Palavras-chave: Meteorologia, Anemômetro; IOT; DIY; Estação; Automatização.

INTRODUÇÃO

As estações meteorológicas são equipamentos para monitoramento e registro de condições climáticas com sensores configuráveis para análise de parâmetros atmosféricos e outras variáveis que afetam as rotinas meteorológicas. Os componentes podem ser: temperatura, umidade, pressão atmosférica, chuva, direção e velocidade do vento. Com os dados coletados e condições passadas, é possível calcular a probabilidade de eventos climatológicos de um futuro próximo.

A importância desses dados está ligada à previsão de como o clima pode se comportar em determinado local. Dessa forma, anteceder o que acontecerá é de extrema relevância, uma vez que se faz a necessidade em várias atividades humanas como a navegação, a aviação, a pesca, o setor energético, o gerenciamento de recursos hídricos, o turismo e a agricultura.

METODOLOGIA

O processo de coleta de dados acontece de forma automática - os sensores ligados a um microcontrolador receberão os dados e enviaram para um servidor (Datalogger). O modelo de estação automática no Brasil ainda é reduzido se comparado ao modelo convencional (um funcionário ir algumas vezes ao dia manualmente anotar os dados).

A parte instrumental do Anemômetro e Direção do vento foram produzidas em uma impressora 3D, a partir de um projeto pronto, adicionando rolamentos, imãs e parafusos.

Os sensores usados nesse projeto são, AS5600 (sensor de angulação), AH49E (sensor de efeito Hall), DTH11(sensor de temperatura e umidade) e BMP280 (sensor de pressão,altitude e temperatura).

Como microcontrolador, um ESP32 se comunica por protocolo MQTT com o servidor aberto, assim torna-se possível encaminhar e armazenar os dados coletados para um monitoramento remoto.

Figure 1: Estação meteorológica. Fonte:

Contents

1 Instrumentação

1.1 Anemometro

Como o anemômetro é um instrumento para medir a velocidade do vento, o que garante sua precisão é de acordo com sua calibração. Para calibrarmos, utilizamos um anemômetro já calibrado, o Vaavud. Esse equipamento possui um aplicativo que detecta a passagem de um imã dentro do anemômetro. Como o celular possui um sensor magnético, consegue captar qual a frequência dos pulsos. Dessa forma, foi utilizado para servir de parâmetro para calibração.

1.1.1 Estrutura de Calibração

Para calibrarmos com o outro equipamento, montamos uma estrutura em impressora 3D em que os dois anemômetros ficassem no mesmo eixo. Como os estam presos no mesmo eixo, têm a mesma frequência angular. Sendo assim, usando um tacômetro **DT-2234C** capturamos a Rotação Por Minuto(RPM). Dessa forma, podemos ter a frequência angular, que pode ser usada para achar a velocidade do objeto.

Para a precisão da linearidade da velocidade e das amostras foi usado um motor, sendo controlado por um modulador PWM com 555.

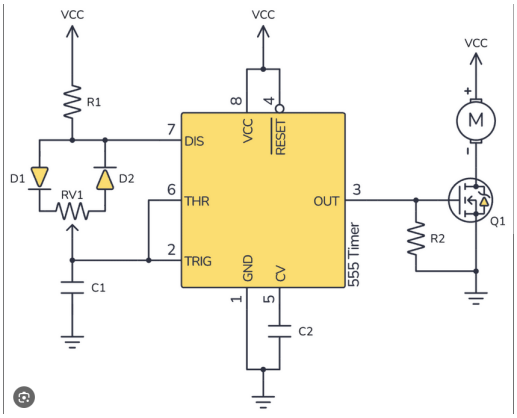

Figure 2: Modulador PWM com 555. Fonte:

Essa foi a primeira estrutura montada para começo da coleta de amostras. Com o decorrer das amostras, foi analisado a imprecisão, pois o Trimpot usado para controlar o DutyCicle, era simples e não tinhamos a mesma velocidade de saída para todas as amostras.

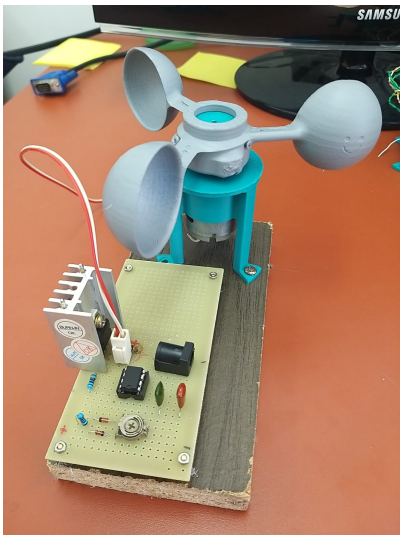

Figure 3: Modulador PWM com 555. Fonte:

Dessa forma, trocamos por um Trimpot de Precisão, de 0kOhm a 10kOhm. Dessa forma, controlamos a resistência que estava sendo aplicada e padronizamos isso durante a calibração.

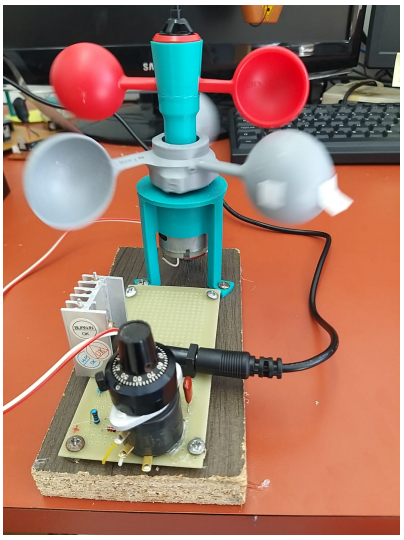

Figure 4: Modulador com o Trimpot de Precisão. Fonte:

1.1.2 Sensor Magnético

Para marcar a frequência do anemômetro, é utilizado um sensor magnético, que capta os pulsos de tensão de um imã, neste anemômetro há três imãs, um em cada pá. Nesse projeto, o sensor magnético escolhido foi o Sensor AH49E de efeito Hall (que descreve a geração de uma diferença de potencial elétrico (tensão) em um condutor quando uma corrente elétrica é aplicada perpendicularmente a um campo magnético externo).

Como o anemômetro possui três pás, são usados três imãs, no mesmo eixo das pás. E para contar um volta, é medido o número de pulsos e dividido pela quantidade de imãs.

Como o sensor de efeito Hall pode perder alguns dos pulsos ou não pegar com pouca intensidade o sinal, fizemos um sistema para que o sinal recebido fosse amplificado para uma leitura mais precisa. Sendo assim, foi desenvolvida uma PCB no software Kicad.

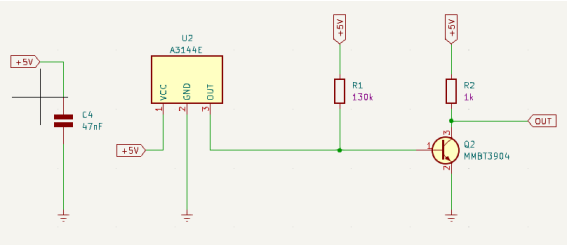

Figure 5: Esquemático sensor magnético. Fonte:

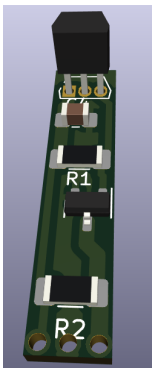

Figure 6: Visão 3D sensor magnético. Fonte:

1.1.3 Coleta de amostras

A coleta foi feita de forma manual com os seguintes passos, em um código python recebia a quantidade de amostras desejadas, em seguida, para a amostra de número um, inseria-se o valor da resistência utilizada no Trimpot, velocidade (em m/s no aplicativo do anemômetro do Vaavud), e cinco amostras de RPM refentes a essa resistência e velocidade. Sendo assim, com mais amostras era feita a média delas, para que fosse mais precisa.

Feita de forma mecânica e padronizando uma sequência na coleta para que as amostras obtidas tivessem as mesmas condições.

A imagem a seguir mostrar como eram feitas as coletas. O celular estava marcando a velocidade em m/s do anemômetro Vaavud, e com o tacômetro era lida a RPM.

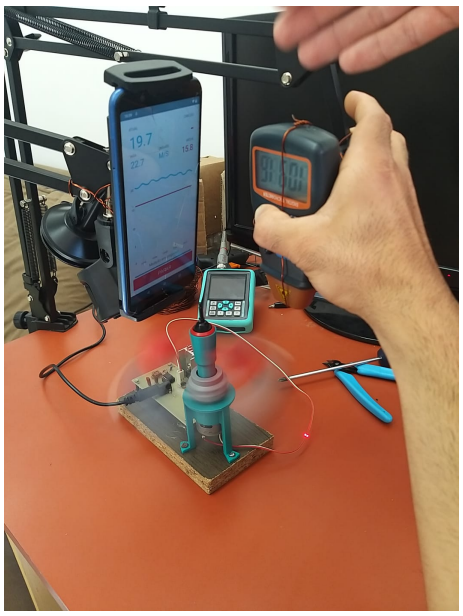

Figure 7: Coleta de amostras. Fonte:

Como os dados podiam sofrer alguma alteração, pois ao apertar com a mão o tacômetro e o tempo de leitura e angulação do laser na fita poderiam mudar, para que isso melhorasse, usamos um botão externo para acionar o tacômetro e para valor regulado do Trimpot de precisão, recebia uma leitura da velocidade e cinco amostras de RPM, tentando manter o mesmo padrão, entre visualizar e digitar.

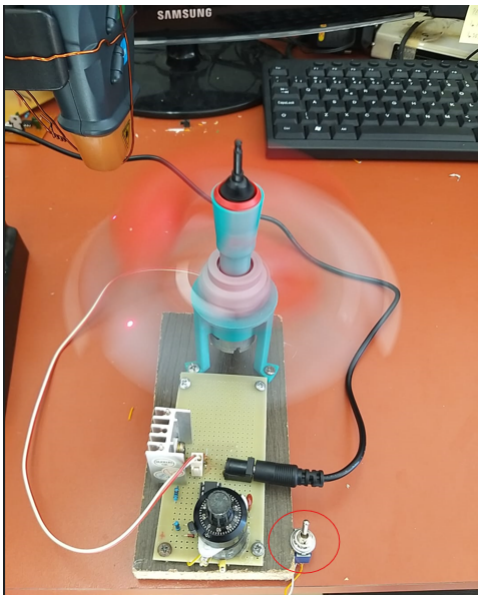

Figure 8: Tacômetro dom botão. Fonte:

1.1.4 Tratamento dos dados

Como citado anteriormente, os anemômetros estão girando no mesmo eixo, dessa forma, possuem mesma velocidade e, consequentemente, mesma frequência angular.

Depois de uma coleta de 120 amostras (excluindo as amostras que foram coletadas antes da última versão da estrutura de calibração), entre 0m/s à 21m/s, tendo como valor mínimo de 25,60rad/s e como máximo 110,93rad/s, frequência angular média de 71,96rad/s e apresentou Desvio Padrão de 23,61. Portanto, como apresentou certa linearidade, não precisamos retirar valores que fossem muito extremos ou que fugissem da reta.

Com as amostras coletadas foi feita uma regressão linear, da velocidade em relação a frequência angular. O valor resultante da equação da reta da regressão linear: Velocidade = 0.17117843*(Frequência Angular) + 0.19744649996588137.

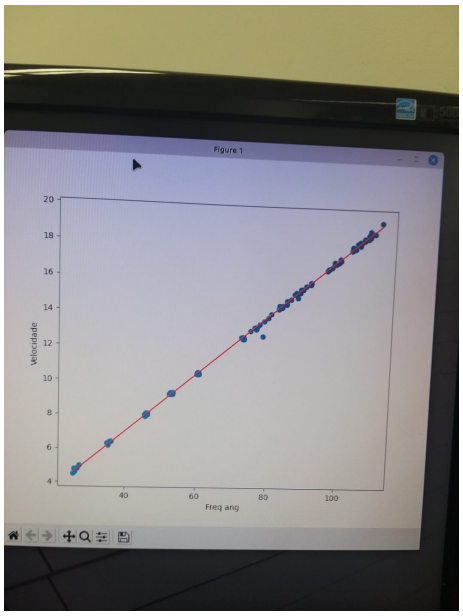

Figure 9: Regressão linear entre frequência angular e velocidade. Fonte:

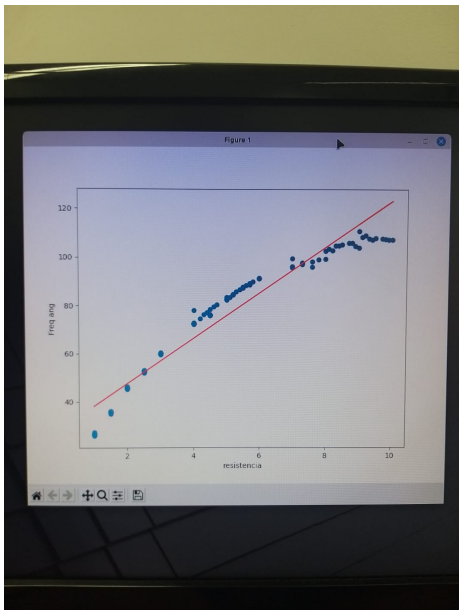

Figure 10: Regressão linear entre resistência e frequência angular. Fonte:

1.1.5 Resultados do Anemômetro

Com o resultado da regressão linear, obtendo o Rotação Por Minuto (RPM) é possível calcular a frequência angular e, assim, a velocidade em m/s.

O código usado para o tratamento da coleta de pulsos com o sensor de

efeito Hall, está disponibilizado em nosso github. Explicando breviamente o código, é feita uma interrupção em uma porta digital do arduino quando a uma borda de subida, ou seja, um pulso elétrico captado pelo sensor. A cada três pulsos é contado como uma volta completa.

Para suavização das amostras, pode ser definido a quantidade de amostras que será feita a média da velocidade. Assim, há uma vetor tem tamanho definido conforme a quantidade inserida como máximo de amostras.

Na imagem a seguir, é mostrado o arduino Nano mostrando o RPM em um display Oled, colocados na mesma PCB em que é controlado o motor para velocidade do anemômetro.

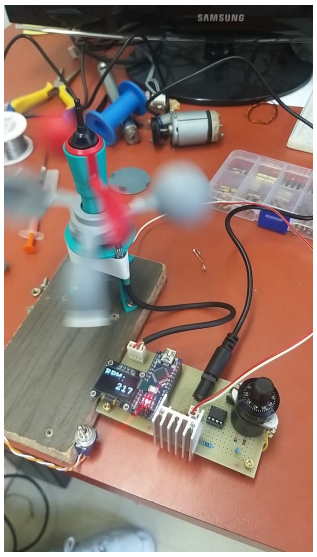

Figure 11: . Fonte:

1.2 Sensores

Além do sensor magnético, foram usados sensores para outros dados meteorológicos, como temperatura, umidade, pressão e altitude.Para temperatura e umidade o sensor DHT11, usando sinal digital em sua saída. E, também, o sensor BMP280 como objetivo principal, medir a pressão atmosférica, e com base nesses dados determinar com precisão a altitude de um ambiente, além de ser capaz de medir temperatura, este utiliza comunicação I2C.

1.3 Microcontrolador

Como microcontrolador usamos o Arduino Uno para calibração do anemômetro e alguns testes rápidos, pela versatilidade, compatibilidade e abundância de recursos que essa placa entrega.

Como a estação pode estar em lugares remotos, para enviar seus dados é necessária a comunicação com a internet, sendo assim, usamos o microocontrolador ESP32, que já possui wifi e bluetooth integrado. Além de outras vantagens em relação ao Arduino Uno, como maior poder de processamento, tendo dois núcleos,clock significativamente maior e memória.

1.4 LoRa

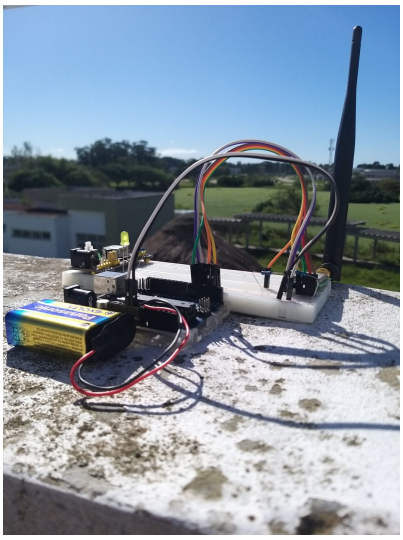

Figure 12: Estação meteorológica. Fonte:

Em alguns casos, podemos ter estações em lugares remotos, dessa forma, é importante haver uma comunicação estável e que não dependa de rede. Com isso, pode ser implementado a comunicação de radiofrequência. Nesse caso, a comunicação LoRa(Long Range) com longo alcance e baixa potência, projetada para aplicações de Internet das Coisas(IOT) e M2M(Machine-to-Machine). A vantagem dessa tecnologia é o baixo consumo de energia, umas de suas funções, apesar do baixo consumo, é a capacidade de ativar o modo Sleep (dormir) e quando solicitado voltar a funcionar e enviar os dados solicitados. Isso se torna ainda mais eficaz quando é usado o módulo LoRa para aplicações alimentadas por baterias.

Há vários modelos no mercado com muitas possibilidades de alcances, para o projeto usamos o módulo LoRa, o módulo usado é o modelo E220-900T22D, com capacidade de até 5km e frequência de 900MHz.

Por exemplo, uma estação com um microcontrolador arduino Nano coletar as informações dos sensores e enviar via LoRa para uma estação também com LoRa e um ESP32 que tem Wifi integrado para mandar os dados para o banco de dados na nuvem.

No projeto estamos enviando uma estrutura com os dados coletados em uma estação e sendo enviados para outra, como no exemplo anterior.

```
// --- Struct with station data ---
struct DATA {
  float Temp;
  float Humid;
  float Press;
  float Altitude;
  float Speed;
};
```

Figure 13: Estrutura sendo enviada via LoRa.

2 Gerenciamento dos Dados

Para que a estação meteorológica seja automatizada, precisa que seus dados sejam tratados e armazenados em algum lugar. Então, mesmo a estação estando em lugares remotos ou não, há a necessidade de que esteja preparada para enviar seus dados por meio de algum protocolo para um banco de dados. Dessa forma, essas informações podem ser usadas em previsões climáticas futuras.

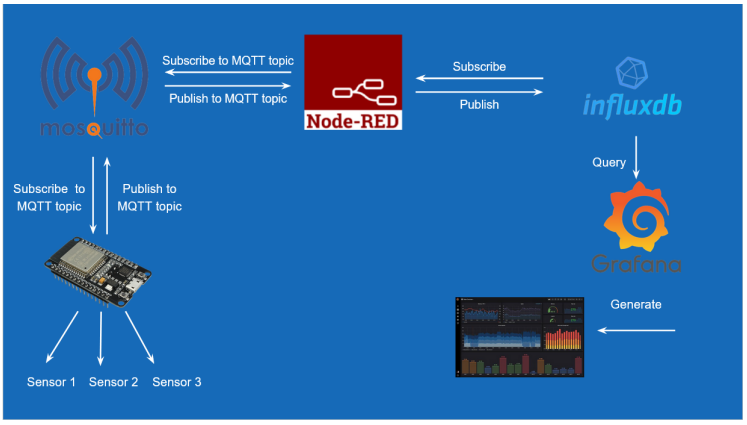

Figure 14: Esquemático do gerenciamento entre as aplicações.

2.1 Conexão MQTT

Após os dados estarem no microcontrolador, com a função de wifi do ESP32, usamos o protocolo de comunicação MQTT(Message Queuing Telemetry Transport) é um protocolo de mensagens leve e eficiente, projetado para comunicação entre dispositivos em redes de IoT (Internet das Coisas) ou em redes de comunicação com largura de banda limitada. Desenvolvido pela IBM nos anos 1990, o MQTT tornou-se um padrão aberto e é amplamente utilizado em uma variedade de aplicações IoT devido à sua simplicidade, eficiência e confiabilidade.

Usar um Broker MQTT facilita a comunicação entre dispositivos na arquitetura de mensagens, pois atua como um intermediário centralizado na comunicação entre dispositivos. Ademais, o gerenciamento de tópicos, broker MQTT

facilita o gerenciamento de tópicos e assinaturas. Ele permite que os dispositivos publiquem mensagens em tópicos específicos e que outros dispositivos subscrevam a esses tópicos para receber as mensagens. Isso simplifica a organização e o controle da comunicação em uma rede de dispositivos.

No contexto apresentado, utilizamos o broker Mosquitto, um servidor MQTT de código aberto. Desenvolvido pela Eclipse Foundation, o Mosquitto é reconhecido como uma escolha amplamente adotada para implementações de IOT e outras aplicações que requerem comunicação baseada nesse protocolo.

2.2 Node Red

O Node Red é um software para programar por meio de fluxogramas, utilizando Nodejs. Usa-se nodos para conectarem entre si.

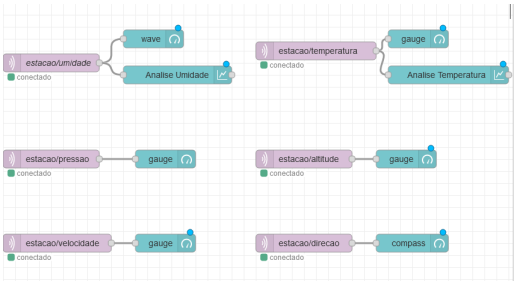

Figure 15: Nodos Node Red. Fonte:

Para configurar a comunicação com o banco de dados Influxdb e para criar uma dashboard do próprio Node Red, foram adicionadas as paletas **node-red-contrib-influxdb** e **node-red-dashboard**.

Como servidor foi usado o test.mosquitto.org:1883, esse é o servidor aberto do Broker Mosquitto. Como é um servidor aberto para uso de protocolo MQTT, o site recomenda não ser usado para aplicações grandes, porém, se for necessário maior resposta e do servidor, recomendasse utilizar um servidor privado.

O **Tópico** é o nome que será usado para acessar a informação publicada. Como exemplo, **estação/temperatura**.

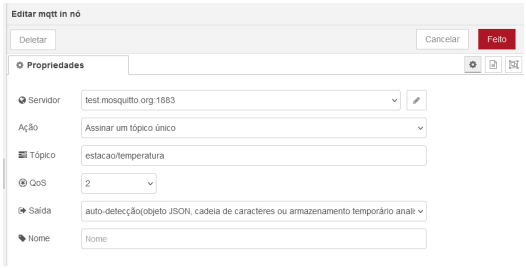

Figure 16: Configurando tópico Node Red. Fonte:

2.3 Influxdb

O InfluxDB foi o banco de dados usado para armazenamento das informações. Muito utilizado para series temporais, o InfluxDB se encaixa muito bem para controlarmos os dados da estação meteorológica.

O banco de dados está rodando localmente na porta 8086 de um computador com sistema operacional linux, MINT **21.3**.

2.3.1 Instalar InfluxDB

Para instalar o InfluxDB em sua máquina, use os seguintes comando no seu terminal:

Para instalar o InfluxDB:
$curl -O https://download.influxdata.com/influxdb/releases/influxdb2_2.7.6−1_amd64.deb
$sudo dpkg − i influxdb2_2.7.6 − 1_amd64.deb

Para rodar o InfluxDB:
$influxd
O programa já estará rodando localmente, em seu navegador acesse: **htpps://localhost:8086/**

2.3.2 Configuração do InfluxDB

Em seu perfil criará uma Organization, depois será necessario gerar um Token para dar permição ao Node-Red enviar os dados. Após ter o token, crie um Bucket, local onde ficarão os dados recebidos, no nosso exemplo o nome do Bucket é estacaoMeteorologica.

2.4 Grafana

Para instalar o Grafana em sua máquina, use os seguintes comando no seu terminal:

Para instalar o Grafana:
$ sudo apt update
$ sudo apt-get install -y adduser libfontconfig1 musl
$ wget https://dl.grafana.com/enterprise/release/grafana-enterprise$_1$0.4.2$_a$*md*64.*deb*
$$sudodpkg - igrafana - enterprise_1 0.4.2_a md64.deb$$
$$sudosystemctlenablegrafana - server$$
$$sudosystemctlstartgrafana - server$$

Para checar se está tudo dando certo:
O programa já estará rodando localmente, em seu navegador acesse: **htpps://localhost:3000/**

A dashboard para vizualição pode ser personalizada.

Referências:

NODE.JS. Node.js. Disponível em: https://nodejs.org/en. Acesso em: 19 maio 2023.

MOSQUITTO. Mosquitto. Disponível em: https://mosquitto.org/. Acesso em: 19 maio 2023.

NODE-RED. Node-RED. Disponível em: https://nodered.org/. Acesso em: 19 maio 2023.

INFLUXDATA. Documentation. Disponível em: https://docs.influxdata.com/. Acesso em: 19 maio 2023.

GRAFANA LABS. Grafana. Disponível em: https://grafana.com/. Acesso em: 19 maio 2023.