



Ministério da Educação  
Secretaria de Educação Profissional e Tecnológica  
Instituto Federal Catarinense  
*Campus Videira*

---

**RYAN KARLING**

**ESTRATÉGIAS DE VISUALIZAÇÃO PARA DADOS COLETADOS  
DE SISTEMAS AGRÍCOLAS INTELIGENTES**

Videira  
2024

**RYAN KARLING**

**ESTRATÉGIAS DE VISUALIZAÇÃO PARA DADOS COLETADOS  
DE SISTEMAS AGRÍCOLAS INTELIGENTES**

Trabalho de Conclusão de Curso apresentado ao  
Curso de graduação em Ciência da Computação  
do Instituto Federal Catarinense – *Campus* Vi-  
deira para obtenção do título de bacharel em Ci-  
ênci a da Computação.

Orientador: Angelita Rettore de Araujo Zanella  
Coorientador: João Hemkemaier

Videira  
2024

## **RESUMO**

A agricultura inteligente representa um avanço significativo na produção agrícola, integrando dispositivos de Internet das Coisas e sistemas de análise de dados para monitorar e otimizar o manejo das culturas. Esse cenário traz consigo a necessidade de sistemas robustos que garantam a integridade e a segurança dos dados coletados. O CEIFA foi projetado para monitorar anomalias no ambiente agrícola e surge como uma solução promissora, permitindo que produtores identifiquem problemas em tempo real. No entanto, a ausência de uma interface gráfica limita seu uso para monitoramento direto, gerando a necessidade de uma plataforma que permita a visualização dos dados de forma intuitiva e fácil. Buscando superar esta limitação, este trabalho avalia plataformas de software como serviço gratuitas e de código aberto que possam ser utilizadas para visualização de dados. A avaliação tem como objetivo determinar a mais adequada para integração ao CEIFA em dispositivos de borda e na nuvem. Para isso foram realizados testes em diferentes cenários com variados números de sensores e acessos simultâneos, visando simular tanto pequenos quanto médios ambientes agrícolas. As plataformas Metabase e Grafana apresentam-se mais adequadas dentre as opções testadas, enquanto outras soluções apresentam limitações técnicas ou alto custo operacional. O Metabase destaca-se por sua interface completa, adequada para análises detalhadas, embora exija maior capacidade computacional, o que o torna ideal para uso em servidores na nuvem. Por outro lado, o Grafana, com menor consumo de recursos e eficiência em dispositivos com restrições, mostra-se a melhor opção para o ambiente de borda.

**Palavras-chave:** Agricultura Inteligente. Software Como Serviço. CEIFA. Interface gráfica de usuário.

## **ABSTRACT**

Smart agriculture represents a significant advancement in agricultural production, integrating Internet of Things devices and data analysis systems to monitor and optimize crop management. This scenario brings with it the need for robust systems that ensure the integrity and security of the collected data. The CEIFA system was designed to monitor anomalies in the agricultural environment and emerges as a promising solution, allowing producers to identify problems in real time. However, the absence of a graphical interface limits its use for direct monitoring, creating the need for a platform that enables intuitive and easy data visualization. To overcome this limitation, this work evaluates free, open-source software-as-a-service platforms that can be used for data visualization. The evaluation aims to determine the most suitable platform for integration with CEIFA in both edge devices and cloud environments. For this purpose, tests were conducted across different scenarios with varying numbers of sensors and simultaneous accesses to simulate both small and medium agricultural environments. Among the tested options, the Metabase and Grafana platforms were found to be the most suitable, while other solutions presented technical limitations or high operational costs. Metabase stands out for its comprehensive interface, suitable for detailed analysis, although it requires greater computational capacity, making it ideal for use on cloud servers. On the other hand, Grafana, with lower resource consumption and efficiency on constrained devices, proves to be the best option for the edge environment.

Key-words: SmartFarming. Software as a service. CEIFA. Graphical User Interface

## **LISTA DE ILUSTRAÇÕES**

Figura 1 – Estrutura geral do CEIFA . . . . .	8
Figura 2 – Estrutura geral da agricultura inteligente . . . . .	11
Figura 3 – Gráfico de Velocidade do Vento na plataforma Zabbix . . . . .	29
Figura 4 – Tela de consulta SQL e configuração do gráfico no Metabase . . . . .	31
Figura 5 – Dashboard principal do Metabase . . . . .	32
Figura 6 – Consumo de recursos do Metabase . . . . .	33
Figura 7 – Painel de gráficos do Grafana . . . . .	35
Figura 8 – Consumo de recursos do Grafana . . . . .	36
Figura 9 – Consumo de Processamento na borda . . . . .	40
Figura 10 – Consumo de Processamento no ambiente de Nuvem . . . . .	40
Figura 11 – Consumo de Memória RAM na borda . . . . .	41
Figura 12 – Consumo de Memória RAM no ambiente de Nuvem . . . . .	42
Figura 13 – Tela de edição da dashboard no Metabase . . . . .	50

## **LISTA DE ACRÔNIMOS**

API	<i>Application Programming Interface.</i>
BI	<i>Business Intelligence.</i>
CEIFA	<i>Cloud-Edge Identifier of Farming Anomalies.</i>
DDOS	<i>Distributed Denial of Service.</i>
GUI	<i>Graphical User Interface.</i>
IoT	<i>Internet of Things.</i>
SaaS	<i>Software as a Service.</i>
TI	<i>Tecnologia da Informação.</i>
VPS	<i>Virtual Private Server.</i>

## SUMÁRIO

<b>1</b>	<b>INTRODUÇÃO . . . . .</b>	<b>7</b>
1.1	Objetivos . . . . .	8
<b>1.1.1</b>	<b>Objetivo Geral . . . . .</b>	<b>8</b>
<b>1.1.2</b>	<b>Objetivos Específicos . . . . .</b>	<b>9</b>
<b>2</b>	<b>REFERENCIAL TEÓRICO . . . . .</b>	<b>10</b>
2.1	Estratégias de Visualização . . . . .	12
<b>2.1.1</b>	<b>Ferramenta para o CEIFA . . . . .</b>	<b>14</b>
<b>2.1.2</b>	<b>Zabbix . . . . .</b>	<b>15</b>
<b>2.1.3</b>	<b>Metabase . . . . .</b>	<b>16</b>
<b>2.1.4</b>	<b>Grafana . . . . .</b>	<b>17</b>
<b>2.1.5</b>	<b>SigNoz . . . . .</b>	<b>19</b>
<b>2.1.6</b>	<b>Kibana . . . . .</b>	<b>20</b>
<b>3</b>	<b>METODOLOGIA . . . . .</b>	<b>23</b>
<b>4</b>	<b>Cenários de Teste . . . . .</b>	<b>25</b>
4.1	Cenários . . . . .	25
4.2	Testes . . . . .	26
<b>4.2.1</b>	<b>Zabbix . . . . .</b>	<b>28</b>
<b>4.2.2</b>	<b>Metabase . . . . .</b>	<b>30</b>
<b>4.2.3</b>	<b>Grafana . . . . .</b>	<b>33</b>
<b>4.2.4</b>	<b>SigNoz . . . . .</b>	<b>36</b>
<b>4.2.5</b>	<b>Kibana . . . . .</b>	<b>37</b>
4.3	Discussão Geral . . . . .	38
<b>5</b>	<b>Conclusão . . . . .</b>	<b>44</b>
	<b>REFERÊNCIAS . . . . .</b>	<b>48</b>
<b>APÊNDICE A</b>	<b>Edição de dashboard no Metabase . . . . .</b>	<b>50</b>

## 1 INTRODUÇÃO

Com o aumento das taxas de produção, a agricultura enfrenta o desafio de melhorar a eficiência produtiva, visando minimizar os impactos causados por condições climáticas, garantindo a segurança e estabilidade do cultivo (ZANELLA, 2022). Nesse contexto, a Agricultura Inteligente (*smart farming*) colabora com o monitoramento e coleta de informações do campo para auxiliar no cultivo de maneira assertiva.

Em todas as etapas do cultivo, a Agricultura Inteligente visa empregar sensores para coletar dados, como por exemplo, níveis de luz, condições do solo, umidade, temperatura entre outros (SCHLEGEL; POLETTI, 2019). Estes dados podem ser utilizados para tomadas de decisões, tanto automáticas (feito por um sistema inteligente) quanto manuais (feitos por um humano). Estes sistemas inteligentes servem para auxiliar nas decisões relacionadas à irrigação da lavoura, aplicação de agroquímicos, semeadura e outras práticas agrícolas essenciais.

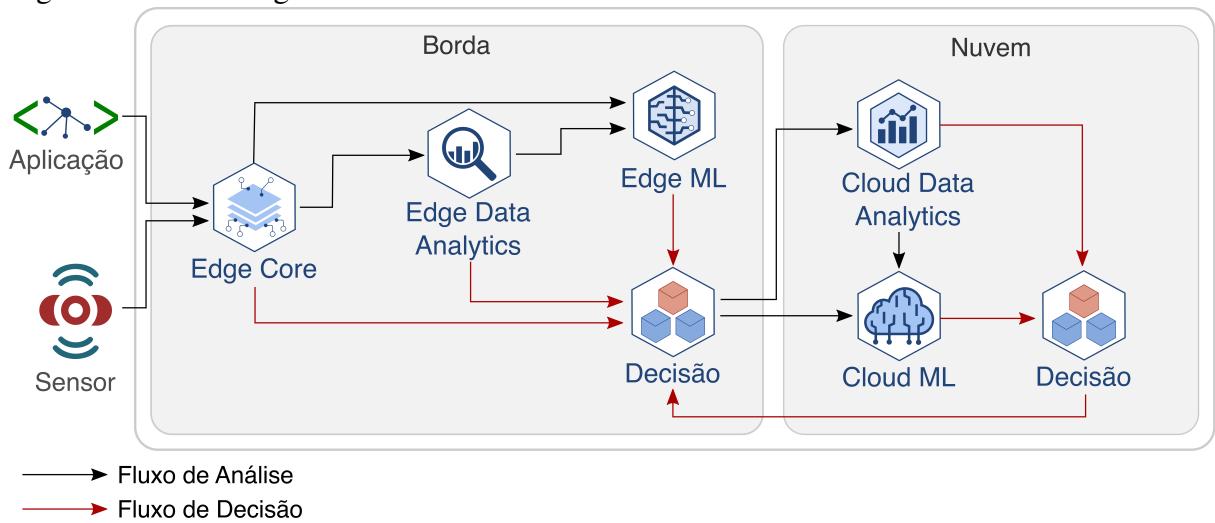
A segurança na coleta e transmissão destes dados não tem sido algo muito notado recentemente, uma vez que os sistemas desenvolvidos não incorporam muitos recursos de segurança (ZANELLA; SILVA; ALBINI, 2020). Isso pode ser um problema visto que agentes maliciosos ou condições adversas podem afetar diretamente o cultivo e a tomada de decisões. Neste contexto o CEIFA (do inglês *Cloud-Edge Identifier of Farming Anomalies*) se apresenta como uma ferramenta acessível e promissora, tanto para pequenos agricultores quanto para grandes fazendeiros. Seu principal objetivo é verificar a integridade e confiabilidade dos dados coletados de sistemas agrícolas inteligentes, por meio da detecção de anomalias.

As decisões e análises dependem diretamente dos dados recebidos pelos dispositivos de coleta de dados. A integridade desses dados é de suma importância, visto que os sensores podem estar sujeitos a várias formas de avarias que podem comprometer suas funcionalidades. Por exemplo, um ou mais sensores podem ser esmagados por um trator, revirados por animais ou sofrer degradação devido às condições climáticas ou ambientais. Essas avarias são críticas porque o sistema inteligente, que atua no ambiente, depende dessas informações para realizar ações automáticas, como acionar bombas de irrigação ou outros atuadores. No entanto, se os dados estiverem incorretos ou inconsistentes, o acionamento pode ocorrer fora das condições para as quais foi programado, podendo prejudicar a lavoura.

O CEIFA opera em dois módulos independentes: um na borda e outro na nuvem, como representado na Figura 1. Os sensores e agentes da agricultura inteligente são conectados

à borda por meio de tecnologia de rede sem fio. Todas as informações coletadas são enviadas para o CEIFA, que tem o objetivo de verificar se o dado coletado é confiável ou se possui algum tipo de inconsistência. Esta análise visa entender se as informações recebidas não sofreram alterações desde a última coleta, e os compara com outros dados de outros sensores (ZANELLA, 2022).

Figura 1 – Estrutura geral do CEIFA



Fonte: ZANELLA (2022)

O CEIFA foi projetado apenas para detectar anomalias, sem qualquer interface gráfica para configuração ou visualização das informações. Os dados coletados são analisados, classificados, armazenados no banco de dados e retornados à aplicação que os enviou, sem uma Interface Gráfica do Usuário (GUI, do inglês *Graphical User Interface*). No entanto, para que os usuários possam monitorar e gerenciar o cultivo de forma eficaz, o acesso a essas informações precisa ser facilitado. Apenas informações em modo texto limitam a interpretação rápida e precisa dos dados, enquanto uma GUI estável permite visualizações claras e intuitivas. Isso contribui para o sistema como um todo, possibilitando tomadas de decisão ágeis e informadas e promove a gestão eficiente da lavoura, especialmente em ambientes onde o hardware da borda é limitado e exige uma apresentação de dados otimizada.

## 1.1 OBJETIVOS

### 1.1.1 Objetivo Geral

O objetivo geral deste trabalho é avaliar formas eficazes de apresentar dados coletados por dispositivos de Internet das Coisas (IoT, do inglês *Internet of Things*) para Agricultura

Inteligente de forma gráfica e estável ao usuário.

### **1.1.2 Objetivos Específicos**

Para atingir o objetivo geral, foram estabelecidos os seguintes objetivos específicos:

- Identificar a melhor forma de hospedar o sistema de monitoramento, localmente (hospedado na borda) ou na nuvem.
- Avaliar softwares gratuitos disponíveis e os seus recursos.
- Analisar o custo computacional de cada abordagem.
- Verificar a estabilidade das plataformas de visualização, garantindo que elas possam lidar com picos de tráfego e volumes de dados sem comprometer a qualidade do serviço.

## 2 REFERENCIAL TEÓRICO

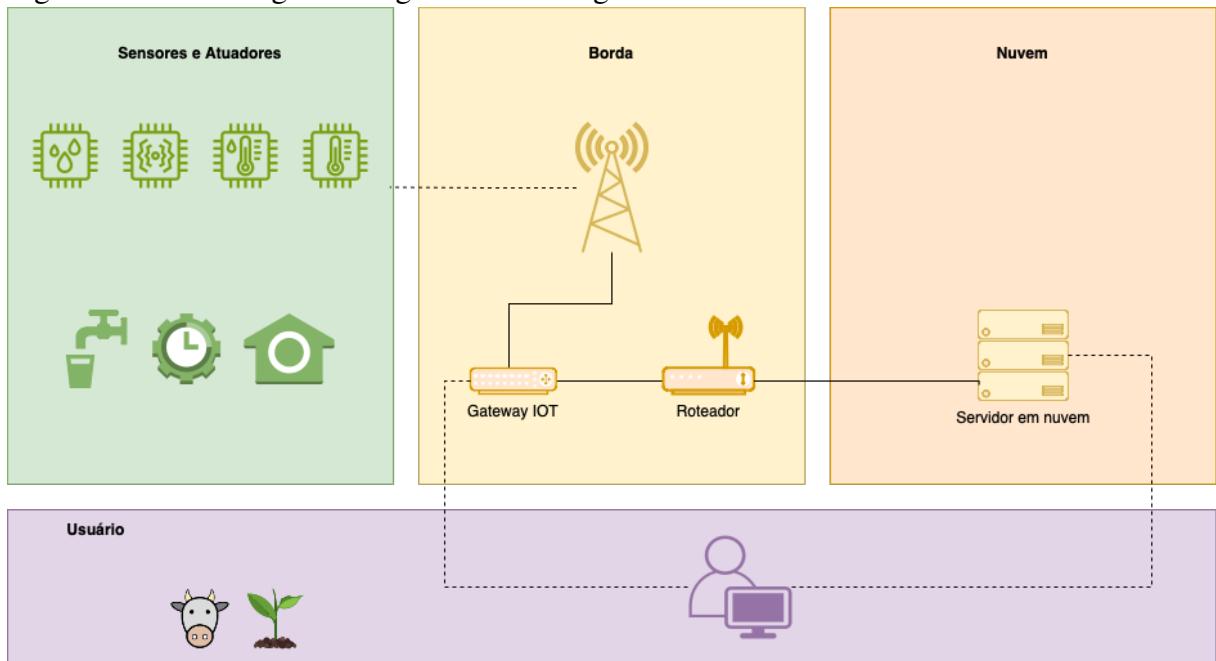
A agricultura inteligente surge a partir da popularização de equipamentos de IoT que começam a ser utilizados em automações industriais, residenciais e em outros ambientes, chegando ao ambiente agrícola. Este tipo de agricultura permite um plantio assertivo, o controle de pragas e de doenças através da análise de dados. A análise de dados emprega tecnologias de *Big Data*, e permite integrar informações do campo com previsões meteorológicas para, por exemplo, proteger as culturas. Essa análise também abrange as condições do solo e da lavoura, sendo essencial principalmente para o monitoramento contínuo dessas culturas (WOLFERT et al., 2017).

Os sistemas agrícolas inteligentes são compostos por sensores, atuadores, dispositivos de comunicação e outros componentes, organizados em camadas que otimizam o funcionamento e a conectividade dos dispositivos em ambientes agrícolas. Essa estrutura é dividida em três camadas principais: percepção, borda e nuvem. A camada de percepção, como ilustrado na Figura 2, inclui sensores que coletam dados diretamente do ambiente, como temperatura, umidade e condições do solo, além de atuadores que realizam ações específicas, como ajuste de irrigação ou controle de pragas. Esses dispositivos são limitados em termos de processamento e armazenamento e, portanto, dependem de camadas superiores para transmitir os dados coletados.

A borda recebe os dados dos sensores e realiza processamento preliminar antes de encaminhá-los para a nuvem. A borda pode conter dispositivos com recursos computacionais limitados, como Arduino, ESP32 ou Raspberry Pi, que atuam como *gateways*, permitindo a transmissão de dados entre a camada de percepção e a nuvem. Alguns desses dispositivos na borda têm capacidade para tomar decisões locais e enviar comandos aos atuadores, reduzindo a necessidade de constante comunicação com a nuvem. Esse processamento local, realizado através de filtros de dados e ferramentas de decisão (ZANELLA, 2022).

Na nuvem, os dados são armazenados e processados em maior escala, oferecendo capacidade robusta de armazenamento e processamento, facilitando o uso de algoritmos avançados e a integração com Inteligência de negócios (BI, do inglês *Business Intelligence*) para transformar dados brutos em informações. No entanto, um dos desafios dessa estrutura é garantir a segurança dos dados. Em muitos casos, esses dispositivos de borda e sensores apresentam poucos recursos de segurança, o que pode comprometer a integridade dos dados e a operação

Figura 2 – Estrutura geral da agricultura inteligente



Fonte: O Autor, 2024

como um todo. A falta de mecanismos robustos de segurança, especialmente na camada de borda e nos sensores, representa um ponto vulnerável e requer atenção para evitar potenciais riscos ao sistema de agricultura inteligente como um todo.

Segundo Schiller et al. (2022), dispositivos de IoT são frequentemente vulneráveis a ataques cibernéticos, principalmente devido ao uso de credenciais fracas. Além disso, a implementação inadequada de medidas de segurança, incluindo níveis insuficientes de proteção, contribui significativamente para ataques e invasões, expondo os dispositivos a riscos adicionais e comprometendo a integridade e a confidencialidade dos dados coletados e transmitidos. Tais deficiências servem como pontos de entrada para uma variedade de ataques, desde Ataque Distribuído de Negação de Serviço (DDOS, do inglês *Distributed Denial of Service*) até alterações maliciosas de informações do sistema, afetando a integridade dos dados e provocando descontrole operacional.

Tendo em vista as falhas e erros de hardware ou software, além dos ataques cibernéticos que podem afetar os sistemas agrícolas inteligentes, Zanella (2022) propôs o CEIFA, um detector híbrido de anomalias projetado para monitorar e identificar dados incorretos em sistemas de agricultura inteligente. Estas anomalias podem ser indicativas de falhas nos dispositivos ou presença de ataques cibernéticos. Operando de forma híbrida, o sistema integra componentes na borda e na nuvem, como visto na Figura 2. Na borda, o gateway pode ser im-

plementado em dispositivos com capacidade limitada como Arduino, ESP32 ou Raspberry Pi, enquanto a nuvem se encarrega de realizar análises complexas e gerenciar o armazenamento de dados. Contudo, uma limitação crítica do CEIFA é a ausência de uma GUI, tanto na borda quanto na nuvem, o que restringe a interatividade dos usuários com o sistema e o acesso direto às informações processadas.

A utilização de uma GUI facilita a detecção de anomalias em sistemas de agricultura inteligente, permitindo uma identificação rápida e intuitiva de dados inconsistentes e viabilizando ações imediatas sendo fundamental para assegurar a integridade e a confiabilidade dos dados coletados. Dados íntegros e confiáveis possibilitam análises precisas, que são essenciais para identificar condições adversas como doenças, pragas ou estresse hídrico antes que impactem significativamente as culturas. Um exemplo é o uso de detectores de anomalias que analisam dados climáticos de colheita para identificar padrões incomuns. Tais sistemas podem alertar os agricultores sobre possíveis problemas, permitindo intervenções rápidas e precisas. A eficácia dessas tecnologias foi demonstrada em estudos que mostram uma melhoria significativa na precisão da detecção de anomalias, o que, por sua vez, pode ser diretamente vinculado à minimização de danos às culturas e melhoria da produtividade geral do cultivo (MOSO et al., 2021).

Essa capacidade de resposta rápida e orientada por dados, viabilizada pela interface gráfica e pela análise de anomalias, destaca o papel da agricultura inteligente na tomada de decisões baseadas em dados, onde as informações coletadas são transformadas em ações eficientes e rentáveis. A gestão de informações de culturas, combinada com a análise inteligente, facilita decisões otimizadas que promovem a sustentabilidade e a produtividade. Estas decisões são suportadas por avanços em tecnologias de análise de dados, permitindo que os agricultores não só economizem recursos, mas também melhorem o manejo ambiental de suas operações. Essas práticas são fundamentais para ajustar as operações agrícolas às demandas e desafios futuros, aumentando a eficácia com que os alimentos são produzidos para atender ao crescimento populacional previsto (SÁIZ-RUBIO; ROVIRA-MÁS, 2020).

## 2.1 ESTRATÉGIAS DE VISUALIZAÇÃO

A eficiência dos sistemas de detecção de anomalias na agricultura inteligente se amplia consideravelmente quando os dados são bem interpretados e visualizados. Outros setores, como gerenciamento de cidades e condomínios inteligentes com infraestrutura crítica, já

utilizam amplamente ferramentas de visualização de dados para monitoramento e tomada de decisões (BOULOUKAKIS et al., 2018). Ferramentas como Grafana (GRAFANA LABS, 2024) e Kibana (ELASTICSEARCH B.V., 2024) desempenham um papel essencial neste processo, transformando dados coletados em visualizações comprehensíveis e interativas. Essas plataformas não apenas simplificam a apresentação de informações complexas, mas também permitem que os agricultores e gestores tomem decisões rápidas. A habilidade de visualizar dados em tempo real<sup>1</sup> e de maneira intuitiva é importante, pois facilita a identificação imediata de padrões críticos que exigem atenção.

Além disso, a integração de ferramentas de visualização em diferentes cenários tecnológicos na agricultura, desde sistemas embarcados até soluções baseadas em nuvem, é de suma importância para garantir que sejam acessíveis e eficazes independentemente das limitações de hardware ou da infraestrutura de Tecnologia da Informação (TI). A seleção de ferramentas assegura que todos os níveis de usuários, com ou sem habilidades técnicas, possam beneficiar-se de suas funcionalidades. A escolha da ferramenta de visualização impacta diretamente a capacidade dos agricultores de reagir a desafios emergentes e de otimizar os processos baseando-se em dados precisos e atualizados, promovendo uma gestão agrícola mais inteligente.

O acompanhamento de informações através de interfaces gráficas é importante em diversos ambientes além da agricultura, pois facilita o acesso rápido à informação provendo uma visão fácil e completa de dados para tomada de decisões. Em sistemas de gerenciamento de edifícios inteligentes, por exemplo, as GUIs facilitam o controle de iluminação, segurança e sistemas de climatização. Em infraestruturas críticas, ajudam a monitorar e a responder a condições operacionais em tempo real. Essas interfaces proporcionam uma visão holística e integrada do estado operacional, permitindo uma resposta rápida a anomalias ou redução do desempenho(BOULOUKAKIS et al., 2018).

Ferramentas de *smart farming*, como Bushel Farm (2024) e Agworld (GROENEVELD, 2021), demonstram a importância das interfaces gráficas na análise de dados agrícolas. Essas plataformas permitem que os agricultores visualizem informações críticas de maneira intuitiva, melhorando significativamente a gestão dos recursos e as práticas de cultivo. Brushel Farm, por exemplo, oferece recursos que ajudam na visualização da umidade do solo e na monitoração das condições climáticas. Já o Agworld facilita a colaboração e o compartilhamento

---

<sup>1</sup> Em sistemas operacionais, “tempo real” refere-se à capacidade do sistema de responder a eventos ou processar dados em um intervalo de tempo previsível, definido como crítico para a aplicação.(<https://wiki.inf.ufpr.br/maziero/lib/exe/fetch.php?media=socm:socm-livro.pdf>)

de dados em tempo real entre equipes, o que é essencial para a tomada de decisões estratégicas na agricultura.

### **2.1.1 Ferramenta para o CEIFA**

O desenvolvimento de uma aplicação própria para a visualização de dados pode parecer uma solução ideal, entretanto é inviável quando se consideram os altos custos financeiros e computacionais envolvidos. Criar uma solução personalizada demanda uma equipe especializada, recursos financeiros substanciais e um longo período de desenvolvimento, além de extensivos processos de testes e validação para garantir a eficácia e segurança do software. Esses fatores contrariam o objetivo de manter baixos os custos financeiros e computacionais. Em contrapartida, utilizar plataformas de Software como Serviço (SaaS, do inglês *Software as a Service*) já existentes, que são gratuitas e robustas, é uma escolha mais prática e econômica.

Ao selecionar softwares para a agricultura inteligente, é essencial considerar aqueles que oferecem a capacidade de operar de forma eficiente tanto em dispositivos de borda quanto em plataformas baseadas em nuvem, garantindo acessibilidade e eficácia em todos os níveis de implementação. A flexibilidade e adaptabilidade das interfaces, que devem ajustar-se bem a diversos dispositivos e resoluções, são características fundamentais para otimizar a interatividade com as informações, reforçando a gestão eficaz dos recursos agrícolas.

Para que possa ser garantida a eficácia e acessibilidade os softwares analisados para geração de GUI foram utilizadas plataformas SaaS gratuitas de código aberto. O objetivo é identificar opções que não apenas minimizem os custos financeiros, mas que também sejam compatíveis com as restrições computacionais dos dispositivos de borda. Estes softwares serão avaliados em dois contextos distintos: operação local em equipamentos de borda e processamento em nuvem. A compatibilidade com a arquitetura CEIFA é um dos critérios fundamentais nesta análise, garantindo que as ferramentas escolhidas se integrem sem problemas aos sistemas existentes.

Plataformas como Zabbix, Metabase, Grafana, Signós e Kibana são conhecidas por sua facilidade de instalação em sistemas operacionais amplamente utilizados e por demandarem recursos limitados. Contudo, cada uma apresenta características distintas em termos de funcionalidades, integrações e aplicações específicas. Uma avaliação detalhada será realizada para elucidar as particularidades e o potencial de cada plataforma. Esse estudo aprofundado ajudará a entender melhor suas capacidades e a determinar sua adequação em variados cenários

operacionais.

### 2.1.2 Zabbix

O Zabbix é uma plataforma robusta e amplamente utilizada para o monitoramento de redes, sistemas e aplicativos. Desenvolvida por Alexei Vladishev e mantida por uma comunidade global, a ferramenta é de código aberto e oferece uma solução flexível e transparente para empresas de diversos tamanhos. Seu propósito principal é fornecer uma visão unificada e centralizada do desempenho de infraestruturas de TI, o que a torna ideal para monitorar ambientes complexos, como aqueles compostos por dispositivos de IoT e redes distribuídas, além de ser compatível com o Raspberry Pi OS (PI MY LIFE UP, 2024).

Uma das principais características do Zabbix é sua escalabilidade. A plataforma pode ser ajustada conforme o volume de parâmetros e dispositivos monitorados, facilitando a implementação em ambientes que vão desde pequenas empresas até grandes corporações com milhares de dispositivos conectados (PIKKARAINEN, 2023). Além disso, a arquitetura do Zabbix, baseada no modelo “agente-servidor”<sup>2</sup>, suporta uma ampla gama de protocolos, incluindo SNMP, ICMP e IPMI, garantindo compatibilidade com diversos sistemas e dispositivos. O armazenamento de dados também é expansível, com suporte a múltiplos servidores de banco de dados, como MySQL, Percona, MariaDB e PostgreSQL, permitindo que o Zabbix gerencie grandes volumes de dados sem comprometer o desempenho (ZABBIX, 2024).

Além das capacidades apresentadas, o Zabbix destaca-se em termos de automação e monitoramento em tempo real. Ele permite a criação de *triggers*<sup>3</sup> personalizadas e alertas automáticos que notificam os administradores sobre anomalias no desempenho do sistema. Essas notificações podem ser configuradas para acionar ações automatizadas, como a execução de *scripts*<sup>4</sup> de correção ou a reinicialização de serviços, evitando falhas críticas e garantindo a continuidade dos serviços (ZABBIX, 2024).

Sua interface é voltada tanto para usuários técnicos quanto para aqueles com menos experiência, proporcionando uma interface amigável e *dashboards* customizáveis, o que facilita a visualização de métricas em tempo real e a criação de relatórios detalhados. Essas fun-

<sup>2</sup> A arquitetura “agente-servidor” refere-se a um modelo de comunicação onde o servidor centraliza o gerenciamento e o monitoramento, enquanto os agentes instalados em dispositivos remotos coletam dados de desempenho e os enviam para o servidor.

<sup>3</sup> Em sistemas de monitoramento como o Zabbix, *triggers* são regras ou condições predefinidas que definem quando um alerta ou ação deve ser disparado.

<sup>4</sup> Um script é um conjunto de comandos escritos em uma linguagem de programação que são executados por um interpretador.

cionalidades são especialmente úteis para a tomada de decisões baseada em dados (ZABBIX, 2024).

A Plataforma Zabbix oferece uma integração eficaz com diversas plataformas e sistemas, o que o torna uma escolha indicada para empresas que precisam monitorar muitos dispositivos e aplicativos, além de suportar o crescimento contínuo da infraestrutura monitorada sem perder a eficiência.

### 2.1.3 Metabase

O Metabase é uma plataforma de análise de dados e BI, lançada em 2014 por um grupo de desenvolvedores que tinha como objetivo criar uma ferramenta de análise acessível e de código aberto. Projetado para ser intuitivo e amigável, o Metabase foi desenvolvido por uma equipe que incluía os fundadores Sam Saffron e Tom Robinson, focando na democratização do acesso a dados, especialmente para usuários não técnicos (METABASE, 2024).

Sua principal funcionalidade é a criação de consultas e relatórios a partir de diversos bancos de dados, como MySQL, PostgreSQL, MongoDB. A plataforma oferece integração com uma vasta gama de fontes de dados, permitindo que empresas utilizem dados centralizados para criar *dashboards* interativos e personalizáveis. Além disso, o Metabase fornece diferentes tipos de visualizações, como gráficos de barras, mapas de calor e tabelas dinâmicas, adaptando-se às necessidades de visualização dos usuários (METABASE, 2024).

Uma das características mais notáveis do Metabase é sua interface voltada para usuários não técnicos. A interface gráfica é intuitiva, eliminando a necessidade de conhecimento em SQL para realizar consultas. Os usuários podem formular perguntas complexas de negócios e obter respostas visuais sem precisar depender de desenvolvedores ou analistas de dados, tornando a análise de dados mais acessível em toda a organização (METABASE, 2024).

Em termos de casos de uso, o Metabase é amplamente utilizado por pequenas e médias empresas em setores como varejo e tecnologia, especialmente para análises de comportamento do consumidor e monitoramento de desempenho de campanhas de marketing digital. Por exemplo, empresas podem utilizar o Metabase para identificar padrões de consumo em plataformas digitais e ajustar suas estratégias com base em dados visuais e de fácil interpretação (JONES, 2019).

A plataforma também oferece alertas automáticos e notificações, que podem ser configurados para monitorar métricas críticas e enviar alertas quando determinados parâmetros

forem atingidos. Apesar dessas funcionalidades, o Metabase enfrenta limitações em termos de escalabilidade e personalização mais profunda quando comparado a ferramentas mais complexas, como o Tableau<sup>5</sup> ou o Power BI. No entanto, ele continua sendo uma solução atrativa para organizações que priorizam uma implementação rápida e de baixo custo (SMITH; DOE, 2020).

Em termos de arquitetura, o Metabase segue um modelo cliente/servidor e pode ser executado em servidores baremetal<sup>6</sup> ou em contêineres, como Docker e Podman, proporcionando flexibilidade operacional. Suporta múltiplas plataformas e protocolos de comunicação com bancos de dados, como JDBC e ODBC, o que facilita sua integração com diferentes sistemas de TI. Essa arquitetura flexível torna o Metabase adaptável a ambientes com crescimento moderado de dispositivos e volumes de dados (METABASE, 2024).

O Metabase oferece uma solução poderosa para empresas que precisam de uma plataforma de BI fácil de usar, com funcionalidades voltadas para relatórios dinâmicos e monitoramento de dados em tempo real, mas com limitações de escalabilidade em grandes ambientes.

#### 2.1.4 Grafana

O Grafana é uma plataforma de visualização e monitoramento de dados desenvolvida em 2014, por Torkel Ödegaard, com o objetivo de fornecer uma solução flexível e de código aberto para a visualização de métricas. Inicialmente, foi projetado para integrar-se ao Prometheus, mas rapidamente evoluiu para suportar uma ampla gama de fontes de dados, transformando-se em uma das ferramentas mais populares para monitoramento de sistemas e análise de dados em tempo real (GRAFANA LABS, 2024). Essa plataforma tem sido amplamente utilizada por equipes de TI e operações para monitorar e visualizar dados de maneira eficiente, com uma interface altamente personalizável.

Uma das grandes vantagens do Grafana é sua arquitetura flexível, que permite integração com diversas fontes de dados, como Prometheus, InfluxDB, MySQL, PostgreSQL. Além disso, suporta a criação de *dashboards* dinâmicos que permitem o monitoramento em tempo real de métricas de desempenho. Ela oferece suporte a *plugins*, o que expande suas funcionalidades e personalização, facilitando a integração de novos tipos de visualização e fontes de dados (HILL, 2022). Essa ferramenta é especialmente eficaz em ambientes com grande volume de dados, tornando-se uma boa escolha para monitoramento de redes, infraestruturas de

<sup>5</sup> Tableau é uma ferramenta de visualização de dados amplamente utilizada em Business Intelligence e análise de dados. Para mais informações, consulte o site oficial: <<https://www.tableau.com/>>.

<sup>6</sup> Servidores baremetal são servidores físicos dedicados que não utilizam virtualização para hospedar sistemas operacionais ou aplicações oferecendo acesso direto ao hardware.

TI e aplicações empresariais.

Em termos de facilidade de uso, o Grafana se destaca por sua interface intuitiva e voltada para usuários com diversos níveis de experiência técnica. A plataforma permite que administradores criem *dashboards* interativos sem a necessidade de escrever código, o que a torna acessível tanto para profissionais técnicos quanto para usuários menos experientes. O sistema de criação de *dashboards* permite a combinação de diversas fontes de dados em uma única interface, facilitando a visualização de informações críticas em tempo real (GRAFANA LABS, 2024).

O monitoramento em tempo real é um de seus pontos fortes, além de ser amplamente utilizado por empresas de hospedagem web para monitorar a carga dos servidores e ajustar os recursos com base na demanda observada. Essa capacidade permite a detecção rápida de anomalias e a otimização de desempenho em infraestruturas complexas. Em centros de operações de rede, a plataforma se mostra essencial para o monitoramento contínuo do tráfego de internet e da saúde dos servidores, proporcionando uma visão centralizada das operações (GRAFANA LABS, 2024).

No que diz respeito à escalabilidade, o Grafana se adapta facilmente a grandes ambientes. Ele pode ser implementado em sistemas de grandes volumes de dispositivos e métricas, mantendo a eficiência com requisitos mínimos de sistema, como 512MB de RAM e 1 CPU. Isso possibilita a operação até em dispositivos com recursos limitados, o que a torna acessível em uma ampla variedade de cenários (GRAFANA LABS, 2024).

Outra característica importante do software é seu sistema de alertas e automação. Os usuários podem configurar notificações automáticas com base em determinados critérios, como a sobrecarga de CPU ou falhas em componentes críticos, permitindo uma resposta imediata e automática. Esses alertas são enviados por meio de canais diversos, como e-mail, Slack, ou Microsoft Teams, garantindo que a equipe responsável seja informada rapidamente sobre qualquer incidente (GRAFANA LABS, 2024).

O Grafana se diferencia de outras plataformas de monitoramento por sua integração com múltiplas fontes de dados, sua capacidade de criar visualizações customizadas com *dashboards* interativos, e sua escalabilidade para ambientes de grande porte. Essas características, aliadas à sua interface intuitiva, tornam-o uma escolha popular para o monitoramento de sistemas e análise de métricas.

### 2.1.5 SigNoz

O SigNoz é uma plataforma de observabilidade de código aberto desenvolvida com o objetivo de oferecer uma solução robusta para o monitoramento e rastreamento de desempenho das aplicações. Criado pela equipe da SigNoz Inc., o propósito do projeto é simplificar a observabilidade de sistemas distribuídos e infraestruturas complexas, oferecendo aos desenvolvedores uma visão clara sobre o comportamento e desempenho de suas aplicações (SIGNOZ, 2024). O SigNoz destaca-se por ser uma alternativa de código aberto a ferramentas como Datadog e New Relic, proporcionando um controle completo sobre os dados e uma integração profunda com o ambiente tecnológico da organização.

Em termos de funcionalidades e arquitetura, o SigNoz oferece suporte a vários de bancos de dados, incluindo MongoDB e MySQL, além de integrar-se facilmente com contêineres Docker e orquestradores como Kubernetes. Sua arquitetura modular e escalável torna o SigNoz adaptável ao crescimento contínuo das necessidades de monitoramento, permitindo a adição de novos nós de maneira simples e eficiente (SIGNOZ, 2024). A plataforma é indicada para monitoramento de infraestruturas dinâmicas, capturando dados críticos sobre a saúde das aplicações e fornecendo *insights* precisos para a resolução de problemas.

Uma característica central do SigNoz é sua facilidade de uso. A plataforma foi projetada para ser intuitiva, mesmo para usuários sem profundo conhecimento técnico. Sua interface gráfica permite que relatórios detalhados e *dashboards* interativos sejam gerados facilmente, facilitando a análise de dados complexos. Usuários podem acessar métricas e realizar rastreamentos distribuídos de forma simples, o que torna o SigNoz uma boa escolha para organizações que desejam democratizar o acesso à análise de dados (Restack Team, 2023).

Os casos de uso do SigNoz são variados. Em empresas de tecnologia, por exemplo, a plataforma é utilizada para monitorar o desempenho de APIs e microserviços, ajudando equipes de desenvolvimento a identificar gargalos e otimizar a eficiência das aplicações. Outro exemplo prático é o uso do SigNoz para monitorar infraestruturas de nuvem, onde as métricas são utilizadas para assegurar a disponibilidade e estabilidade de serviços críticos (TIWARI, 2021).

No que diz respeito às características técnicas, o SigNoz adota uma arquitetura “servidor-agente” para coleta e análise de dados, suportando protocolos amplamente utilizados como SNMP, ICMP, garantindo compatibilidade com uma vasta gama de dispositivos e sistemas. O SigNoz também inclui um sistema robusto de *triggers*, que permite configurar aler-

tas automáticos e ações proativas quando são detectadas anomalias ou desvios de desempenho (SIGNOZ, 2024).

O sistema de alertas e automação do SigNoz é eficiente, permitindo que os administradores configurem notificações personalizadas que são acionadas com base em eventos pré-determinados, como falhas no sistema ou queda de desempenho. Essa funcionalidade proporciona uma resposta rápida e automatizada ao usuário, ajudando a evitar falhas críticas e a manter o desempenho das aplicações estável (TIWARI, 2021).

Em termos de escalabilidade, o SigNoz foi projetado para crescer com o aumento das demandas de monitoramento. Sua arquitetura distribuída permite que a plataforma seja implantada em ambientes de grande porte, com muitos dispositivos monitorados, sem comprometer o desempenho. Isso o torna uma escolha interessante para organizações que precisam de uma solução de monitoramento escalável, capaz de lidar com infraestruturas em constante expansão (SIGNOZ, 2024).

O SigNoz oferece uma plataforma poderosa e flexível para o monitoramento de infraestruturas tecnológicas e aplicações. Sua capacidade de se integrar a múltiplas fontes de dados, personalizar *dashboards*, e oferecer suporte a alertas automatizados faz dele uma solução interessante para organizações.

### **2.1.6 Kibana**

O Kibana é uma plataforma de visualização de dados criada pela Elastic NV, projetada especificamente para trabalhar com o Elasticsearch. Lançado pela primeira vez em 2013, foi desenvolvido com o propósito de permitir que os usuários visualizassem e navegassem em grandes volumes de dados indexados no Elasticsearch. Além de oferecer uma maneira eficaz de transformar dados complexos em novas percepções visuais, plataforma está disponível como uma solução de código aberto. Entretanto possui uma versão comercial, chamada Kibana Enterprise, que inclui funcionalidades adicionais, como suporte técnico e mais opções de personalização (ELASTICSEARCH B.V., 2024).

Entre suas principais funcionalidades estão a capacidade de realizar consultas complexas, criar *dashboards* interativos e gerar uma variedade de visualizações, como gráficos de barras, tabelas dinâmicas e mapas geoespaciais. Essas visualizações podem ser facilmente combinadas em painéis dinâmicos, oferecendo uma visão abrangente dos dados operacionais e métricas do sistema. Além disso, suporta a criação de relatórios detalhados, que podem ser perso-

nalizados conforme as necessidades de diferentes equipes e setores da empresa (ELASTICSEARCH B.V., 2024).

Uma das grandes vantagens deste software é sua facilidade de uso, especialmente voltada para usuários não técnicos. Sua interface gráfica amigável e intuitiva permite que pessoas sem conhecimento profundo em programação ou análise de dados possam configurar consultas, explorar grandes volumes de dados e criar visualizações complexas. A interface centrada no usuário facilita a exploração de dados indexados no Elasticsearch, o que o torna acessível para uma variedade de usuários dentro de uma organização (TEAM, 2023).

No que diz respeito aos casos de uso, o Kibana é amplamente utilizado em ambientes organizacionais para monitoramento de *logs* e eventos em tempo real. Empresas de tecnologia, telecomunicações e finanças, por exemplo, empregam a plataforma para detectar anomalias e gerar diagnósticos rápidos sobre problemas em suas infraestruturas de TI. Outro exemplo prático é sua aplicação em sistemas de detecção de fraudes, onde os dados em tempo real são essenciais para identificar padrões suspeitos e tomar decisões rápidas (ITUH, 2023).

Em termos de características técnicas, foi adotada uma arquitetura modular que permite a personalização e a adição de *plugins*, facilitando sua extensão conforme as demandas específicas da organização. A plataforma oferece suporte a múltiplos protocolos, incluindo SNMP e ICMP, o que a torna compatível com diversos sistemas e dispositivos. O Kibana também é capaz de se integrar com outras ferramentas da *Elastic Stack*, como o Logstash e o Beats, para a coleta de dados (ELASTICSEARCH B.V., 2024).

Seu sistema de alertas e automação é altamente flexível. Ele permite configurar notificações automáticas para eventos críticos, como a ocorrência de falhas ou desvios nos parâmetros monitorados. Esses alertas podem ser acionados através de diferentes canais, como e-mail ou ferramentas de comunicação como o Slack, permitindo uma resposta rápida e automatizada para incidentes que possam comprometer o desempenho dos sistemas (ELASTICSEARCH B.V., 2024).

A escalabilidade da plataforma permite o monitoramento de milhares de dispositivos, sendo amplamente utilizado em empresas de grande porte e em infraestruturas distribuídas. Sua capacidade de lidar com grandes volumes de dados sem comprometer o desempenho o torna ideal para empresas que precisam de uma solução de monitoramento e visualização robusta, que possa crescer junto com as demandas de seus sistemas (ELASTICSEARCH B.V., 2024).

O Kibana se destaca por sua integração com diversas fontes de dados. Embora seja

especialmente projetado para trabalhar com o *Elasticsearch*, a plataforma permite a conexão com outros tipos de fontes, como bancos de dados relacionais (MySQL, PostgreSQL) e sistemas de monitoramento de métricas, como Prometheus e InfluxDB. Isso o torna uma ferramenta versátil para monitoramento e análise de dados complexos em diferentes contextos operacionais (ELASTICSEARCH B.V., 2024).

### 3 METODOLOGIA

O objetivo desta pesquisa foi identificar e avaliar softwares SaaS de código aberto adequados para visualização de dados para o CEIFA. Os softwares devem ser capazes de operar tanto em dispositivos de borda com recursos limitados quanto em plataformas baseadas na nuvem. A finalidade é garantir que esses softwares ofereçam estabilidade e integridade dos dados apresentados, além de proporcionar uma interface simples e intuitiva, que possa ser facilmente utilizada por usuários leigos.

A metodologia adotada combinou duas abordagens: bibliográfica e experimental. A pesquisa foi qualitativa e aplicada, com o objetivo de explorar e descrever as capacidades de diferentes softwares SaaS de código aberto para o CEIFA. Inicialmente foi realizada uma pesquisa bibliográfica para identificar as ferramentas potencialmente adequadas. Em seguida, foram realizados testes experimentais para avaliar o desempenho, a escalabilidade e o consumo de recursos dessas plataformas em dispositivos de borda e no ambiente em nuvem. Essa abordagem permitiu uma avaliação abrangente e fundamentada das soluções tecnológicas, assegurando que a plataforma escolhida atenda aos objetivos operacionais e técnicos estabelecidos.

O objetivo dos experimentos foi avaliar qual plataforma se adapta melhor aos requisitos do CEIFA, considerando os recursos limitados dos dispositivos de borda e a complexidade dos ambientes de nuvem. Para essa análise, cada uma das ferramentas selecionadas foi instalada e testada em dois cenários: na borda, utilizando um Raspberry Pi, e na nuvem, em uma máquina virtual.

Em cada cenário, as plataformas foram monitoradas durante dois dias consecutivos, registrando o consumo de memória RAM, processamento (CPU) e espaço em disco, além de verificações de disponibilidade e capacidade de suportar acessos simultâneos. Esses experimentos foram essenciais para identificar as plataformas que conseguem operar de forma estável e eficiente dentro das limitações de recursos, garantindo compatibilidade com a arquitetura do CEIFA.

A combinação de métodos bibliográfico e experimental permitiu uma avaliação detalhada e multidimensional. A pesquisa bibliográfica oferece uma base teórica sólida, identificando as melhores práticas e tecnologias emergentes, enquanto os testes experimentais proporcionam dados concretos sobre como essas tecnologias funcionam em cenários reais. Essa abordagem integrada assegura que a escolha final dos softwares foi baseada em evidências robustas,

garantindo a implementação de uma solução eficaz para o monitoramento e a visualização de dados.

O procedimento adotado nesta pesquisa incluiu desenvolvimento, coleta de dados, experimentação e testes. Inicialmente, foram analisados os softwares de *Software as a Service* compatíveis com o dispositivo de borda utilizado pelo CEIFA até o momento e os ambientes de nuvem compatíveis com os SaaS selecionados. Foram selecionados cinco softwares potencialmente compatíveis: Zabbix, Metabase, Grafana, SigNoz e Kibana. Cada software foi instalado e testado nos cenários descritos na Tabela 2 para avaliar desempenho e capacidade de escalabilidade dentro do escopo definido.

Os dados climáticos coletados foram simulados como os que o sistema CEIFA analisa e salva no banco de dados. As plataformas se conectam diretamente ao banco de dados para gerar gráficos e visualizações. Foram testados os comportamentos dos softwares em ambientes de borda e nuvem, observando seu desempenho em diferentes condições operacionais.

A fase de testes incluiu a sobrecarga de informações para verificar a capacidade das plataformas em lidar com grandes volumes de dados e acessos simultâneos. Foram avaliados critérios como custo computacional, melhor forma de hospedar as plataformas (borda ou nuvem), e a estabilidade em lidar com quantidades diferentes de dados visando a agricultura familiar. Após a validação experimental, os testes práticos envolveram simulações em um ambiente realista, garantindo que a plataforma escolhida atenda aos objetivos de estabilidade e escalabilidade dos cenários propostos. Esses passos garantem uma pesquisa abrangente e rigorosa, fornecendo uma base sólida para a seleção de uma solução tecnológica eficaz e eficiente para o monitoramento e visualização de dados na agricultura inteligente.

## 4 CENÁRIOS DE TESTE

Com o objetivo de avaliar o desempenho das plataformas SaaS na borda e na nuvem, foram criados três cenários de testes. Esses cenários fornecem métricas para avaliar a viabilidade de instalação e o desempenho das plataformas. Cada cenário foi devidamente analisado e testado nas plataformas mencionadas no referencial teórico.

### 4.1 CENÁRIOS

O cenário 1, denominado cenário inicial, é voltado para o uso de 20 sensores e 1 usuário com acesso à plataforma de visualização. Este cenário é adequado para agricultura familiar ou de pequeno porte, haja visto que nestes ambientes a área monitorada não é muito grande, além de que o uso de muitos sensores acarreta em um maior custo financeiro. Para fins de avaliação, estabeleceu-se que 20 sensores, o equivalente à metade das entradas digitais da placa do Raspberry Pi são o suficientes para este cenário. O acesso é limitado a um único usuário, considerando a natureza da agricultura familiar, onde poucas pessoas estão envolvidas no processo de produção.

O cenário 2, denominado cenário intermediário, envolve a utilização de 30 sensores. Estes sensores ocupam 30 conectores da placa do Raspberry Pi, adequando-se a uma propriedade de médio porte. O aumento no número de sensores permite coletar mais dados, para monitorar e otimizar diversos aspectos da produção. Devido à maior complexidade e volume de dados gerados, este cenário propõe o uso simultâneo de 3 usuários na plataforma de visualização. Isso facilita a colaboração entre os membros da equipe, permitindo uma análise mais eficiente e tomada de decisões mais informadas.

O cenário 3 prevê um uso de mais dispositivos, totalizando 40 sensores, que ocupam todas as entradas digitais disponíveis na placa do Raspberry Pi. Além disso, este cenário permite o uso simultâneo de 5 usuários na plataforma de visualização, possibilitando uma análise mais colaborativa dos dados coletados. Este cenário é projetado para operações familiares que, apesar de manterem um perfil de pequena escala, necessitam de um monitoramento mais intensivo de suas atividades agrícolas para melhorar a produtividade.

Um cenário maior que o cenário 3, foge do escopo deste trabalho, pois a inclusão de mais sensores exigiria o uso de uma placa com maior capacidade de processamento e um número ampliado de entradas digitais e/ou analógicas. Isso implicaria no uso de equipamentos

Tabela 2 – Cenários de Sensores e Usuários

Cenários		
Cenário	Sensores	Usuários
1	20	1
2	30	3
3	40	5

mais robustos e custosos, que não se alinham ao objetivo deste estudo, que é desenvolver soluções de baixo custo e de fácil acesso para produtores familiares. A complexidade adicional e os custos associados a essa expansão não são viáveis dentro das restrições e objetivos do projeto atual.

#### 4.2 TESTES

Os testes foram executados utilizando sensores virtuais, conectados através da Interface de Programação de Aplicações (API, do inglês *Application Programming Interface*) do Climatempo ([climatempo.com.br](http://climatempo.com.br)). A quantidade de sensores foi ajustada conforme cada cenário especificado. Esta API foi integrada por meio de um *script Python* (.py), que é executado de hora em hora como um *daemon*<sup>1</sup>, coletando as informações meteorológicas necessárias. A cada execução, os dados coletados alimentaram a quantidade de sensores especificada para cada cenário, garantindo uma simulação precisa e contínua.

Para cada cenário, os sensores virtuais realizam consultas à API do Climatempo a cada uma hora. Embora a frequência de consultas à API seja horária, os dados dos sensores virtuais foram inseridos no sistema por um *script Python* (.py) que roda a cada minuto. Isso permite uma avaliação detalhada do desempenho das plataformas SaaS em diferentes condições de carga. Os cenários foram testados de maneira a garantir o funcionamento de todos os aspectos do sistema, desde a coleta de dados até a sua respectiva visualização nas *dashboards*. Essa abordagem possibilitou identificar possíveis sobrecargas de dados e otimizações necessárias para a implementação na arquitetura do sistema CEIFA.

Os dados coletados da API do Climatempo, que simulam sensores em uma agricultura familiar, incluem a temperatura atual, a umidade relativa do ar, a velocidade e direção do vento. Esses dados fornecem uma base abrangente para monitorar e analisar as condições ambientais, permitindo uma gestão mais eficiente das atividades agrícolas.

<sup>1</sup> Um daemon é um programa de computador que executa como um processo em segundo plano, em vez de estar sob o controle direto de um usuário interativo.

Além disso, esses dados são atualizados capturando o último valor registrado e aplicando uma variação de 0,05% para mais ou para menos. Esse ajuste sutil simula mudanças contínuas e graduais nas condições ambientais, criando um cenário dinâmico de monitoramento para as plataformas de visualização, que assim podem ser avaliadas em um ambiente de dados realistas e em constante atualização.

Para o ambiente de nuvem, foi configurado um Servidor Virtual Privado (VPS, do inglês *Virtual Private Server*) em um datacenter privado, com uma configuração de entrada que inclui 2 vCPUs, 4 GB de memória RAM e 40 GB de armazenamento em disco. O Debian 12, última versão disponível do sistema operacional (2024), foi utilizado devido à sua confiabilidade e compatibilidade com o objetivo do CEIFA, oferecendo suporte robusto para as dependências e ferramentas necessárias ao sistema.

No dispositivo de borda foi utilizado um Raspberry Pi 3 Model B v1.2, lançado em 2015. Este modelo oferece uma solução de baixo custo, possui uma CPU quad-core de 1,2 GHz e 1 GB de memória RAM. Com 40 pinos GPIO<sup>2</sup> para conexão de sensores e suporte integrado a redes sem fio, o Raspberry Pi 3 atende às necessidades do sistema, permitindo a coleta e processamento local de dados antes do envio à nuvem.

As plataformas foram testadas tanto na borda quanto na nuvem, durante 2 dias consecutivos com 20 sensores, 2 dias com 30 sensores e 2 dias com 40 sensores, mantendo a mesma configuração para comparação de resultados. Durante o período de testes, foram monitoradas métricas como uso de memória, CPU e espaço em disco, assegurando que as plataformas sejam avaliadas de forma justa e precisa em diferentes cenários de carga.

Além disso, os dados de processamento, memória e uso de disco são coletados por um agente do Zabbix, que opera de forma eficiente, exigindo pouco processamento, menos de 1% de todo o processamento dos dispositivos. Essa abordagem permite validar a eficiência de cada plataforma monitorada, garantindo que o impacto no desempenho do sistema seja minimizado.

Os acessos simultâneos às plataformas foram simulados através de uma máquina virtual que está na mesma rede interna do Raspberry Pi, simulando o acesso via LAN<sup>3</sup>. Já o acesso ao ambiente de nuvem foi feito via IP público através da máquina virtual cliente. Todos

---

<sup>2</sup> GPIO (General Purpose Input/Output) são pinos de entrada e saída de uso geral em dispositivos eletrônicos, utilizados para conectar e controlar sensores, atuadores e outros componentes externos.

<sup>3</sup> LAN, do inglês Local Area Network, é uma rede de área local que conecta dispositivos em uma área limitada, como uma casa, escritório ou prédio. Essa rede permite a comunicação rápida entre dispositivos próximos e é usada para compartilhar recursos como arquivos, impressoras e internet.

os acessos fora feitos por navegadores com o cache desativado, respeitando a quantidade de acessos previstos para cada cenário.

#### 4.2.1 Zabbix

A instalação da plataforma Zabbix é simples e bem documentada no site oficial, tornando o processo acessível e direto tanto para o Raspberry Pi OS, quanto para uma máquina virtual. O Zabbix oferece uma versão oficial compatível com o Raspberry Pi OS, facilitando a implementação no dispositivos de borda. Já para a instalação em uma máquina virtual, com uma versão oficial e um instalador para o Debian 12, que garante compatibilidade e otimização no ambiente de nuvem. Com instruções claras e um instalador quase totalmente automatizado, a instalação da plataforma em ambos os ambientes é descomplicada, atendendo a diversos cenários com facilidade.

Seu método de coleta de dados funciona através de agentes que monitoram métricas e eventos de dispositivos e sistemas. Estas informações são armazenadas diretamente em seu próprio banco de dados. A plataforma utiliza tecnologias como Zabbix Agent, SNMP, e IPMI, para garantir um monitoramento contínuo. Vale destacar que o Zabbix é projetado para exibir *dashboards* e relatórios dos dados que ele próprio coleta, sem se conectar a bancos de dados de terceiros para a visualização de informações externas (ZABBIX, 2024).

Diante do fato de que o CEIFA salva todos as suas informações em um banco de dados SQLite e considerando que o Zabbix não possui suporte direto a ferramenta, foi implementada uma solução paliativa utilizando um *script* em Python. Este *script* extrai os dados do banco de dados e os envia para a plataforma através do método zabbix-trapper<sup>4</sup>. Isso permite que dados externos, sejam capturados e exibidos na plataforma Zabbix.

O *script* em Python conecta-se ao banco de dados SQLite e executa uma consulta SQL para capturar os dados mais recentes registrados. Para cada registro, verifica-se o tipo de dado, por exemplo, velocidade e direção do vento, associando esses tipos de dados a chaves específicas definidas dentro da plataforma. Em seguida, o *script* utiliza o comando zabbix-sender<sup>5</sup> para enviar os dados à plataforma.

Na plataforma Zabbix os dados são cadastrados como itens de coleta, que especificam as métricas monitoradas de cada dispositivo. Cada item possui uma chave única que

<sup>4</sup> Zabbix Trapper é um tipo de item no Zabbix que permite a recepção de dados enviados ativamente de uma fonte externa, em vez de coletá-los passivamente.

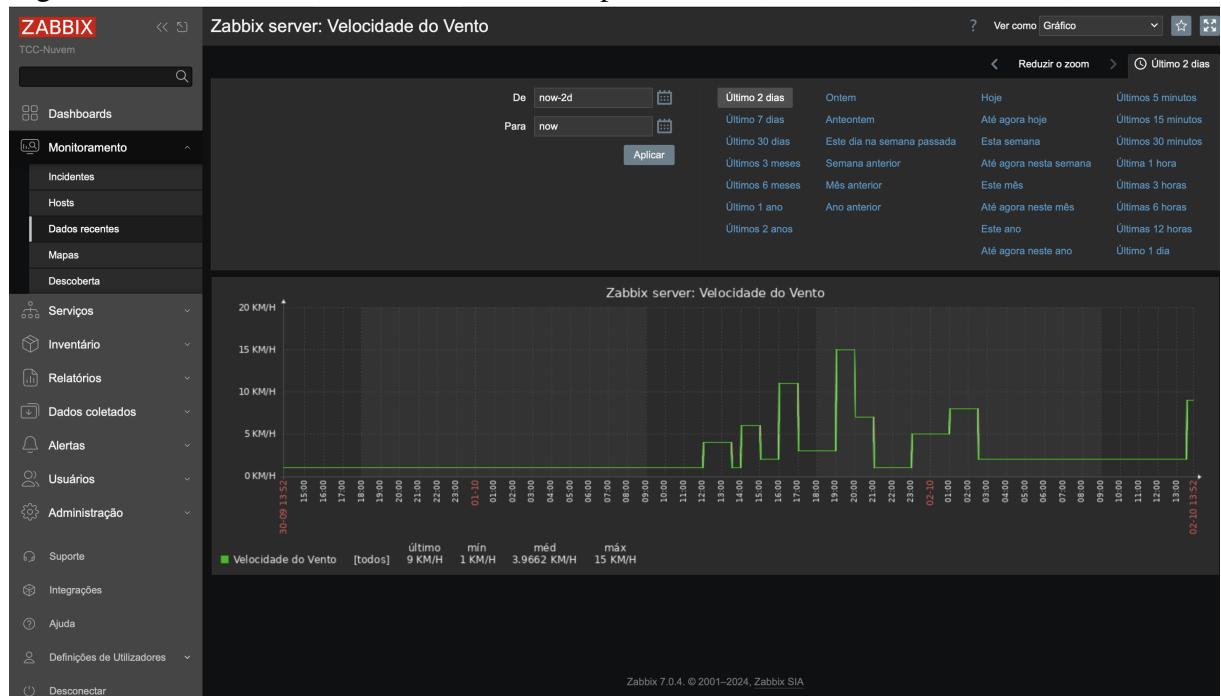
<sup>5</sup> zabbix-sender é uma ferramenta de linha de comando do Zabbix que permite o envio de dados diretamente para o servidor, usada para enviar métricas e informações de maneira ativa.

identifica o tipo de dado, como ‘*wind.velocity*’ para a velocidade do vento apresentado na Figura 3. Durante a fase de configuração, estes itens recebem parâmetros específicos, incluindo o tipo de coleta, a unidade de medida e a frequência de atualização. Essa configuração permite a recepção das informações através do *script*, que as adapta à coleta da plataforma.

Embora a Figura 3 demonstre o funcionamento de uma das coletas na plataforma, o *script* utilizado como solução alternativa mostrou-se ineficaz. Diversas coletas apresentaram erros, e alguns dados, como as condições do tempo em modo texto, não foram suportados pela plataforma. Devido a essas limitações, a plataforma não se mostrou adequada para o CEIFA.

A plataforma Zabbix apresentou diversas limitações, exigindo frequentes intervenções manuais, além de diversas adaptações complexas para se encaixar os cenários propostos. O uso de soluções alternativas, além de gerar aumento no trabalho de configuração e manutenção, também pode comprometer a confiabilidade do sistema em ambiente de produção. Desta forma o Zabbix não foi eleito como uma opção válida ao cenário proposto por este trabalho. Deste modo a plataforma foi desconsiderada antes de passar pelos testes de desempenho.

Figura 3 – Gráfico de Velocidade do Vento na plataforma Zabbix



Fonte: O Autor, 2024

#### 4.2.2 Metabase

A instalação do Metabase é simples e bem documentada em seu site oficial. A plataforma disponibiliza um arquivo .jar<sup>6</sup> que pode ser executado em qualquer ambiente compatível com Java, facilitando sua implementação em diversas configurações de hardware e software. No caso de instalação em uma máquina virtual ou em dispositivos de borda, como o Raspberry Pi, o processo é igualmente fácil e eficiente, permitindo rápida inicialização e configuração sem necessidade de ajustes complexos.

Para garantir que a plataforma funcione continuamente, ela foi configurada para rodar como um serviço de sistema, o que assegura sua operação em segundo plano de forma estável e autônoma. Executar como serviço evita a necessidade de manter uma janela de terminal aberta, o que interromperia a execução caso a janela fosse fechada. Devido a essa configuração, o Metabase inicia automaticamente junto com o sistema e reinicia em caso de falhas, garantindo disponibilidade e eliminando o risco de interrupções que comprometeriam a visualização de dados.

A conexão da plataforma com o banco de dados SQLite foi realizada utilizando o suporte nativo a esse tipo de banco. O arquivo do banco de dados foi alocado em um diretório onde fica acessível ao software de visualização, especificando seu caminho no painel de configurações. No Metabase, foi adicionada a fonte de dados, selecionando o SQLite como tipo de banco, o que permitiu a configuração da conexão de forma direta. As configurações padrões do Metabase, como a frequência de atualização das consultas e a interface padrão de visualização de dados, não foram alteradas.

A transformação dos dados em gráficos inciou com a criação das consultas SQL. Para cada sensor simulado foi necessário criar sua própria consulta, o que tornou a configuração da plataforma um pouco custosa. Estas consultas extraíam do banco de dados o tipo de dado coletado, o ID do sensor em questão, o parâmetro coletado e seu respectivo *timestamp*<sup>7</sup>.

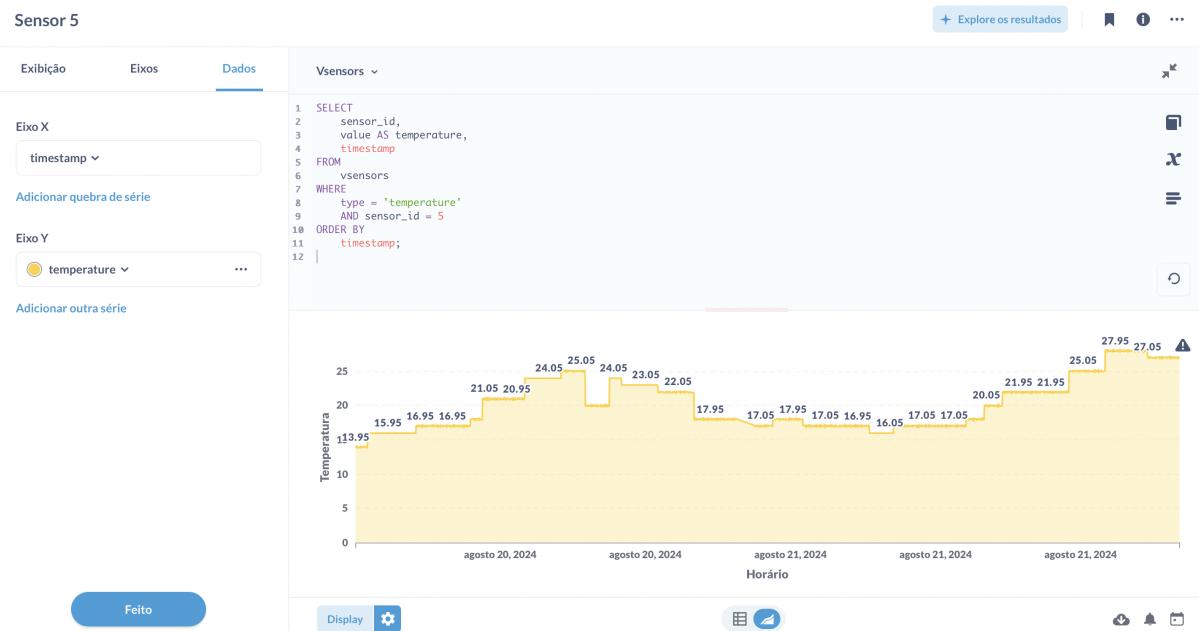
Após a execução da consulta SQL, os dados são exibidos em uma tabela. A plataforma oferece várias opções de tipos de gráficos para seleção, desde que seja compatível com os itens retornados. O gráfico do tipo “Área” possibilita a definição de parâmetros de visualização, com eixos X e Y: o eixo Y representa a variação da unidade coletada, enquanto o eixo X apresenta o período dos dados. Estas configurações podem ser vistas de forma clara na Figura 4.

---

<sup>6</sup> Um arquivo .jar (Java ARchive) é um pacote que contém arquivos compilados e recursos necessários para a execução de aplicações Java.

<sup>7</sup> Carimbo de data e hora que registra o momento exato de um evento ou de uma ação

Figura 4 – Tela de consulta SQL e configuração do gráfico no Metabase



Fonte: O Autor, 2024

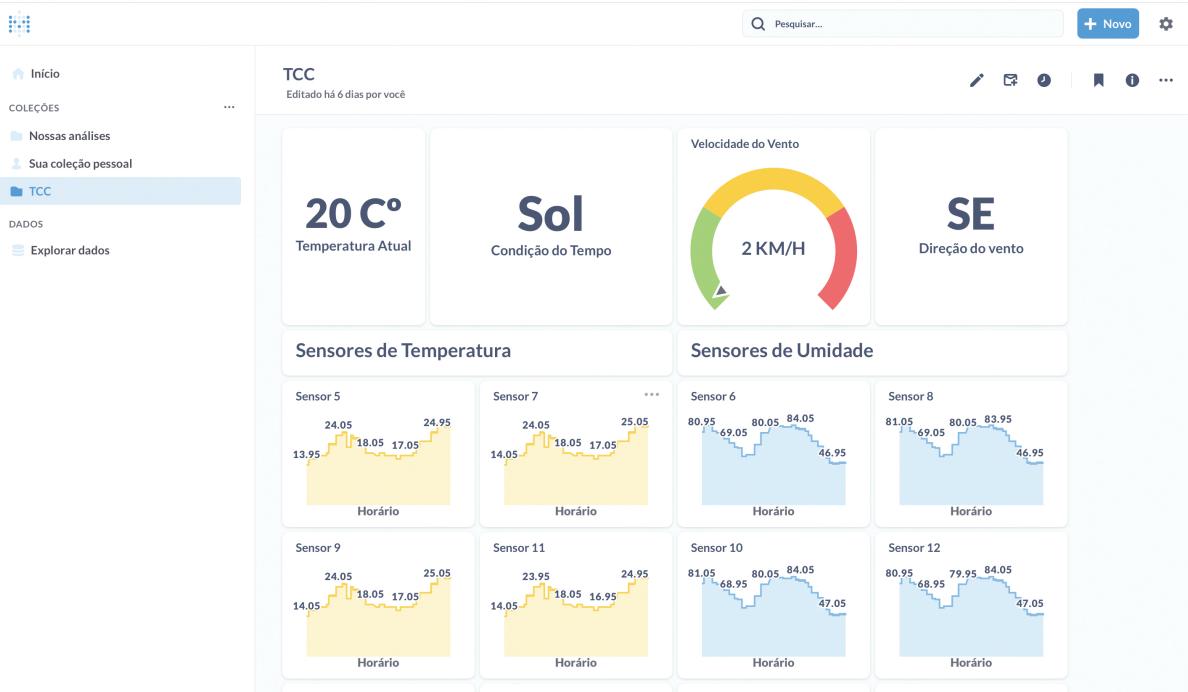
Os tipos de gráficos utilizados foram, o do tipo “Área” para sensores de temperatura e umidade, a fim de mostrar o histórico já registrado. Para direção do vento, condições de tempo, e temperatura atual utilizou-se o modo texto, que apresenta de forma clara e simples a informação na tela. No item velocidade do vento foi utilizado o “Indicador” que apresenta o dado atual similarmente a um velocímetro. Estes gráficos colocados na *dashboard* podem ser visualizados na Figura 5.

Com as consultas e configurações do gráfico ajustadas, o Metabase permite que adicionar a *dashboards* novas ou já existentes. Utiliza um sistema de blocos e o gráfico da consulta em questão pode ser facilmente ajustado em sua largura e altura na tela de visualização, como pode ser visto no apêndice A.

Durante a instalação da plataforma foi monitorado o consumo de recursos computacionais nos dois ambientes. Na nuvem, a instalação da plataforma usou em média 40% do processador virtual e 1200 MB (30% de 4GB), em média, da Memória RAM. Já durante o período de configuração de gráficos e *dashboard* o processamento ficou em média de 70% atingindo em alguns momentos críticos 100%. O consumo de memória RAM se manteve estável durante a configuração na média de 40% (1600 MB).

A instalação no Raspberry Pi OS demorou mais do que na nuvem devido aos recursos serem limitados e o poder de processamento menor. Durante a instalação o processamento

Figura 5 – Dashboard principal do Metabase



Fonte: O Autor, 2024

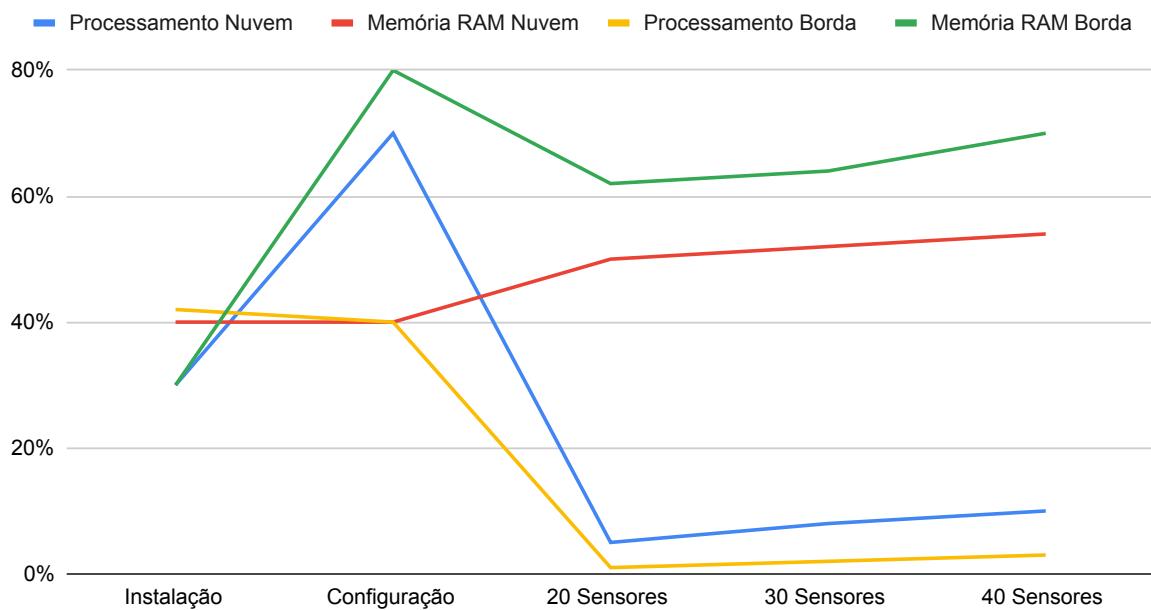
ficou e média 42% e consumo de memória RAM ficou em torno de 300 MB (30% de 1GB). Durante o período de configuração, o processamento utilizado foi 40% em média, e a memória RAM em 80% (800 MB), atingindo 100% em momentos críticos. Um aspecto negativo a ser ressaltado, foi a necessidade de configurar manualmente o Metabase em cada ambiente, pois as configurações realizadas na borda não puderam ser replicadas na nuvem, exigindo a configuração completa.

O Metabase possui um banco de dados próprio para armazenar itens como credenciais, preferências dos usuários e configurações. Este arquivo do banco fica junto ao diretório do .jar da plataforma. Ao todo, após ser instalada e configurada, a plataforma utilizou 800 MB de disco, sendo um ponto positivo pois na borda o armazenamento é limitado e na nuvem é pago.

Durante a instalação e configuração do Metabase, foi observado um uso considerável de processamento ultrapassando 70%. No entanto, durante os testes em cenários com sensores fixos, o consumo de CPU foi mínimo ficando abaixo de 20%, como indicado na Figura 6. O consumo de memória RAM, por outro lado, apresentou um comportamento elevado. Na nuvem, o consumo médio foi de 2080 MB (52% de 4GB), o que se mostrou aceitável devido à dedicação exclusiva da máquina virtual a essa aplicação. Já na borda, onde o Raspberry Pi possui apenas 1 GB de RAM, o consumo foi significativamente alto. Apesar disso, esse uso

elevado de RAM não prejudicou o desempenho do sistema, a única diferença perceptível foi o tempo de carregamento da página da *dashboard*, que demorou cerca cinco vezes mais o tempo na borda (43 segundos em média) em comparação com a nuvem (8 segundos em média). Após o carregamento inicial, a plataforma funcionou normalmente.

Figura 6 – Consumo de recursos do Metabase



Fonte: O Autor, 2024

O Metabase mostrou-se uma opção robusta e eficaz para a visualização de dados no CEIFA. Ele apresentou um bom desempenho tanto no dispositivo de borda quanto na nuvem. A plataforma mostrou-se capaz de lidar com o volume de dados coletados e os acessos simultâneos predefinidos, comprovando sua estabilidade em diferentes configurações. Durante os testes, o Metabase conseguiu processar e exibir informações de forma clara e acessível, mesmo em um dispositivo com recursos limitados, como o Raspberry Pi.

#### 4.2.3 Grafana

O processo de configuração do Grafana é também intuitivo e bem documentado, tornando-o acessível tanto para o Raspberry Pi OS quanto para máquinas virtuais. No Raspberry Pi OS, o Grafana já vem pré-instalado, o que facilitou sua implementação no dispositivo de borda e eliminou a necessidade da instalação manual. Na máquina virtual, a instalação foi simples, com uma versão oficial e um instalador dedicado para o Debian. A instalação e a configuração foram rápidas e quase totalmente automatizadas.

Para conectar o Grafana ao banco de dados SQLite, foi necessário instalar o *plugin*<sup>8</sup>: *Grafana SQLite Datasource*, pois o suporte ao SQLite não é nativo na plataforma. A instalação foi realizada diretamente na interface de administração da plataforma. Após a instalação, o *plugin* passou a estar disponível nas configurações como uma opção para integração com o banco de dados. Na configuração de fonte de dados, foi especificado o caminho do arquivo de banco de dados SQLite contendo os dados coletados.

O processo de transformação dos dados em gráficos segue uma lógica semelhante à do Metabase, começando pela criação de consultas SQL para cada sensor simulado. Cada consulta é personalizada para extrair informações específicas, como o tipo de dado coletado, o ID do sensor, o parâmetro monitorado e seu respectivo *timestamp*. Com a execução da consulta, os dados são exibidos em modo tabela, permitindo que o usuário selecione o tipo de gráfico mais adequado para a visualização.

Utilizou-se os gráficos do tipo “Barras” para sensores de temperatura e umidade, para mostrar o histórico já registrado. Para temperatura atual utilizou-se o “calibre de Barra”, que apresenta a informação como uma barra de progresso. No item velocidade do vento foi utilizado o “calibre” que apresenta o dado atual similarmente a um velocímetro, estes gráficos podem ser vistos no painel da plataforma (Figura 13). A informações como velocidade do vento e condição do tempo não foram possíveis de ser adicionadas, pois consultas em SQL que retornavam em modo texto não foram suportadas pela plataforma. Para compensar estes sensores que não foram utilizados, adicionou-se mais um de umidade e temperatura, mantendo a quantidade de sensores e dados previstos nos cenários.

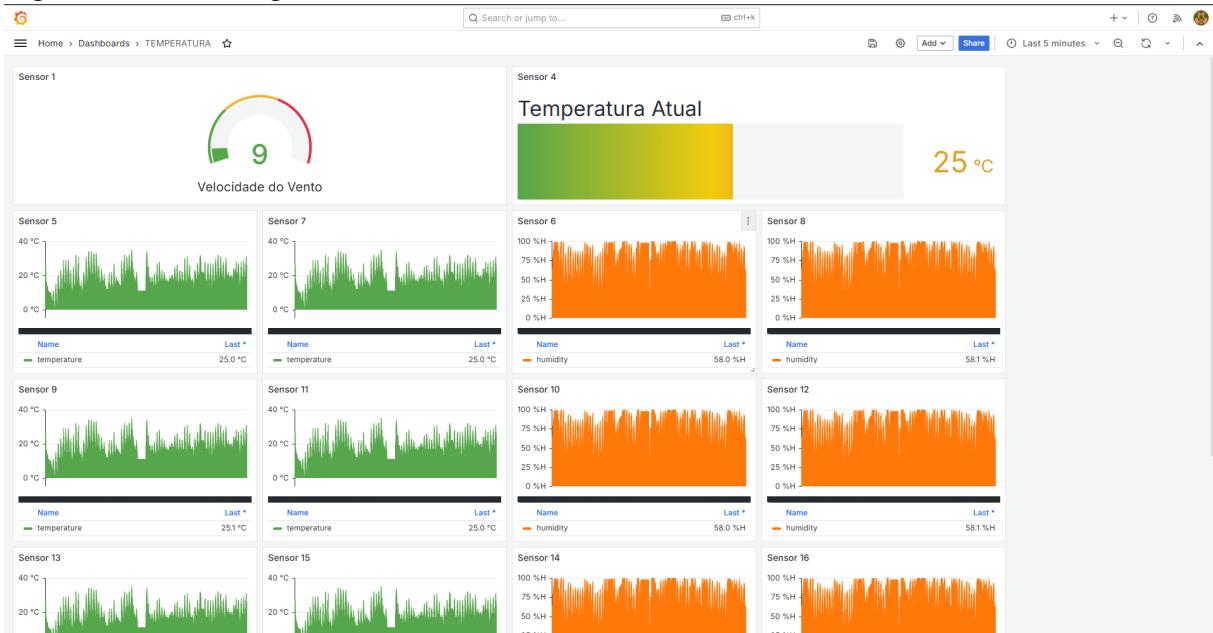
Após definir e executar as consultas SQL, os gráficos gerados podem ser organizados em painéis, com ajustes detalhados de tamanho, posição e tipo de visualização. Cada gráfico no painel pode ser configurado individualmente para mostrar informações específicas, com opções para ajustar os eixos, escalas e filtros. Essa flexibilidade permite que os painéis sejam adaptados às necessidades específicas de monitoramento.

Assim como no processo de instalação do Metabase, o consumo de recursos computacionais também foi monitorado durante a configuração do Grafana. Na nuvem a instalação da plataforma usou em média 10% do processador virtual, e 2080 MB (52% de 4GB) em média da Memória RAM. Já durante o período de configuração de gráficos e painéis o processamento ficou em média de 5%, e o consumo de memória RAM em média de 50% (2000 MB).

---

<sup>8</sup> Um plugin é um módulo ou extensão que adiciona funcionalidades extras a um software principal, permitindo sua personalização e expansão de recursos.

Figura 7 – Painel de gráficos do Grafana



Fonte: O Autor, 2024

No dispositivo de borda, não foi necessária a instalação do Grafana, pois ele já vem pré-instalado no Raspberry Pi OS. Durante a configuração, o uso médio de processamento ficou em 40%, enquanto o consumo de memória RAM manteve-se em 600 MB (60% de 1GB). Um ponto positivo desta plataforma é que, após montar as consultas e painéis na nuvem, foi possível exportar a configuração como um arquivo YAML<sup>9</sup> e importá-lo diretamente no dispositivo de borda. Esse recurso simplificou significativamente o processo de configuração, eliminando a necessidade de recriar consultas e painéis manualmente.

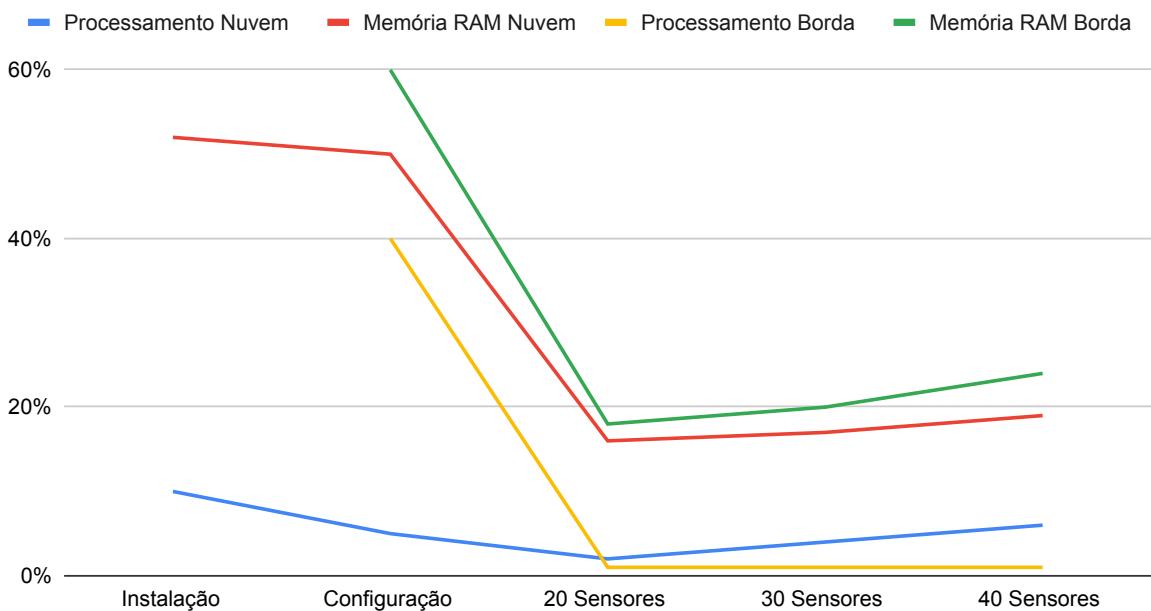
O Grafana também possui seu próprio sistema de armazenamento para salvar configurações, credenciais de usuários, preferências e dados de painéis. Após instalação e configuração inicial, a plataforma ocupa cerca de 200 MB de espaço em disco, o que é vantajoso em dispositivos de borda com armazenamento limitado e em ambientes de nuvem onde o armazenamento tem custo.

Semelhante ao Metabase, durante a instalação e configuração o uso de processamento foi mais elevado, utilizando mais de 50% (500 MB) da memória RAM. Entretanto, nos testes em cenários com sensores fixos, o processamento não foi utilizado (Figura 6). O consumo da memória RAM ficou em média de 220 MB (22%) em ambos os cenários, sendo um consumo considerado baixo. O tempo de carregamento nos clientes também foi maior quando acessava-

<sup>9</sup> YAML (Ain't Markup Language) é um formato de serialização de dados, de fácil leitura e escrita, utilizado principalmente para configuração de arquivos.

se o dispositivo de borda, em torno de 45 segundos. Já na nuvem, o tempo de carregamento não ultrapassou de 3 segundos, vale ressaltar que após o carregamento completo da página a plataforma permite a atualização automática sem precisar ficar recarregando o navegador, durante estas atualizações os dois cenários tiveram tempos de resposta iguais e rápidos.

Figura 8 – Consumo de recursos do Grafana



Fonte: O Autor, 2024

A plataforma Grafana se mostrou robusta e viável para atender às necessidades de monitoramento do CEIFA, principalmente devido ao seu baixo custo computacional e à facilidade de configuração. A compatibilidade com o Raspberry Pi OS e sua pré-instalação no dispositivo de borda simplificaram a implementação. Além disso, a possibilidade de exportar a configuração da nuvem para a borda através de um arquivo YAML foi um grande diferencial, permitindo replicar consultas e painéis de forma rápida, sem necessidade de retrabalho. O consumo moderado de recursos computacionais tornou o Grafana uma opção elegível para cenários com recursos limitados, oferecendo uma boa experiência e um tempo de resposta satisfatório em ambos os ambientes.

#### 4.2.4 SigNoz

A instalação do SigNoz em um ambiente auto-hospedado apresentou certos desafios devido a documentação para esse tipo de implantação ser limitada. O SigNoz foi projetado para

operar exclusivamente em contêineres, o que restringe a flexibilidade de instalação em sistemas que não adotam essa tecnologia.

A utilização de contêineres, apresenta uma limitação importante no contexto do Raspberry Pi OS, já que essa plataforma é restrita com certas arquiteturas de contêineres, principalmente devido ao uso de processadores ARM. Essa incompatibilidade torna inviável a instalação do SigNoz diretamente no dispositivo de borda, reduzindo as opções de monitoramento neste contexto com recursos limitados. Mesmo no ambiente de nuvem, o uso de contêineres para o SigNoz torna a configuração mais complexa e trabalhosa, exigindo a gestão de múltiplos serviços e dependências. Esse processo demanda maior conhecimento técnico e pode impactar o tempo de implementação e manutenção, especialmente nesta pesquisa que prioriza simplicidade e baixo custo operacional.

Devido às limitações identificadas, como a dependência de contêineres e a incompatibilidade com o Raspberry Pi OS, a plataforma requer adaptações complexas para operar nos cenários propostos. Essas restrições aumentam o risco de instabilidade em ambientes de produção. Assim, o SigNoz foi considerado inadequado para os objetivos deste trabalho e foi desconsiderado antes de passar pelos testes de desempenho de dois dias em cada cenário simulado, conforme descrito na Tabela 2.

#### 4.2.5 Kibana

A instalação do Kibana é simples, com um instalador personalizado para cada versão de sistema operacional disponível no site oficial da plataforma. Além disso a instalação da software é totalmente automática e rápida não exigindo muitos conhecimentos técnicos. Sua interface é muito similar ao Grafana, apesar desta informação não ser citada oficialmente.

As fontes de dados são exclusivamente integradas ao Elasticsearch, que atua como o motor de busca e banco de dados para armazenar e indexar os dados visualizados na plataforma. Devido a esse vínculo com o Elasticsearch o Kibana, não é compatível com outros bancos de dados, como o SQLite. Similar ao que aconteceu com a plataforma Zabbix, foi testada uma solução paliativa, que utiliza um *script* para transferir os dados para o motor do Elasticsearch.

Apesar da facilidade de instalação, a plataforma apresentou diversas limitações no contexto desta pesquisa, principalmente devido à sua dependência do Elasticsearch como fonte de dados. Mesmo tentando aplicar uma solução paliativa, com a criação de um *script* para transformar os dados do SQLite para o formato compatível com o Elasticsearch, a implementação

mostrou-se complexa, demandando elevado custo computacional e operacional. Mesmo após múltiplos testes e ajustes, a solução não apresentou efetividade em nenhum tipo de sensor. A necessidade de intervenção manual constante e as limitações de integração com outras fontes de dados levaram à decisão de não eleger o Kibana como uma opção viável para o CEIFA, pois estes desafios comprometeriam a estabilidade e escalabilidade do sistema.

#### 4.3 DISCUSSÃO GERAL

Os testes destacam o processo de avaliação das plataformas *SaaS* de código aberto para monitoramento e visualização de dados no sistema CEIFA, com testes conduzidos em cenários variados de operação na borda e na nuvem. Esses testes foram estruturados para avaliar a capacidade de cada plataforma em lidar com diferentes volumes de sensores e acessos simultâneos, com uma análise detalhada de métricas de desempenho, como o uso de memória, processamento e armazenamento em disco. O objetivo foi identificar as opções que melhor equilibrassem simplicidade, estabilidade, baixo custo operacional e computacional, atendendo às necessidades da agricultura inteligente em pequenos e médios ambientes agrícolas.

Durante esse período, o Zabbix, o SigNoz e o Kibana, revelaram-se inviáveis para o contexto do CEIFA, devido ao elevado custo operacional, à complexidade de configuração e às limitações de compatibilidade. O Zabbix, depende de *scripts* adicionais para transformar e adaptar os dados ao formato exigido. No entanto, os *scripts* não se mostraram efetivos, gerando um esforço manual significativo e reduzindo a confiabilidade. O SigNoz, mostrou-se restrito ao uso de contêineres, o que não apenas elevou a complexidade de implementação, mas também inviabilizou seu uso direto no dispositivo de borda, devido a incompatibilidade com a arquitetura ARM. Já o Kibana, apesar de uma instalação inicial facilitada, exige o uso exclusivo do Elasticsearch como fonte de dados, o que demandou uma complexa transformação dos dados do SQLite, a qual não foi efetiva e aumentou muito o custo operacional. Essas limitações comprometem a estabilidade e a escalabilidade, levando à decisão de descartar essas plataformas.

As plataformas Grafana e Metabase destacaram-se como soluções adequadas para o CEIFA. O Grafana, apresentou vantagens significativas, pois vem pré-instalado no Raspberry Pi OS, eliminando a necessidade de instalação manual na borda. Além disso, sua capacidade de exportar configurações via arquivo YAML facilitou a replicação de painéis entre os ambientes de borda e nuvem, otimizando o tempo de configuração e reduzindo o retrabalho. O Metabase, embora tenha demonstrado bom desempenho e confiabilidade, exigiu configurações manuais de

consultas e gráficos em cada ambiente, aumentando a complexidade e o tempo de configuração. No entanto, o Metabase ofereceu um diferencial ao suportar a visualização de dados de coletas em modo texto, como a direção do vento e condições do tempo, algo que não foi possível no Grafana.

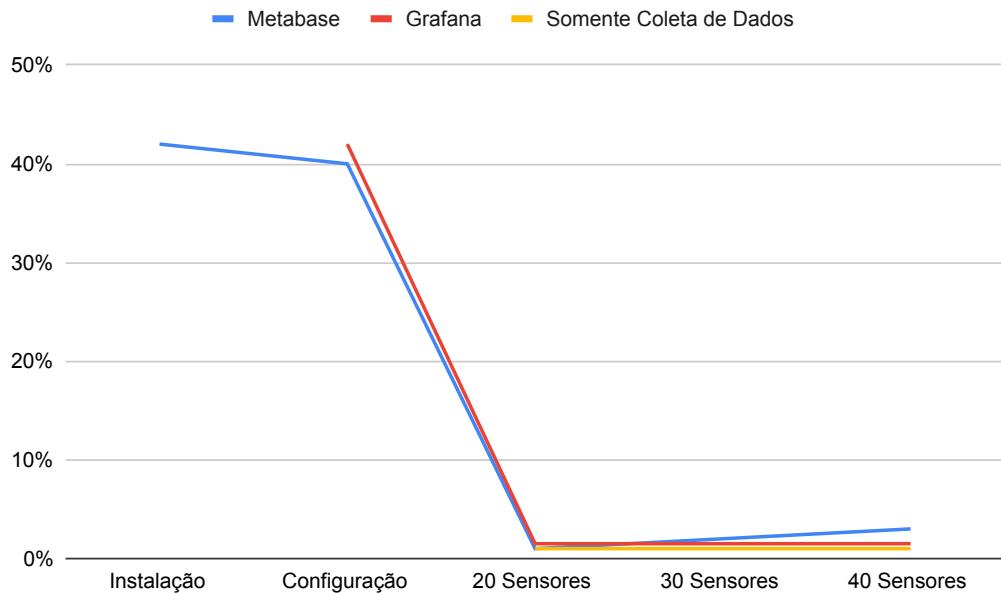
Para fins de comparação, durante o período de testes, os cenários foram também simulados sem a presença de nenhuma plataforma de visualização ativa, considerando apenas o funcionamento das coletas de dados. Esse procedimento visou isolar o impacto computacional das plataformas sobre o sistema, permitindo um comparativo entre o consumo de recursos nas condições com e sem a carga adicional das plataformas de visualização. Ao monitorar o uso de CPU, memória e armazenamento exclusivamente das coletas de dados, foi possível estabelecer uma linha de base, de forma que as métricas de desempenho das plataformas pudessem ser analisadas com precisão em relação a um cenário de coleta.

O Metabase apresentou um consumo de processamento consideravelmente mais elevado na borda em comparação com a nuvem. Durante a instalação e configuração, o processamento no dispositivo de borda permaneceu próximo de 40%, aumentando significativamente nos testes com sensores, especialmente quando utilizados 40 sensores, onde atingiu picos de 70%, como mostra a Figura 9. Em contraste, na nuvem, o uso de CPU foi mais estável, com um valor médio de 30% nos mesmos cenários de teste (Figura 10). Como referência adicional, o teste sem plataforma utilizou um processamento de 1% em ambos os ambientes, evidenciando o impacto significativo da plataforma Metabase sobre a carga de CPU, especialmente na borda.

Já no Grafana, o consumo de processamento na borda foi mais moderado em comparação ao Metabase, com uma média de 20% durante a configuração e 30% ao lidar com 40 sensores, como mostra a Figura 9. Na nuvem, o processamento foi ainda mais eficiente como demonstra a Figura 10, oscilando em torno de 15% mesmo com o máximo de sensores, destacando-se como uma opção que demandou menos processamento. Os testes de coleta de dados sem a plataforma instalada indicaram um consumo mínimo de CPU de 1%, evidenciando que a maior parte do uso de processamento durante a execução do Grafana está associada à execução e visualização dos dados, especialmente no ambiente de borda.

Ao comparar os dados apresentados nas Figuras 9 e 10, o Grafana se mostrou mais eficiente em termos de processamento em ambos os cenários, consumindo menos CPU em relação ao Metabase. A diferença foi mais significativa no ambiente de borda, onde o Metabase ultrapassou os 70% de uso de CPU com 40 sensores, enquanto o Grafana manteve-se em torno

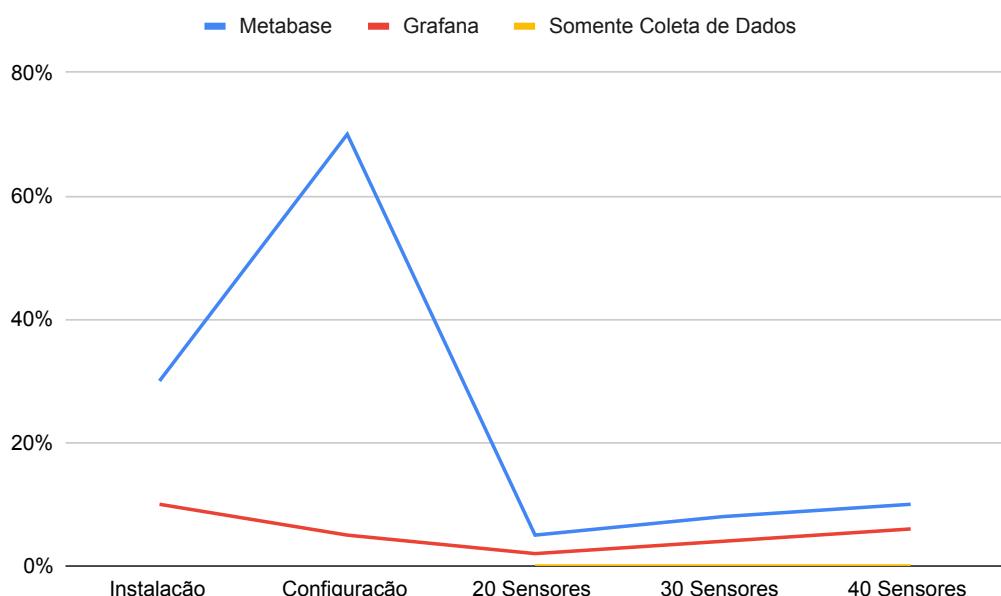
Figura 9 – Consumo de Processamento na borda



Fonte: O Autor, 2024

de 30%. No quesito processamento, a plataforma Grafana mostrou ter um bom desempenho utilizando 40% a menos do que o Metabase.

Figura 10 – Consumo de Processamento no ambiente de Nuvem



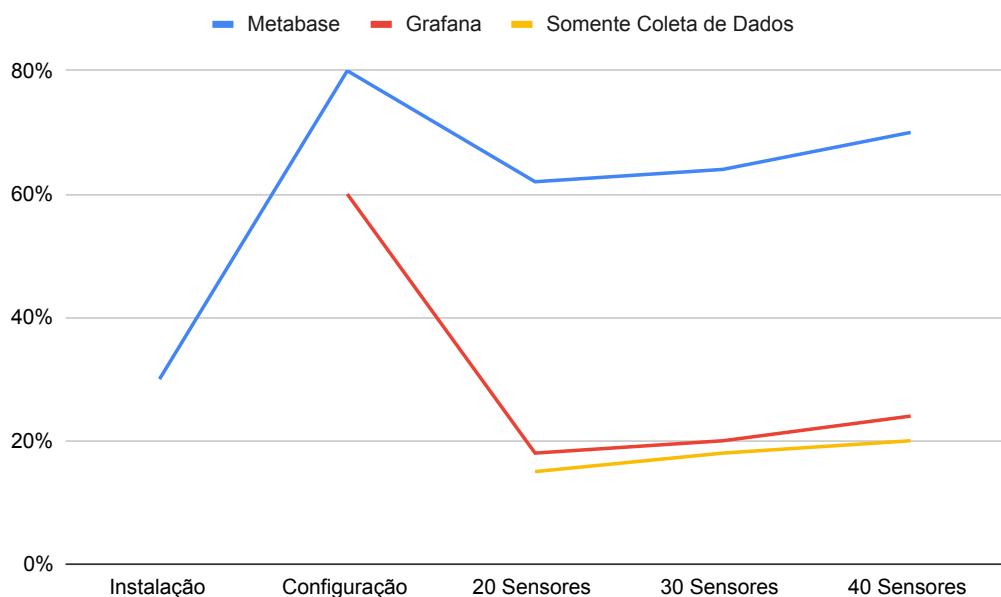
Fonte: O Autor, 2024

Em relação ao uso de memória RAM, o Metabase demonstrou um consumo elevado em ambos os cenários, principalmente no ambiente de borda. Como apresentado na Figura 11, o uso de memória RAM atingiu 600 MB (60% de 1GB) já na fase de configuração e ultrapassou

80% (800 MB) durante o teste com 40 sensores. Na nuvem, o Metabase manteve um consumo mais estável, variando entre 1600 MB (40%) e 2400 MB (60%) conforme o número de sensores, com picos durante a configuração inicial. Nos testes apenas com coleta de dados, o uso de memória RAM foi significativamente menor, indicando que o alto consumo da execução da plataforma.

O Grafana apresentou um uso menor de memória RAM em comparação ao Metabase, principalmente na borda. A Figura 11 mostra que o uso médio de RAM variou entre 300 MB (30%) e 500 MB (50%), mesmo com o aumento do número de sensores, enquanto na nuvem (Figura 12) o consumo manteve-se estável em torno de 200 MB (20%) a 400 MB (40%). Os testes somente com coleta de dados apresentaram um uso reduzido de RAM em ambos os ambientes, evidenciando que o impacto do Grafana sobre o consumo de memória é menor que o do Metabase.

Figura 11 – Consumo de Memória RAM na borda



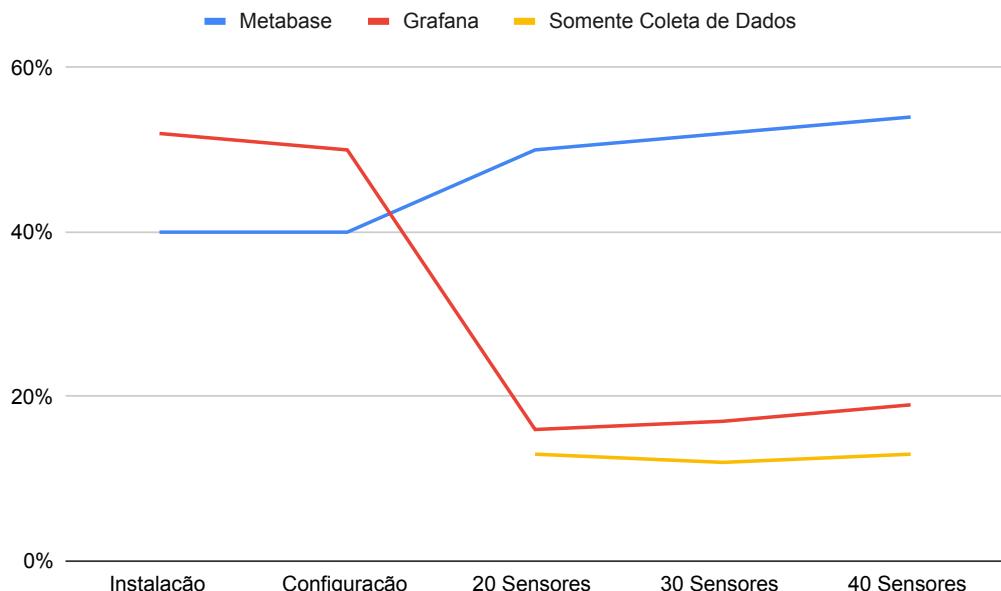
Fonte: O Autor, 2024

Ao comparar as duas plataformas no quesito consumo de memória RAM, o Grafana novamente demonstrou um desempenho superior ao Metabase, onde o uso de RAM do Metabase alcançou níveis críticos entre 700 MB (70%) e 800 MB (80%) podendo comprometer o desempenho do dispositivo, causando lentidão e travamento da plataforma e de outros processos. Enquanto isso o Grafana manteve-se dentro de limites mais aceitáveis (menos de 40%). Esta diferença é mais visível nos testes com maior número de sensores, onde o Metabase exigiu mais de 80% da RAM disponível na borda, em contraste com o Grafana, que permaneceu

em 50% ou menos. Isto fica bem visível no Figura 11. Esse desempenho torna o Grafana uma escolha viável em dispositivos com restrições de memória RAM.

Analizando o consumo dos recursos computacionais CPU e memória RAM, observa-se que o Grafana se destaca pela eficiência e menor consumo destes recursos, especialmente no dispositivo de borda, onde o Metabase se mostrou mais exigente tanto em processamento quanto em memória. Vale ressaltar que nas Figuras 9 e 11 não é apresentado o consumo de recursos do Grafana durante a instalação pois, como mencionado, ele já vem pré instalado no dispositivo de borda. Ambos apresentaram compatibilidade com os cenários de teste, entretanto o Metabase exigiu maior custo computacional para manter a desempenho adequado na borda, enquanto o Grafana utilizou uma configuração menor.

Figura 12 – Consumo de Memória RAM no ambiente de Nuvem



Fonte: O Autor, 2024

O uso de espaço em disco foi notavelmente diferente entre as plataformas, com o Metabase ocupando aproximadamente 800 MB devido ao seu banco de dados interno para armazenar configurações e credenciais. Já o Grafana utilizou cerca de 200 MB, refletindo uma estrutura menor e modular. O pouco consumo de disco torna o Grafana apropriado para ambientes com restrições de armazenamento, enquanto o Metabase oferece uma solução completa para cenários com maior disponibilidade de espaço em disco.

Para o CEIFA, a plataforma Metabase se destaca por sua capacidade de lidar com consultas complexas e por oferecer uma interface mais completa para a visualização de dados, embora exija mais recursos computacionais. Em contrapartida, o Grafana apresenta-se como

uma solução mais prática e eficiente em termos de consumo de recursos, mostrando-se ideal para ambientes com restrições de processamento e memória, como dispositivos de borda.

## 5 CONCLUSÃO

O CEIFA é um detector de anomalias híbrido, desenvolvido para detectar anomalias em dados no contexto agrícola inteligente. Porém, esse detector não possui uma interface gráfica que permita a visualização e o monitoramento dos dados coletados. Este trabalho, avaliou e testou diferentes soluções para visualização de dados capazes integrar-se ao CEIFA, com o objetivo de oferecer aos usuários uma interface intuitiva e eficiente para monitorar as condições do ambiente agrícola.

O desenvolvimento de uma aplicação personalizada para visualização de dados mostrou-se inviável devido aos elevados custos financeiros e computacionais envolvidos. Criar uma solução própria requer uma equipe especializada, investimentos e um longo período de desenvolvimento, seguido por processos de testes e validação para garantir o pleno funcionamento da plataforma. Esses fatores contrastam com o objetivo do CEIFA de operar com custos financeiros e computacionais reduzidos. Assim, o uso das plataformas Grafana, Metabase, Zabbix, SigNoz e Kibana, todas consolidadas, gratuitas e com funcionalidades robustas, mostrou-se uma alternativa mais prática e econômica, atendendo aos requisitos operacionais de baixo custo deste projeto.

Os testes das plataformas foram realizados em três cenários distintos, cada um configurado com diferentes quantidades de sensores e usuários simultâneos, simulando pequenas e médias propriedades de agricultura familiar. Essa abordagem buscou não apenas avaliar o desempenho das plataformas na borda e na nuvem, mas também investigar a viabilidade de integração com o CEIFA. Em cada cenário, foram coletadas métricas detalhadas de consumo de CPU, memória RAM e espaço em disco, proporcionando uma análise completa dos recursos exigidos em diferentes condições de operação.

A escolha desses cenários foi fundamentada na necessidade de verificar o comportamento das plataformas diante de volumes de dados e níveis variados de acessos simultâneos. Essa configuração permitiu observar o desempenho e a capacidade de cada plataforma em adaptar-se ao ambiente específico do CEIFA, validando sua adequação para o monitoramento agrícola em diferentes configurações e contextos operacionais.

Na fase de testes, as plataformas Zabbix, SigNoz e Kibana foram avaliadas, mas demonstraram-se inadequadas para o contexto do CEIFA devido a diversos obstáculos técnicos e operacionais. O Zabbix exigiu *scripts* adicionais para adaptar os dados ao formato exigido,

o que comprometeu a confiabilidade, uma vez que os *scripts* não funcionaram de forma consistente. O SigNoz depende exclusivamente de contêineres, elevou significativamente a complexidade de implementação e revelou-se incompatível com o dispositivo de borda Raspberry Pi devido à sua arquitetura ARM. No caso do Kibana, embora a instalação inicial tenha sido simplificada, a exigência do Elasticsearch como única fonte de dados impôs a necessidade de conversões complexas do SQLite, processo que não se mostrou eficaz e elevou os custos operacionais e computacionais. Essas restrições tornaram estas plataformas inviáveis para o CEIFA nos objetivos deste trabalho.

As plataformas Grafana e Metabase mostraram-se mais adequadas para atender às necessidades do CEIFA. O Grafana destacou-se pela vantagem de já vir pré-instalado no Raspberry Pi OS, eliminando a necessidade de instalação manual na borda e facilitando a implementação. Sua capacidade de exportar e importar configurações por meio de arquivos YAML também agilizou a replicação de painéis entre borda e nuvem, reduzindo o retrabalho. Já o Metabase, embora exigisse configurações manuais de consultas e gráficos em cada ambiente, alcançou bom desempenho e robustez, além de suportar a visualização de dados textuais, como direção do vento e condições climáticas, uma funcionalidade que o Grafana não ofereceu.

Durante os testes comparativos, o Metabase e o Grafana apresentaram comportamentos distintos em relação ao consumo de CPU e memória RAM. O Metabase, apesar de ter uma instalação inicial com menor custo computacional, mostrou-se mais exigente durante os cenários com sensores ativos, alcançando até 70% de uso de CPU na borda e 30% na nuvem. O Grafana, por outro lado, utilizou mais memória RAM durante a instalação chegando a 60%, entretanto uma vez configurado, apresentou um consumo significativamente menor durante os testes, com 30% de uso de CPU na borda e apenas 15% na nuvem. Como a instalação é uma etapa única, o impacto computacional menor do Grafana durante a execução faz dele uma escolha mais vantajosa para ambientes de operação contínua, especialmente em dispositivos de borda.

No quesito memória RAM, o Metabase novamente demandou mais recursos, alcançando até 80% de uso de RAM na borda em cenários com 40 sensores, enquanto na nuvem ficou em torno de 50%. O Grafana, em contraste, manteve-se dentro de um uso mais moderado de RAM, em torno de 40% na borda e 30% na nuvem. Esse perfil de consumo torna o Grafana mais adequado para ambientes com restrições de memória, onde o Metabase pode enfrentar limitações.

Enquanto o Metabase oferece uma solução mais robusta e completa, permitindo visualizações textuais adicionais, ele exige um custo computacional mais elevado para manter a operação em dispositivos de borda. O Grafana, com seu menor consumo de CPU, RAM e espaço em disco, mostrou-se ideal para cenários que priorizam eficiência em ambientes com hardware limitado.

No contexto do CEIFA, a escolha entre Metabase e Grafana depende das necessidades específicas de cada operação e da disponibilidade de recursos computacionais. O Metabase é uma excelente opção para quem busca uma plataforma com visualizações mais completas e detalhadas, especialmente em situações onde consultas complexas e diversidade de gráficos são necessárias. No entanto, essa funcionalidade mais robusta vem com o custo de um consumo computacional elevado, exigindo mais processamento e memória, o que pode ser um fator limitante em dispositivos de borda com restrições computacionais.

Por outro lado, o Grafana se destaca como a alternativa ideal para quem prioriza desempenho e economia de recursos. Seu baixo custo de processamento e memória faz com que seja adequado para ambientes onde a eficiência é essencial, oferecendo uma experiência de visualização de dados eficiente sem sobrecarregar a infraestrutura. Assim, para configurações do CEIFA com recursos limitados, o Grafana atende de forma satisfatória, enquanto o Metabase se mantém uma escolha atraente para ambientes que possam arcar com uma carga computacional maior para obter gráficos e consultas mais detalhados.

A análise dos ambientes de borda e nuvem permitiu compreender as particularidades e vantagens de cada opção, considerando a infraestrutura disponível e o consumo de recursos. Na borda, a pré-instalação do Grafana na Raspberry Pi OS demonstrou-se uma solução prática para reduzir custos de instalação e simplificar o gerenciamento, enquanto nuvem ofereceu uma configuração robusta com maior capacidade de processamento para a execução das plataformas. Esta comparação evidenciou as condições ideais para hospedar o sistema em ambos os cenários, maximizando a eficiência e adequação ao contexto agrícola do CEIFA.

As plataformas de visualização de dados foram avaliadas quanto à sua compatibilidade com o CEIFA, dando ênfase a softwares gratuitos que proporcionassem as funcionalidades necessárias. Metabase e Grafana se destacaram no atendimento aos requisitos de visualização, ambos permitindo a criação de gráficos e consultas complexas, com compatibilidade para diversas fontes de dados. O Metabase ofereceu uma visualização avançada e personalizável, enquanto o Grafana se mostrou eficiente e leve, adaptando-se bem aos dispositivos de borda e mantendo

um bom desempenho com baixo consumo de recursos.

Os testes realizados em ambos os ambientes forneceram dados detalhados sobre o consumo de CPU e memória RAM, revelando o custo computacional associado a cada plataforma. O Grafana, por ser mais leve, teve um desempenho superior em dispositivos de borda, demonstrando baixo consumo de processamento e memória. Já o Metabase, embora mais exigente, provou-se funcional em um ambiente de nuvem, oferecendo funcionalidades completas e compatíveis com contextos de maior capacidade. Esses resultados permitiram identificar o custo computacional de cada abordagem e adequar a escolha de plataforma ao ambiente apresentado.

A estabilidade das plataformas foi testada com volumes diferentes de dados e acessos simultâneos, simulando cenários reais que variavam entre pequenas e médias propriedades agrícolas. Durante os testes, o Grafana e o Metabase mantiveram desempenho estável, com variações no tempo de resposta e consumo de recursos de acordo com o número de sensores e a capacidade de acesso simultâneo configurada. O Grafana, em especial, demonstrou ser altamente estável na borda, enquanto o Metabase se mostrou eficaz em ambientes de maior robustez na nuvem. Ambos suportaram as demandas, validando sua estabilidade e adequação às necessidades do CEIFA.

Este estudo avaliou a viabilidade de diferentes plataformas SaaS para a visualização de dados no CEIFA, evidenciando que o Grafana e o Metabase são as soluções mais adequadas. O Grafana, com seu baixo consumo de CPU e memória, demonstrou-se ideal para dispositivos de borda como o Raspberry Pi, além de permitir uma instalação ágil e a replicação de configurações via arquivos YAML. Em contrapartida, o Metabase, apesar de exigir mais recursos computacionais, oferece uma interface mais completa e suporte para visualizações detalhadas, sendo mais apropriado para ambientes de nuvem com maior capacidade. A escolha entre essas plataformas, portanto, dependerá do equilíbrio entre a necessidade de desempenho e a disponibilidade de recursos, com o Grafana se destacando em ambientes de hardware limitado e o Metabase, em cenários que exigem funcionalidades mais robustas e ricas em detalhes.

## REFERÊNCIAS

- BOULOUKAKIS, M. et al. Virtual reality for smart city visualization and monitoring. In: **Progress in IS**. [s.n.], 2018. Accessed: 2024-11-07. Disponível em: <<https://consensus.app/papers/reality-smart-city-visualization-monitoring-bouloukakis/ba9e74d85e805f16811e9f61c325d3c4/>>.
- BUSHEL FARM. **Bushel Farm: Farm Management Software**. 2024. Acessado em: 21 de maio de 2024. Disponível em: <<https://www.bushelfarm.com/>>.
- ELASTICSEARCH B.V. **Kibana: Your Window into the Elastic Stack**. 2024. Acessado em: 01 de maio de 2024. Disponível em: <<https://www.elastic.co/kibana>>.
- GRAFANA LABS. **Grafana: The open platform for beautiful analytics and monitoring**. 2024. Acessado em: 01 de maio de 2024. Disponível em: <<https://grafana.com>>.
- GROENEVELD, René. **Agworld: Crossing the hurdle of combining data**. 2021. Acessado em: 21 de maio de 2024. Disponível em: <<https://www.futurefarming.com/Management/Articles/2021/12/Agworld-Crossing-the-hurdle-of-combining-data-8213/>>.
- HILL, Lindsay. **Using Telegraf, InfluxDB and Grafana to Monitor Network Statistics**. 2022. Disponível em: <<https://lkhill.com/grafana-telegraf-influxdb-monitor-network/>>.
- ITUAH, Charles. **Managing Application Logs and Metrics With Elasticsearch and Kibana**. 2023. Acessado em: 22 de maio de 2024. Disponível em: <<https://dzone.com/articles/managing-application-logs-and-metrics-with-elasticsearch>>.
- JONES, Emily. Implementing data analytics in marketing strategies. **Journal of Marketing Analytics**, v. 7, n. 3, p. 210–223, 2019.
- METABASE. **Metabase: The Simplest, Fastest Way to Share Data and Analytics Inside Your Company**. 2024. Acessado em: 01 de maio de 2024. Disponível em: <<https://www.metabase.com>>.
- MOSO, J. C. et al. Anomaly detection on data streams for smart agriculture. **Agriculture**, 2021. Disponível em: <<https://consensus.app/papers/anomaly-detection-data-streams-smart-agriculture-moso/4d71808b7fc5ae2b38250fc1333fce7/>>.
- PI MY LIFE UP. **Installing Zabbix on to a Raspberry Pi**. 2024. Acessado em: 21 de maio de 2024. Disponível em: <<https://pimylifeup.com/raspberry-pi-zabbix/>>.
- PIKKARAINEN, Janne. **Raspberry Pi 4: Goodbye or Good Buy for Running Zabbix?** 2023. Acessado em: 21 de maio de 2024. Disponível em: <<https://blog.zabbix.com/raspberry-pi-4-goodbye-or-good-buy-for-running-zabbix>>.
- Restack Team. **SigNoz tool overview**. 2023. Acessado em: 22 de maio de 2024. Disponível em: <<https://restack.io>>.
- SCHILLER, Eryk et al. Landscape of iot security. **Computer Science Review**, Elsevier, v. 44, p. 100467, 2022.

SCHLEGEL, Georg Augusto; POLETTO, Alex Sandro Romeo de Souza. Smart agriculture: Estudo exploratório sobre a agricultura orientada pela tecnologia da informação e comunicação. **Revista INTELECTO• Fema• Assis• ISSN**, v. 2596, p. 0806, 2019.

**SIGNOZ. Signoz: Open Source Platform for Data Analysis and Reporting.** 2024. Acessado em: 01 de maio de 2024. Disponível em: <<https://signoz.io>>.

SMITH, John; DOE, Jane. Exploring the use of open source bi tools in small businesses. **Business Horizons**, Elsevier, v. 63, n. 2, p. 123–132, 2020.

SÁIZ-RUBIO, V.; ROVIRA-MÁS, F. From smart farming towards agriculture 5.0: A review on crop data management. **Agronomy**, 2020. Disponível em: <<https://consensus.app/papers/from-smart-farming-towards-agriculture-review-crop-data-sAqizrubio/a197993760a2515f9eca47f32fd49c15/>>.

TEAM, Logit.io. **What is Kibana?** 2023. Acessado em: 22 de maio de 2024. Disponível em: <<https://logit.io>>.

TIWARI, Nitish. **Application observability with Apache Kafka and SigNoz.** 2021. Acessado em: 22 de maio de 2024. Disponível em: <<https://opensource.com/article/21/4/observability-apache-kafka-signoz>>.

WOLFERT, Sjaak et al. Big data in smart farming—a review. **Agricultural systems**, Elsevier, v. 153, p. 69–80, 2017.

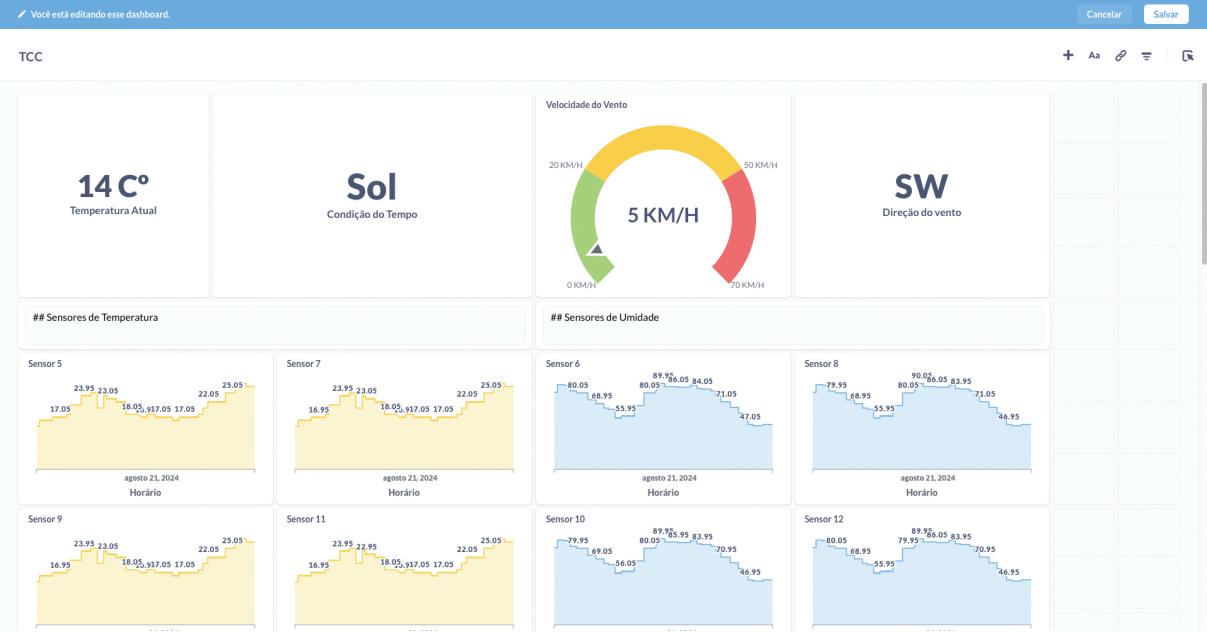
**ZABBIX. Zabbix Official Website.** 2024. Acessado em: 01 de maio de 2024. Disponível em: <<https://www.zabbix.com>>.

ZANELLA, Angelita Rettore de Araujo. **Detector Híbrido de Anomalias para Agricultura Inteligente.** Tese (Doutorado) — Universidade Federal do Paraná, 2022. Acessado em 19 de jun. de 2023. Disponível em: <<https://acervodigital.ufpr.br/handle/1884/80740>>.

ZANELLA, Angelita Rettore de Araujo; SILVA, Eduardo da; ALBINI, Luiz Carlos Pessoa. Security challenges to smart agriculture: Current state, key issues, and future directions. **Array**, v. 8, p. 100048, 2020. ISSN 2590-0056. Disponível em: <<https://www.sciencedirect.com/science/article/pii/S2590005620300333>>.

## APÊNDICE A – EDIÇÃO DE DASHBAORD NO METABASE

Figura 13 – Tela de edição da dashboard no Metabase



Fonte: O Autor, 2024

A tela de edição da *dashboard* no Metabase oferece uma interface fácil para organizar e personalizar a visualização de dados em formato de painéis. Utilizando um sistema de blocos, a plataforma permite que cada gráfico, tabela ou componente seja disposto de acordo com a preferência do usuário. Esse sistema de blocos facilita a organização dos elementos, pois possibilita ajustar a posição e o tamanho de cada item na tela. Entre os recursos oferecidos, a plataforma permite selecionar o tipo de visualização. Essa personalização ajudou criar uma visão geral para necessidades do monitoramento no dos dados.