteúdo da mensagem para ser comparado no destino e verificar se a mensagem não foi modificada. Esse formato faz parte dos *frames* do medidor, mas ele também utiliza de formatos próprios desenvolvidos pelo fabricante.

Já a resposta enviada para o *master* apresenta o seguinte formato: endereço do *slave*, código da função, número de bytes de dados, seguindo dos bytes de dados conforme a contagem anterior, e por final o CRC (Figura 3).

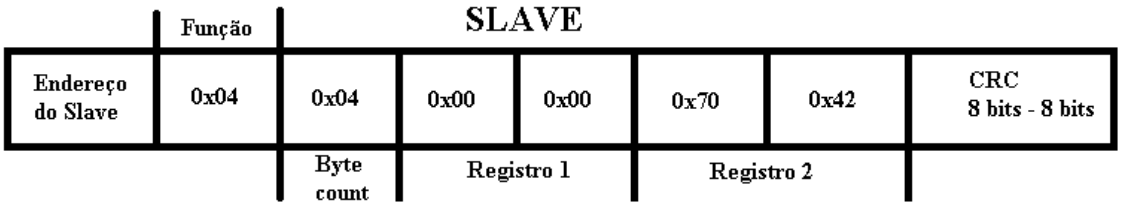


**Figura 3: Mensagem recebida do medidor**

O contador de bytes de dados possui 1 byte e o restante possui o mesmo tamanho dos campos anteriores da mensagem do *master*. É por esse formato que os dados de medição são lidos pela placa de prototipação.

Além do apresentado também possuem outros comandos padronizados pelo fabricante da Modbus-RTU, porém não serão utilizados no projeto. Algumas dessas funções no sistema implementado pela Kron estão descritas abaixo[Kron 2020b]:

- *Read Input Status:* lê o status das entradas e saídas digitais.
- *Read Holding Register:* lê registros de configuração.

- *Force Single Coil:* faz o aparelho executar funções específicas como, por exemplo, reiniciar o dispositivo.
- *Preset Single Register:* envia parâmetros para um único registro de configuração.
- *Read Exception Status:* verifica se existe algum problema de funcionamento.
- *Preset Multiple Register:* envia parâmetros para vários registro de configuração.
- *Report Slave ID:* lê o identificador da rede Modbus do dispositivo.

Nos exemplos passados, cada nome corresponde a um número de função específico, que pode ser consultado/acionado dependendo das necessidades de um desenvolvedor.

### 3.7. Plataformas de hardware para IoT

A plataforma de prototipação a ser escolhida precisa ser capaz de satisfazer as condições de comunicação do equipamento alvo. Com essas condições aprovadas ele precisa ter uma interface serial integrável com conversor de RS-485, para então poder ser programado o *firmware* para a comunicação via protocolo Modbus-RTU e posteriormente enviar os dados para um ambiente de cidades inteligentes.

A comunicação entre computador e microcontrolador poderá ser desenvolvida por dois meios principais no quesito simplicidade, por *USB* ou *Wifi*. Por *USB* será necessária uma ligação física com o computador, onde os dados processados serão enviados via serial, serão lidos no computador e posteriormente enviados para uma plataforma web. Por *Wifi* não será necessário a ligação física entre computador e microcontrolador, o que torna o sistema mais prático e fácil para integrar em projetos com múltiplas leituras de dados, porém o custo de desenvolvimento se torna mais alto. O uso de conexão *Ethernet* foi descartado pelo motivo de ter um custo similar ao do *Wifi*, precisar de ligação física e dificilmente algum kit de prototipação possuir uma porta do tipo integrada.

Visto isso, os microcontroladores prováveis para o uso são Arduino UNO[“What is Arduino? | Arduino” 2021], o qual pode ser adicionado um módulo *Wifi* caso não seja utilizado a serial via USB no computador. Algum modelo de Raspberry Pi[“Raspberry Pi Foundation - About Us” 2021] que possui um hardware mais potente que o do Arduino e já é, na maioria dos modelos, integrado com *Wifi*. Um ESP-32[Espressif Systems 2016], que também possui um hardware mais potente que o do Arduino e é integrado com módulos *built in* de *Bluetooth* e *Wifi*. Existem mais microcontroladores que satisfazem as condições de comunicação, mas os considerados foram selecionados com base na simplicidade de programação e facilidade de obtenção. Dos modelos descritos o que foi mais utilizado em outros projetos desenvolvidos e é mais fácil de ser obtido é o Arduino UNO, porém não se descarta a substituição do mesmo no decorrer do projeto.

## 4. Arquitetura do monitoramento elétrico de prédios no contexto de Smart Campus

Os elementos do projeto consistem de 5 elementos principais, um conversor RS485-TTL, uma placa de desenvolvimento rápido para IoT, uma API para interfacear os dados, um Banco de Dados (BD), e uma aplicação de visualização dos dados coletados. A Figura 4 contém a arquitetura do projeto.

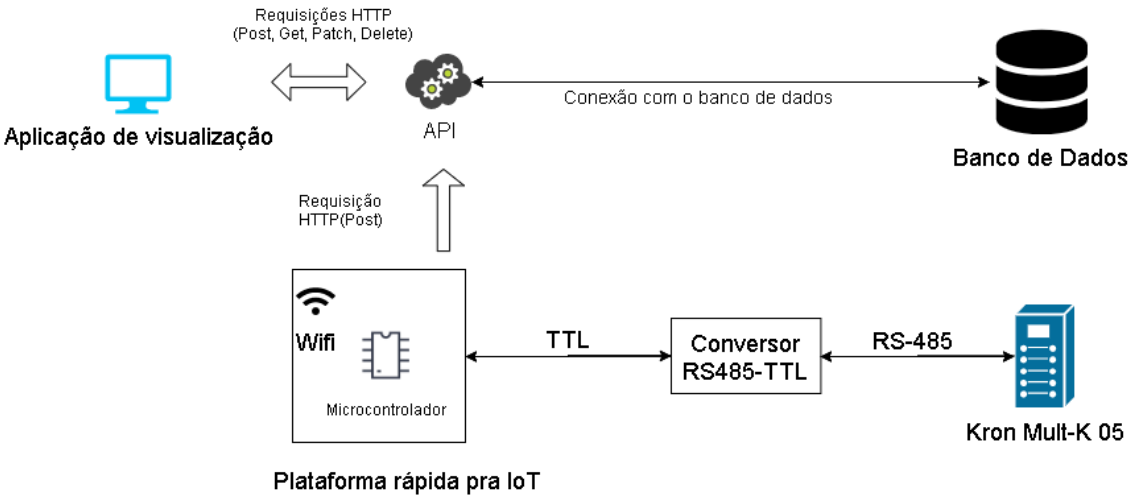


**Figura 4: Arquitetura do Projeto**

A placa de prototipação contém um microcontrolador, onde poderá ser executado um *firmware* de comunicação com o equipamento, através do conversor RS485-TTL. A API será um dos elementos chave, recebendo os dados coletados pelo microcontrolador e enviando ao BD. Além disso, a API vai fornecer os acesso aos dados à aplicação, servindo como meio de acesso ao BD.

## 5. Implementação da solução proposta

O desenvolvimento da solução seguiu na ordem de programação de baixo nível para alto nível. Primeiramente foi trabalhado o *firmware* para a comunicação com o medidor, seguindo para a transmissão de dados pela rede *WIFI*, o desenvolvimento da API para receber os dados da placa de prototipação, consequentemente conectando a API com um banco de dados, e após, o desenvolvimento da aplicação para mostrar os valores obtidos.

### 5.1. Desenvolvimento da comunicação com o equipamento

Inicialmente foi desenvolvido um código simplificado para fazer testes utilizando simuladores do protocolo Modbus RTU, pois em meio a pandemia existiam restrições de acesso e as atividades não essenciais estavam sendo realizadas de maneira remota. O código foi feito basicamente para testar essa comunicação a fim de tentar entendê-la, dessa forma dando um referencial didático para o desenvolvimento presencial diretamente via hardware.

Quando as restrições da pandemia se abrandaram, foi feita uma solicitação de retirada de um dos equipamentos medidores para a realização dos testes em uma bancada. O equipamento foi ligado a uma tomada apenas para a sua alimentação, sem nenhuma entrada para medição, dessa forma servindo para testar a comunicação com o protocolo. Os fios internos de um cabo *ethernet* foram utilizados para conectar com a serial do medidor, desse modo foram conectados 3 fios, um no campo Data +, outro no Data – e outro no GND. Em redes pequenas o campo GND não se faz necessário na RS 485, por isso acabou não sendo utilizado. Foi utilizado um Arduino UNO e se fez necessário um adaptador para entrada da serial do equipamento. O adaptador escolhido

foi o MAX 485[Maxim Integrated Products 2014], que converte o sinal em TTL, podendo então desenvolver a comunicação do *firmware* como uma serial comum da placa.

Na parte de software foi utilizada a biblioteca ModbusMaster para se comunicar via protocolo Modbus, o uso da mesma foi da seguinte forma. Primeiramente foi instanciado um objeto com o endereço do medidor e a serial da placa como parâmetro. Por meio desse objeto foi utilizada uma função para enviar uma solicitação de leitura, com parâmetros de endereço de registro e número de registros de dados, e outra, após a validação do resultado da operação, para extrair o conteúdo da resposta recebida. Após isso, dependendo do tipo do dado, a resposta foi convertida para ponto flutuante ou inteiro. Para a conversão para ponto flutuante foi implementada uma função utilizando como base as instruções disponibilizadas pelo fabricante do equipamento[Kron 2021].

Com a comunicação concluída ainda era preciso desenvolver a transmissão desses dados pela rede, porém ainda não tinha sido obtido um módulo E*thernet*, nem W*ifi*. Por isso e pela possibilidade da placa faltar capacidade na implementação completa, a placa foi alterada para uma ESP32.

Para a transmissão de dados foi utilizada a biblioteca Wifi para a conexão com a rede local e em sequência foi testada uma comunicação via Socket TCP com a mesma biblioteca e um simples servidor local criado usando a linguagem Python[“About Python™ | Python.org” 2021]. Feito isso, foi iniciada a implementação das requisições HTTP usando a biblioteca httpclient[“httpclient arduino - esp32” 2021]. Apenas operações POST foram utilizadas na implementação, onde a cada *loop* de leitura da serial os valores lidos eram agregados em uma variável json da biblioteca arduinojson[“ArduinoJson: Efficient JSON serialization for embedded C++” 2021] e ao fim das leituras a variável é transformada em *string* e enviada para a api conforme a frequência estabelecida.

### 5.2. Desenvolvimento da API

A tecnologia escolhida para o desenvolvimento da API foi Node.js. Mesmo com a utilização do Python na a criação de um servidor *socket* anteriormente, o Node.js foi escolhido pelo fato de ser uma ferramente já trabalhada em outros projetos, dessa forma podendo se utilizar dela de forma mais complexa e com maior facilidade.

Inicialmente foi desenvolvido um servidor local na porta 3000 (no final do projeto alterado para a 8080) para apenas operações de POST e GET apenas com amostra de valores no console. Os testes foram feitos utilizando o navegador, o software Postman[“Postman | The Collaboration Platform for API Development” 2021] e requisições vindas da ESP32. Nesse momento foram instalados pacotes auxiliares para o servidor: body-parser[“body-parser - npm” 2021], para separar a requisição em partes mais fáceis de serem acessadas, express[“express - npm” 2021], que fornece recursos para a criação do servidor, morgan[“morgan - npm” 2021], serve de debug para as requisições, nodemon[“nodemon - npm” 2021], para atualizar a API sem precisar reiniciá-la, mysql[“mysql - npm” 2021], para acessar o banco de dados, mysql2[“mysql2 - npm” 2021], outra opção de acesso ao banco de dados, mas focado em desempenho, cors[“cors - npm” 2021], para liberar uma restrição de acesso em navegadores. O mysql e o mysql2 foram instalados, mas o utilizado foi o mysql2. Todas foram instaladas utilizando o NPM.

O próximo passo foi a adição de um banco de dados no sistema. Antes de uma

real modelagem para o projeto, foram feitos testes em uma versão simples com uma única tabela e duas variáveis, fazendo tentativas de GET, POST, PATCH e DELETE. Quando efetivada essa comunicação, foi pensado como fazer a modelagem do banco de dados e as funções da API.

### 5.3. Dados coletados, tempos de amostragens e banco de dados

Para fazer a modelagem do banco de dados foi preciso primeiro definir a frequência de leitura dos dados e quais deles seriam de utilidade para diagnósticos da rede elétrica, para isso, foi feita uma reunião[“Reuniao de Orientação” 2021], no dia 27/04/21 às 15:30, com o responsável pelas instalações elétricas da UPF. Na reunião ficaram definidas frequências de leitura para cada um dos tipos de dados escolhidos, assim dividindo em três tabelas de valores para leitura, frequenciaalta, frequenciamedia e frequenciabaixa. O banco de dados (BD) escolhido para a modelagem foi o MySQL[“MySQL” 2021].

Na tabela frequenciaalta ficaram valores que necessitam de uma atualização constante. Inicialmente foi pensado em uma frequência de 1 Hertz, mas como isso geraria uma quantidade muito grande de dados para se manter em um BD e ficou estabelecido de manter as configurações padrões da serial do equipamento, a frequência foi diminuída para 0,1 Hertz. Importante ressaltar que, mesmo implementando a frequência de 1 hertz, muitas variações bruscas ainda passariam despercebidas, pois necessita-se de uma frequência de leitura bem maior que a capacidade de transmissão da serial para se conseguir captar todas as variações de uma rede elétrica de 60 Hertz. Os dados selecionados para essa tabela consistem em valores de tensão e corrente trifásicos, tensões de cada linha em relação ao neutro (tensão de fase), tensões entre linhas (tensão de linha), correntes em cada linha, demandas ativa e aparente, corrente de neutro e dois parâmetros operacionais, o número de série do equipamento e o erro da medição. Os valores são todos em ponto flutuante com exceção dos parâmeros operacionais.

Na tabela frequenciamedia ficaram valores que não variam de forma brusca a ponto de precisarem serem lidos em uma frequência alta. A frequência estabelecida para esses valores foi de 1 leitura a cada minuto. Os dados selecionados para essa tabela consistem em valores de potência ativa, reativa e aparente de cada linha e trifásico, fator de potência de cada linha e trifásico, Distorção Harmônica Total (*Total Harmonic Distortion* - THD) de tensão e corrente de cada linha, frequência, energias ativa e reativa separadas em positivas e negativas. Com exceção dos valores de THD que são inteiros, todos os valores são de ponto flutuante,

Na tabela frequenciabaixa ficaram valores que não mudam muito em relação ao tempo. A frequência de leitura ficou estabelecida em 1 leitura a cada hora. Nessa tabela ficaram valores máximo, a tensão máxima, corrente máxima e demandas ativa e aparente máximas. Todos os valores são em ponto flutuante.

Todas as tabelas possuem o número de série lido em conjunto com dois valores individuais criados, um contador para servir de chave primária e uma variável de tempo para cada leitura. A modelagem completa pode ser vista na Figura 5.

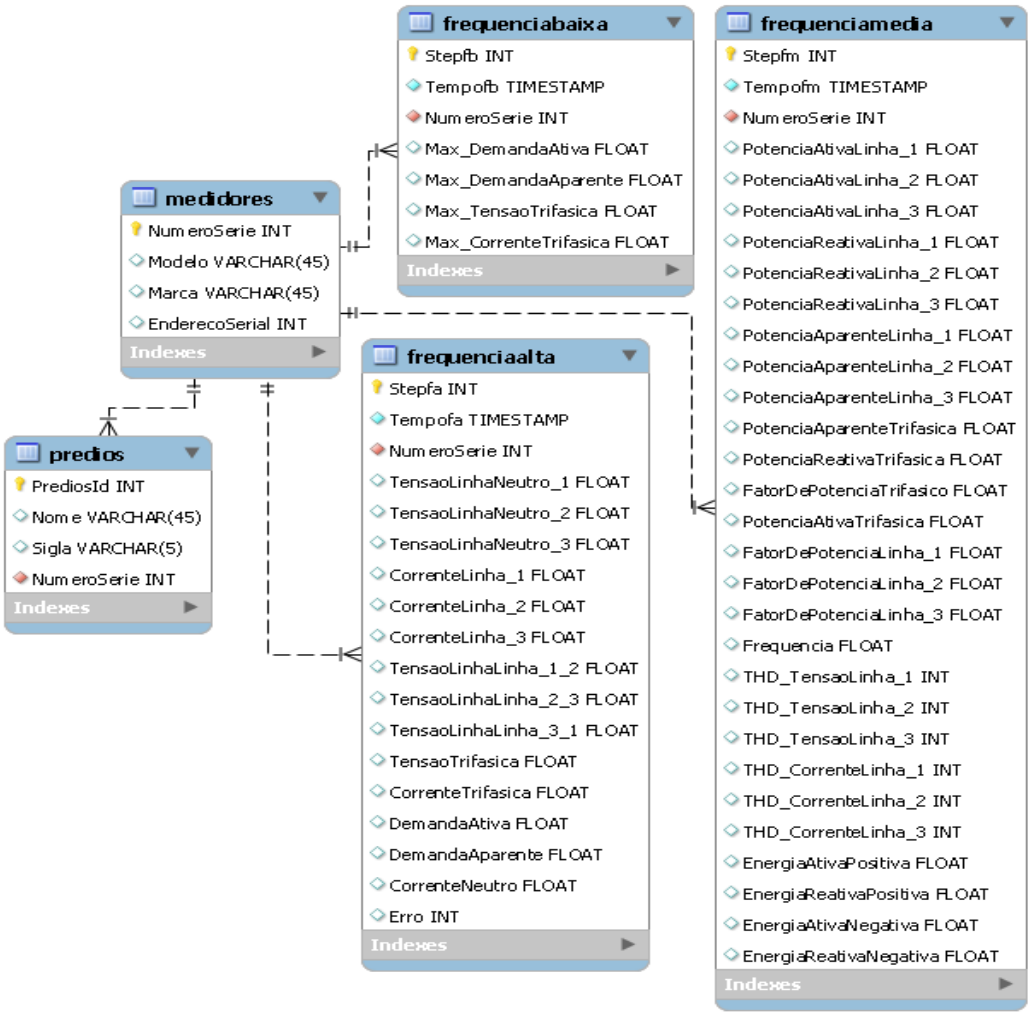


**Figura 5: Modelagem do banco de dados**

Junto das tabelas de valores de leitura foram criadas outras duas tabelas para registro de dados estruturais, a tabela predios e a tabela medidores. Na tabela predios ficaram os valores de identificação do prédio por nome, sigla, id e número de série do equipamento no local. Na tabela medidores ficaram dados do equipamento do qual virá as leituras, como número de série do medidor, marca, modelo e endereço configurado para a rede modbus. Todas as tabelas possuem chave estrangeira do número de série direcionado para o número de série da tabela medidores, campo que é chave primária da mesma.

## 5.4. Desenvolvimento da Aplicação de Visualização

Para fazer a aplicação visual inicialmente foi realizada uma tentativa de uso do NodeRed[“Node-RED” 2021]. O NodeRed oferece uma interface de desenvolvimento em blocos e pode servir para a criação de *dashboards*, motivo pelo qual tinha sido escolhido, porém acabou sendo substituído pelo ReactJs[“React – Uma biblioteca JavaScript para criar interfaces de usuário” 2021]. O motivo da troca foi a dificuldade de configurar alguns detalhes em um sistema de blocos, pois era preciso desenvolver telas de cadastro e visualização para vários tipos de dados. No Reactjs o desenvolvimento é

um pouco mais lento que no NodeRed, mas compensa pela quantidade de detalhes possíveis de se alterar.

A plataforma foi instalada utilizando Nodejs e o aplicativo foi criado utilizando o comando create-react-app[“Create-react-app” 2021]. Para o desenvolvimento do aplicativo foram utilizados pacotes do react em conjunto com o react-router-dom, para a navegação de telas, axios[“axios” 2021], para requisições HTTP, componentes da material-ui[“Material-UI: Um framework popular de React UI” 2021] para tabelas e ícones e pacotes da biblioteca recharts[“Recharts” 2021] para gráficos. O aplicativo foi dividido em seis telas, uma de home, duas de cadastro e três de visualização de medições. Cada tela possui duas partes fixas, uma barra lateral, com ícones para navegação entre as páginas, e uma barra superior, onde fica o título da aplicação.

A tela de home mostra uma tabela com a junção da tabela de medidores e de predios e possui um campo de texto para selecionar o número de série do medidor do qual se queira ver os dados. Se nada for digitado o sistema configura um número de série padrão.

As telas de cadastro permitem inserir, editar ou deletar valores das tabelas de medidores e predios. Cada uma delas fornece sua tabela de valores atuais no sistema e os campos específicos para sua configuração. A figura 6 contém uma visualização do cadastro da tabela de medidores.

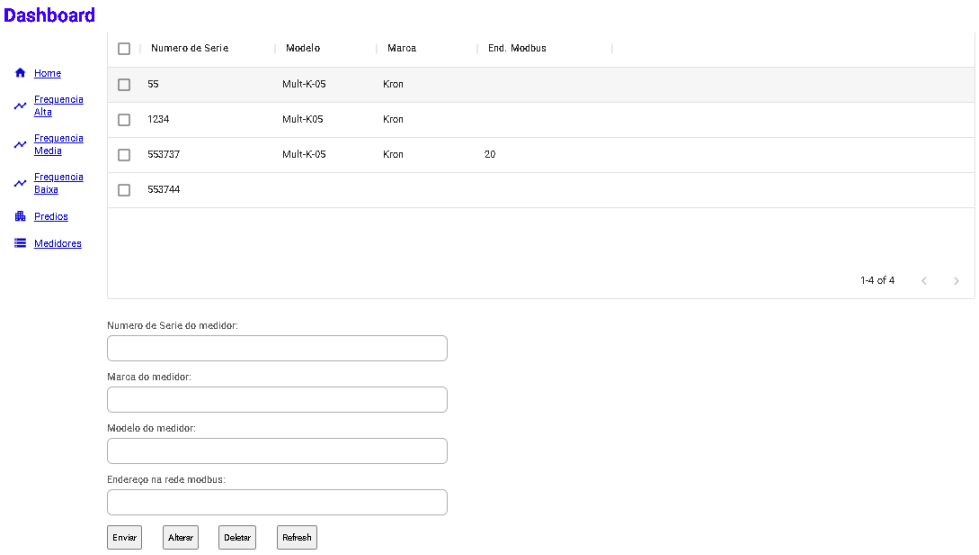


**Figura 6: Tela de Cadastro de Medidores**

As telas de visualização das medições são separadas para cada uma das tabelas: frequenciaalta, frequenciamedia e frequenciabaixa, e mostram os dados selecionados na página home. A visualização dos dados é feita através de gráficos de linha, onde valores únicos ou agrupados podem ser analisados em relação ao tempo com a interação do cursor do mouse. Cada uma das telas possuem taxa de atualização dos valores conforme a frequência de leitura, e acada atualização são visualizados os últimos 60 valores inseridos nas tabelas frequenciaalta e frequenciamedia, e 24 valores no caso da tabela de frequenciabaixa.

## 6. Resultados

Para testar o sistema funcionando, nos dias 22/06/21 e 23/06/21 no turno da tarde, foram feitos testes e coleta de amostras de dados reais de um prédio da UPF. O prédio de onde as amostras foram retiradas foi o Centro de Convivência. Os dados lidos pela plataforma de hardware montado (Figura 7) foram analisados e mostrados ao responsável pelas instalações elétricas da UPF, que confirmou de estarem coerentes e dentro das expectativas, dessa forma validando o sistema de coleta de dados.

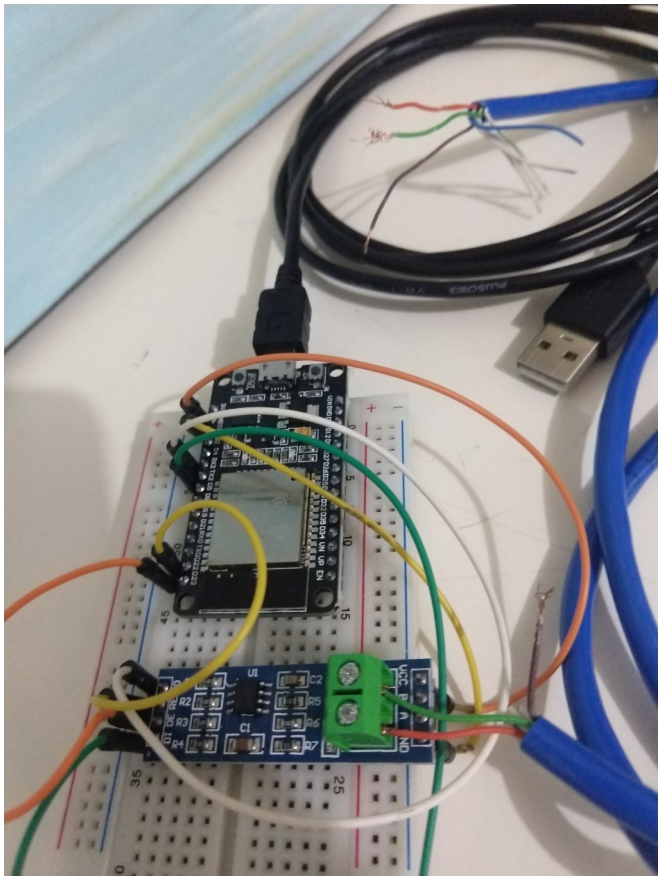

**Figura 7: Plataforma de hardware montada**

A parte do sistema pós leitura se mostrou funcional com atualizações dentro dos limites de tempo e com atualização dos dados na interface conforme atualização do banco de dados. As Figuras 8 e 9 contem algumas das amostras obtidas.

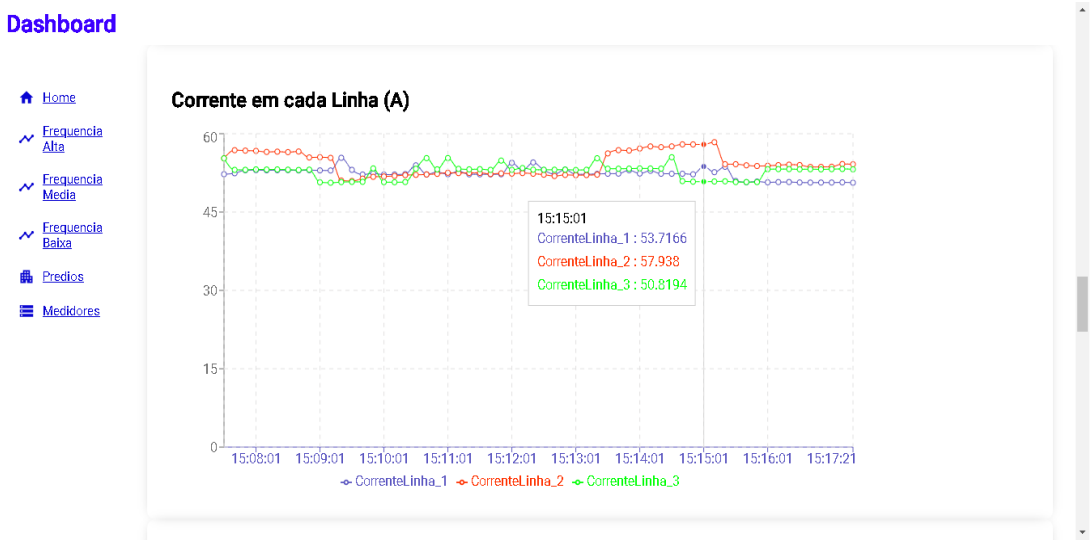


**Figura 8: Tela da aplicação mostrando corrente por linha**

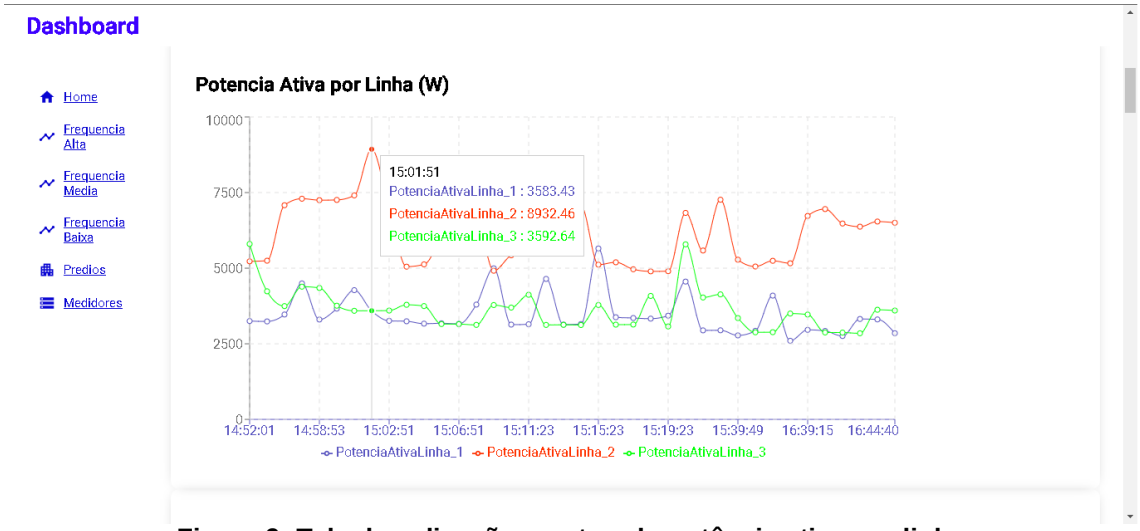


**Figura 9: Tela da aplicação mostrando potência ativa por linha**

Na figura 8, os dados são lidos a cada 10 segundos e na figura 9 a cada 1 minuto. Em ambas existe uma variação de periodicidade entre leituras por contra de testes estarem sendo feitos, o que gera desligamentos do sistema para atualização. Mas as leituras e respectivos tempos de cada estão corretos.

## Considerações Finais e trabalhos futuros

Este trabalho teve por objetivo desenvolver um protótipo de um sistema de monitoramento elétrico em um ambiente de Smart Campus a partir de um medidor de grandezas comercial. Foram realizados diversos estudos e testes, com diferentes tecnologias de hardware e de software, até encontrar uma solução de implementação que fosse adequada ao modelo projetado.

No desenvolvimento do trabalho percebeu-se uma grande variedade de ferramentas possíveis para se usar em sistemas inteligentes e muitas delas podendo ser integradas em conjunto para o desenvolvimento de projetos mais complexos. Muitas das

utilizadas podem ser incorporadas em aplicações inteligentes não voltadas especificadamente para monitoramento elétrico, mas com captura de dados de outros tipos de sensores, como de condições atmosféricas por exemplo.

Espera-se para trabalhos futuros a adição de processamento dos dados para diagnóstico da rede e redução de taxas da concessionaria de energia, em conjunto com uma otimização do sistema e aumento de frequência de leitura para alguns parâmetros.

## Referências


About | Node.js (2021). https://nodejs.org/en/about/, [accessed on Jul 3].

About npm | npm Docs (2021). https://docs.npmjs.com/about-npm, [accessed on Jun 3].

About Python™ | Python.org (2021). https://www.python.org/about/, [accessed on May 3].

ArduinoJson: Efficient JSON serialization for embedded C++ (2021). https://arduinojson.org/, [accessed on Apr 20].

Atzori, L., Iera, A. and Morabito, G. (2010). The Internet of Things: A survey. *Computer Networks*, v. 54, n. 15, p. 2787–2805.

axios (2021). https://github.com/axios/axios, [accessed on Jun 24].

Barth de Lima, M. R., Brino, G. and Baptista Cardia Neto, J. (18 dec 2020). CIDADES INTELIGENTES. *Revista Interface Tecnológica*, v. 17, n. 2, p. 180–192.

BNDES (2018). Cartilha de Cidades. https://www.bndes.gov.br/wps/wcm/connect/site/db27849e-dd37-4fbd-9046-6fda14b53ad0/produto-13-cartilha-das-cidades-publicada.pdf?MOD=AJPERES&CVID=m7tz8bf, [accessed on Apr 13].

body-parser - npm (2021). https://www.npmjs.com/package/body-parser, [accessed on Jul 13].

cors - npm (2021). https://www.npmjs.com/package/cors, [accessed on Jul 13].

Create-react-app (2021). https://github.com/facebook/create-react-app, [accessed on Jun 22].

Cunha, M. A., Przeybilovicz, E., Macaya, J. F. M. and Burgos, F. (2016). *Smart cities: Transformação Digital de cidades*. Programa Gestão Pública e Cidadania. v. 16

Espressif Systems (2016). ESP32 Datasheet. https://www.espressif.com/sites/default/files/documentation/esp32_technical_reference_manual_en.pdf, [accessed on May 3].

express - npm (2021). https://www.npmjs.com/package/express, [accessed on Jul 13].

Glossary Definition for vrms (2021). https://www.maximintegrated.com/en/glossary/definitions.mvp/term/vrms/gpk/1204, [accessed on Jun 13].

httpclient arduino - esp32 (2021). https://github.com/espressif/arduino-esp32/tree/master/libraries/HTTPClient, [accessed on Apr 3].

JavaScript | MDN (2021). https://developer.mozilla.org/pt-BR/docs/Web/JavaScript,

[accessed on Jun 3].

Kron (2018). Mult-K 05 e Mult-K 120 Medidores de Energia e Transdutores de Digitais de Grandezas Elétricas. https://kron.com.br/wp-content/uploads/2020/01/Manual-do-Usuario-Mult-K-05-e-Mult-K-120-Rev-4.7.pdf, [accessed on Mar 20].

Kron (2020a). MULT-K 05 -K Multimedidor de Grandezas Elétricas. https://kron.com.br/wp-content/uploads/2020/03/Mult-K-05-Ficha-Tecnica.pdf, [accessed on Mar 20].

Kron (2020b). Protocolo MODBUS – Família Mult-K. https://kron.com.br/wp-content/uploads/2020/01/Mapas-de-registros-modbus-geral-linha-Mult-K.zip, [accessed on Mar 20].

Kron (2021). IEEE-754 Float Point 32-Bit Conversão p/ Decimal. https://kron.com.br/wp-content/uploads/2020/01/Mapas-de-registros-modbus-geral-linha-Mult-K.zip, [accessed on Mar 20].

Kugelstadt, T. (2008). The RS-485 Design Guide. *Application Report, Texas Instruments*, n. May.

Liu, G., Zhu, W., Saunders, C., Gao, F. and Yu, Y. (2015). Real-time Complex Event Processing and Analytics for Smart Grid. *Procedia Computer Science*, v. 61, p. 113–119.

Material-UI: Um framework popular de React UI (2021). https://material-ui.com/pt/, [accessed on Jun 24].

Maxim Integrated Products, I. (2014). MAX481/MAX483/MAX485/ MAX487–MAX491/MAX1487. . https://datasheets.maximintegrated.com/en/ds/MAX1487-MAX491.pdf.

Maxim Integrated Products, I. (2021). Fundamentals of RS-232 Serial Communications. https://www.maximintegrated.com/en/design/technical-documents/tutorials/8/83.html, [accessed on Mar 14].

Mella, M. J. (2019).Tese. Medidor De Energia Elétrica Por Eventos. Universidade de Passo Fundo.

Min-Allah, N. and Alrashed, S. (1 aug 2020). Smart campus—A sketch. *Sustainable Cities and Society*, v. 59, p. 102231.

MINISTÉRIO DA ECONOMIA (2020). O Controle de Tráfego em Cidades Inteligentes: um panorama dos depósitos de patente no Brasil e no Mundo. https://www.gov.br/inpi/pt-br/assuntos/informacao/controle-de-trafego-inteligente_estudo_estendido_v30062020.pdf, [accessed on Apr 5].

Modicon (1999). Modicon Modbus Protocol Reference Guide. https://modbus.org/docs/PI_MBUS_300.pdf.

morgan - npm (2021). https://www.npmjs.com/package/morgan, [accessed on Jul 13].

Morvaj, B., Lugaric, L. and Krajcar, S. (2011). Demonstrating smart buildings and smart grid features in a smart energy city. In *Proceedings of the 2011 3rd International Youth Conference on Energetics, IYCE 2011*.

MySQL (2021). https://www.mysql.com/, [accessed on Jun 3].

mysql - npm (2021). https://www.npmjs.com/package/mysql, [accessed on Jul 13].

mysql2 - npm (2021). https://www.npmjs.com/package/mysql2, [accessed on Jul 13].

Node-RED (2021). https://nodered.org/, [accessed on Jun 20].

nodemon - npm (2021). https://www.npmjs.com/package/nodemon, [accessed on Jul 13].

npm (2021). https://www.npmjs.com/, [accessed on Jun 3].

Osorio-Comparán, R., Peña-Cabrera, M., López-Juarez, I., Lefranc, G. and Tovar-Medina, R. (2017). Smart semaphore using image processing. In *2017 CHILEAN Conference on Electrical, Electronics Engineering, Information and Communication Technologies, CHILECON 2017 - Proceedings*.

Oussous, A., Benjelloun, F. Z., Ait Lahcen, A. and Belfkih, S. (2018). Big Data technologies: A survey. *Journal of King Saud University - Computer and Information Sciences*

Postman | The Collaboration Platform for API Development (2021). https://www.postman.com/, [accessed on May 3].

Raspberry Pi Foundation - About Us (2021). https://www.raspberrypi.org/about/, [accessed on Jul 3].

React – Uma biblioteca JavaScript para criar interfaces de usuário (2021). https://pt-br.reactjs.org/, [accessed on Jun 22].

Rebonatto, M. T., Eckstein, C. and Rebonatto, C. S. (30 sep 2020). Utilização de beacons em locais externos: Um estudo em um campus universitário no sul do Brasil. In *Anais do Simpósio Brasileiro de Computação Ubíqua e Pervasiva (SBCUP)*. . Sociedade Brasileira de Computacao - SB. https://sol.sbc.org.br/index.php/sbcup/article/view/11219, [accessed on Jul 3].

Recharts (2021). https://recharts.org/en-US/, [accessed on Jun 24].

Reuniao de Orientação (2021). https://drive.google.com/file/d/1WVoDybTlFTQTjWH8cKm6a4zpT5snZSWz/view?usp=sharing, [accessed on Apr 27].

Rizzardi, G. (2020).Tese. Monitoramento de energia elétrica em residências no ambiente de cidades inteligentes. Universidade de Passo Fundo.

Sadiku, M. N. O. and Alexander, C. K. (2013). *Fundamentos de Circuitos Elétricos*. 5ª ed. Porto Alegre: AMGH.

Sales Rocha, W. and Anhesine, M. W. (2020). AUTOMAÇÃO RESIDENCIAL POR COMANDO DE VOZ. *Revista Interface Tecnológica*, v. 17, n. 1.

Snoonian, D. (2003). Smart buildings. *IEEE Spectrum*. Institute of Electrical and Electronics Engineers Inc.

Soltero, M., Zhang, J., Cockril, C., et al. (2010). RS-422 and RS-485 Standards Overview and System Configurations. *Configurations*, n. May, p. 25.

Wanzeler, T., Fülber, H. and Merlin, B. (2016). Desenvolvimento de um sistema de automação residencial de baixo custo aliado ao conceito de Internet das Coisas (IoT).

What is Arduino? | Arduino (2021). https://www.arduino.cc/en/Guide/Introduction, [accessed on Jul 3].

# Apêndice A

## Outras telas de configuração do projeto

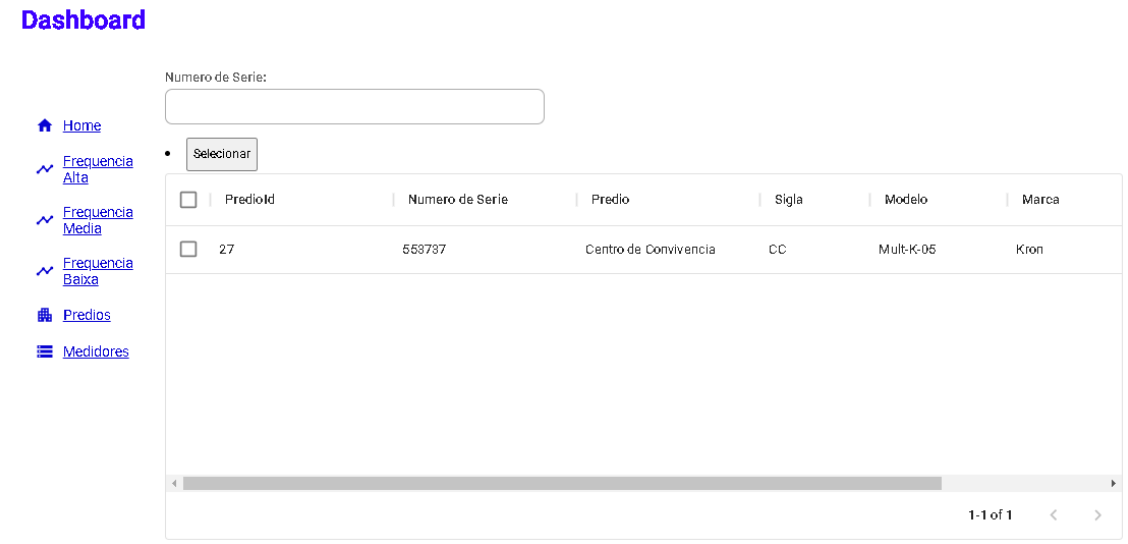


**Figura 10: Tela Home da aplicação**

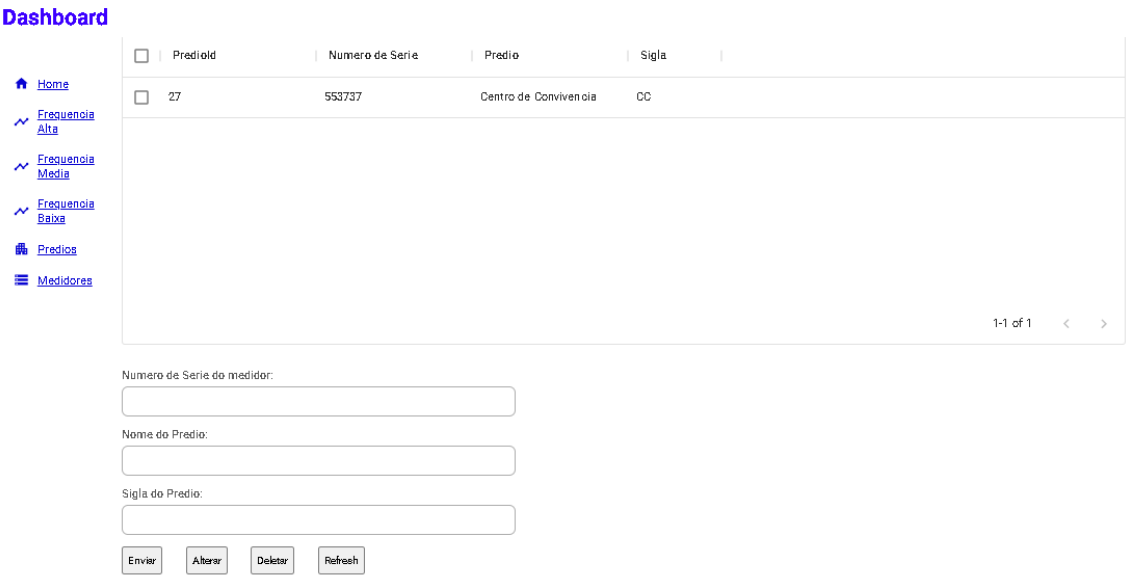


**Figura 11: Tela Cadastro de prédios**