

WAGNER JUNIO OLIVEIRA NEVES

**IMPLEMENTAÇÃO DE UMA ARQUITETURA
IIOT PARA MONITORAMENTO EM TEMPO
REAL DE PROCESSO DE COCÇÃO DE BACON**

Uberlândia

2025

WAGNER JUNIO OLIVEIRA NEVES

IMPLEMENTAÇÃO DE UMA ARQUITETURA IIOT PARA MONITORAMENTO EM TEMPO REAL DE PROCESSO DE COCÇÃO DE BACON

Trabalho de Conclusão de Curso da Engenharia de Controle e Automação da Universidade Federal de Uberlândia - UFU - Campus Santa Mônica, como requisito para a obtenção do título de Graduação em Engenharia de Controle e Automação.

Universidade Federal de Uberlândia - UFU
Faculdade de Engenharia Elétrica

Orientador Prof. Dr. Fábio Vincenzi Romualdo da Silva

Uberlândia

2025

Agradecimentos

Primeiramente, expresso minha profunda gratidão a Deus, que iluminou meu caminho e me guiou em cada passo desta jornada, pois Nele encontro força e confiança para superar todos os desafios.

Aos meus pais, meu eterno agradecimento pelo apoio incondicional, tanto financeiro quanto emocional, ao longo dos anos de graduação. Mesmo diante de altos e baixos, eles foram meu alicerce, incentivando-me a persistir apesar das dificuldades.

À minha namorada, Monique Carvalho, meu carinho e gratidão por estar ao meu lado nos momentos de alegria e adversidade. Seu incentivo constante e compreensão, mesmo nos períodos em que estive mais ausente, foram fundamentais para minha perseverança.

Aos meus professores, que contribuíram significativamente para meu desenvolvimento acadêmico e pessoal, destaco com carinho Renato Fernandes, Fábio Vicenzi e Renato Carrijo. Suas orientações pós-aula, paciência e disponibilidade para esclarecer dúvidas sobre empresas e projetos foram essenciais para meu crescimento.

À Universidade Federal de Uberlândia (UFU), minha gratidão pelo compromisso com a excelência no ensino, proporcionando um ambiente de aprendizado que está entre os melhores do país.

Por fim, ao engenheiro Houston Fornazier e ao técnico João Junio e ao supervisor Paulo Freitas, agradeço pela confiança e pela oportunidade de desenvolver este projeto. Foi uma honra poder demonstrar meu comprometimento e a qualidade do meu trabalho junto à equipe.

“Consagre ao Senhor tudo o que você faz, e os seus planos serão bem-sucedidos.”
(Provérbios 16:3)

Resumo

Este trabalho teve como objetivo desenvolver um sistema de supervisão e monitoramento industrial aplicado ao processo de cocção de bacon, utilizando tecnologias de aquisição de dados e visualização em tempo real. O sistema implementou a comunicação entre um Controlador Lógico Programável (CLP) e plataformas de supervisão, por meio da integração entre o KEPServerEX, o protocolo MQTT e o ambiente Node-RED. As variáveis de processo foram coletadas, armazenadas em banco de dados e apresentadas em dashboards no Grafana. Como resultado, o sistema permitiu o acompanhamento contínuo de parâmetros críticos, como temperatura, tempo de cozimento e etapas da receita, proporcionando maior confiabilidade no controle do processo. A solução desenvolvida foi validada em operação real, eliminando a necessidade de investimentos em um retrofit, atendendo às exigências do SIF e permitindo a replicação da arquitetura em outros processos industriais, demonstrando sua viabilidade, escalabilidade e impacto econômico positivo.

Palavras-chaves: Automação industrial, IIoT, sistemas de monitoramento.

Abstract

This work aimed to develop an industrial supervision and monitoring system applied to the bacon cooking process, using data acquisition and real-time visualization technologies. The system implemented communication between a Programmable Logic Controller (PLC) and supervision platforms through the integration of KEPServerEX, the MQTT protocol, and the Node-RED environment. The process variables were collected, stored in a database, and displayed in Grafana dashboards. As a result, the system enabled continuous monitoring of critical parameters such as temperature, cooking time, and recipe steps, providing greater reliability in process control. The developed solution was validated in real operation, eliminating the need for investments in a retrofit, meeting the requirements of the SIF, and allowing the replication of the architecture in other industrial processes, demonstrating its feasibility, scalability, and positive economic impact.

Key-words: Industrial automation, IIoT, monitoring systems.

Lista de ilustrações

Figura 1 – Esquema de funcionamento	22
Figura 2 – Configurações Ruffus.	25
Figura 3 – Tela principal do Proxmox.	26
Figura 4 – Configuração Device KEPServerEX.	28
Figura 5 – Configuração KEPServerEX CLP.	29
Figura 6 – Configuração de uma Tag no KepServerEx.	29
Figura 7 – Quick Client.	30
Figura 8 – Configuração KEPServerEX MQTT.	31
Figura 9 – Imagens ISOS.	32
Figura 10 – Containers.	33
Figura 11 – Nó MQTT.	34
Figura 12 – Fluxo NodeRed	36
Figura 13 – Tabela PgAdmin.	37
Figura 14 – Seleção da Estufa Grafana.	39
Figura 15 – IHM Processo Bacon.	41
Figura 16 – Dashborard Estufas Bacon.	42
Figura 17 – Dashboard Estufas Cozidas	43

Lista de abreviaturas e siglas

BBDD	Banco de Dados (Bases de Datos, termo usado em espanhol)
DNS	Domain Name System
HD	Disco Rígido (Hard Disk)
IIoT	Industrial Internet of Things / Internet das Coisas Industrial
IP	Internet Protocol
ISO	International Organization for Standardization (no contexto, imagem ISO)
JSON	JavaScript Object Notation
KVM	Kernel-based Virtual Machine
LXC	Linux Containers
MQTT	Message Queuing Telemetry Transport
OPC	OLE for Process Control / Open Connectivity
OPC DA	OPC Data Access
OPC UA	OPC Unified Architecture
PLC	Programmable Logic Controller / Controlador Lógico Programável
PTC	Power To Create
RTU	Remote Terminal Unit / Unidade Terminal Remota
SCADA	Supervisory Control and Data Acquisition
SIF	Serviço de Inspeção Federal
SO	Sistema Operacional
SQL	Structured Query Language
SSD	Solid State Drive
TCP/IP	Transmission Control Protocol / Internet Protocol
VM / VMs	Virtual Machine(s) / Máquina(s) Virtual(is)

Sumário

	Sumário	8
1	INTRODUÇÃO	10
1.1	Justificativa	10
1.2	Objetivos	11
1.2.1	Objetivos Específicos	12
2	REFERENCIAL TEÓRICO	13
2.1	Cocção industrial	13
2.2	SIF	13
2.3	IIOT	14
2.4	Sistemas de supervisão e monitoramento	15
2.4.1	KEPServerEX	15
2.4.2	Node-Red	16
2.4.3	MQTT	17
2.4.4	PostgreSQL	17
2.4.5	Grafana	18
2.5	Sistemas de virtualização	18
2.5.1	Proxmox	19
2.5.2	Docker	20
3	METODOLOGIA	22
3.1	Máquina do Servidor	23
3.2	Proxmox	23
3.2.1	Baixar ISOs e Criar mídia de instalação	23
3.3	VM windows - configurando KEPServerEX	27
3.4	VM Linux - configurando Docker	31
3.5	Node-Red	33
3.6	PostgreSQL	36
3.7	Grafana	37
4	RESULTADOS E DISCUSSÕES	40
4.1	Comparativo entre a situação anterior e a solução proposta	41
4.1.1	Supervisão e monitoramento	41
4.1.2	Gestão operacional e suporte à decisão	42
4.1.3	Conformidade regulatória e rastreabilidade (SIF)	42

4.1.4	Aplicação do Sistema ao Processo de Cocção de Linguças	42
5	CONCLUSÃO	44
	REFERÊNCIAS BIBLIOGRÁFICAS	45
6	ANEXO B	48
7	ANEXO C	93

1 Introdução

A Indústria 4.0 tem promovido transformações significativas nos processos industriais ao integrar tecnologias digitais, sistemas ciberfísicos e conectividade em tempo real entre máquinas e plataformas computacionais. Um dos pilares fundamentais dessa nova era industrial é a Internet das Coisas Industrial (IIoT), que permite a coleta, transmissão e análise de dados operacionais visando otimizar processos, aumentar a eficiência, reduzir falhas e permitir tomadas de decisões mais inteligentes e autônomas (AINSWORTH et al., 2021).

Dentro desse contexto, destaca-se a crescente aplicação de arquiteturas IIoT em ambientes produtivos diversos, incluindo o setor alimentício. A implementação de sensores inteligentes e protocolos de comunicação como MQTT e OPC UA tem possibilitado o monitoramento remoto de variáveis críticas, como temperatura e tempo, garantindo não apenas a qualidade do produto final, mas também a rastreabilidade do processo e a conformidade com normas sanitárias rigorosas (KUMARESAN et al., 2023). Além disso, plataformas como o Node-RED e o Grafana têm ganhado espaço por oferecerem interfaces intuitivas e escaláveis para o tratamento e a visualização de dados em tempo real.

Estudos recentes apontam que a integração entre PLCs e sistemas em nuvem, aliada ao uso de protocolos padronizados, representa uma solução viável e promissora para o controle e supervisão de processos industriais em tempo real (KUMARESAN et al., 2023). Por outro lado, a transição para ambientes industriais conectados também traz desafios relacionados à segurança, o que tem motivado pesquisas voltadas à aplicação de algoritmos de aprendizado de máquina para detecção de intrusões em redes IIoT (MATKE; KUMAR; SINGH, 2023).

Diante disso, o presente trabalho propõe a implementação de uma arquitetura IIoT aplicada ao monitoramento do processo de cocção de bacon. A solução integra controladores lógicos programáveis, servidores de comunicação (KEPServerEX), protocolos de transmissão (MQTT), ferramentas de visualização (Grafana) e tratamento de dados (Node-RED). O objetivo é desenvolver um sistema que possibilite o acompanhamento contínuo e remoto de variáveis do processo, promovendo maior controle operacional, rastreabilidade e suporte à manutenção preditiva.

1.1 Justificativa

A implementação deste projeto foi realizada em uma das maiores companhias de alimentos do mundo, reconhecida por seu portfólio multinacional com mais de 30 marcas

consolidadas no mercado. Por questões de sigilo industrial e estratégico, o nome da empresa não será mencionado neste trabalho.

A motivação para a realização deste trabalho está diretamente relacionada à necessidade de otimizar o processo de monitoramento e armazenamento das informações de temperatura nas estufas de cocção do bacon. O acompanhamento operacional, anteriormente realizado com planilhas físicas e anotações diárias, tornava-se vulnerável a falhas humanas, perdas de dados e falta de rastreabilidade, especialmente em situações de auditorias ou referente a qualidade do produto, nas quais é exigida a apresentação de registros históricos confiáveis.

Além disso, a ausência de um sistema centralizado obrigava os operadores e a equipe de qualidade a se deslocarem até o campo para obter informações em tempo real, que aumentava o tempo e comprometia a eficiência do processo.

Dessa forma, justificou-se o desenvolvimento de uma solução capaz de registrar automaticamente os dados de temperatura, disponibilizá-los em dashboards acessíveis de qualquer ponto da fábrica e assegurar maior confiabilidade, rastreabilidade e agilidade no acompanhamento das variáveis críticas do processo de cocção industrial.

- Antes do Projeto

O acompanhamento do processo de cozimento do bacon era feito de forma manual. Os operadores anotavam as informações em planilhas de papel todos os dias. Isso causava alguns problemas: as anotações podiam conter erros, os papéis podiam se perder e, quando era preciso verificar informações antigas, muitas vezes elas não estavam completas. Em auditorias ou quando havia problemas de qualidade, encontrar e comprovar os dados era difícil.

- Depois do Projeto

Agora, com a instalação do sistema, todo o processo é monitorado automaticamente e em tempo real. As informações ficam salvas em um banco de dados e o dashboard pode ser acessado de qualquer lugar da fábrica. Isso deixa o trabalho mais rápido, seguro e organizado. Além disso o mesmo fluxo pode ser aplicado em diferentes tipos de processo (e já vem sendo aplicado), buscando mostrar os dados em tempo real, para integrantes da qualidade, supervisores e gerentes que ajudam também em outros âmbitos como na manutenção preditiva.

1.2 Objetivos

Desenvolver e implementar um sistema de monitoramento e registro histórico em tempo real do processo de cocção industrial de bacon, utilizando tecnologias de virtualização,

aquisição de dados, comunicação por protocolo MQTT, tratamento lógico de variáveis e visualização gráfica, com foco em garantir rastreabilidade, confiabilidade operacional e acessibilidade das informações do processo produtivo.

O sistema proposto integra diferentes ferramentas tecnológicas como o KepserverEX para aquisição de dados dos CLPs, o Node-RED para tratamento e roteamento das informações, o PostgreSQL como banco de dados relacional para armazenamento histórico, o Grafana para visualização em tempo real, e o Proxmox VE como plataforma de virtualização centralizada visando atender às exigências operacionais e de auditoria em ambientes industriais, com ênfase em robustez, escalabilidade e autonomia local.

1.2.1 Objetivos Específicos

- Historiar os dados de temperatura do produto, possibilitando rastrear lotes com possíveis avarias e garantir a conformidade com normas de qualidade e segurança alimentar.
- Facilitar auditorias e análises de desempenho, mantendo registros históricos organizados e de fácil acesso.
- Reduzir tempo de parada e custos de manutenção por meio de insights extraídos do histórico de dados, favorecendo ações de manutenção preditiva.
- Visualização e análise por dashboards no Grafana.

2 Referencial Teórico

Neste tópico são apresentados os fundamentos teóricos e tecnológicos que embasam o desenvolvimento do sistema de monitoramento e historiamento aplicado ao processo de cocção industrial. Serão abordados conceitos relacionados à aquisição de dados industriais, protocolos de comunicação, tratamento e armazenamento de informações, bem como visualização em tempo real voltada para ambientes industriais.

2.1 Cocção industrial

A cocção industrial é o processo de aplicação controlada de calor a alimentos em ambiente fabril, visando assegurar qualidade, padronização e segurança alimentar. Esse processo pode englobar etapas de cozimento, pasteurização, desidratação parcial e defumação, dependendo do tipo de produto.

No caso do bacon, a cocção é realizada em estufas industriais que permitem a combinação de calor e fumaça, garantindo a elevação da temperatura interna até níveis seguros (aproximadamente 70 °C) e atendendo às normas de inspeção sanitária.

As principais variáveis de controle no processo de cocção industrial incluem a temperatura, a umidade relativa, o tempo de exposição, o fluxo de ar e, quando aplicável, a injeção de fumaça. Essas variáveis precisam ser monitoradas e registradas em tempo real para atender às exigências de rastreabilidade impostas pelo Serviço de Inspeção Federal (SIF), além de possibilitar auditorias e garantir conformidade regulatória.

Nesse contexto, a automação e o monitoramento contínuo tornam-se indispensáveis para reduzir falhas humanas, aumentar a confiabilidade dos registros e otimizar a gestão do processo produtivo.

2.2 SIF

O Serviço de Inspeção Federal, conhecido mundialmente pela sigla S.I.F. e vinculado ao Departamento de Inspeção de Produtos de Origem Animal – DIPOA, é o responsável por assegurar a qualidade de produtos de origem animal comestíveis e não comestíveis destinados ao mercado interno e externo, bem como de produtos importados. Atualmente, o SIF tem atuação em mais de 5 mil estabelecimentos brasileiros, todos sob a supervisão do DIPOA.([PECUÁRIA, 2025b](#))

De acordo com Decreto nº 9.013, de 29 de março de 2017 — RIISPOA. Regulamenta o sistema de inspeção industrial e sanitária de produtos de origem animal, definindo

diretrizes claras para fiscalização, inclusive em relação à rastreabilidade e autocontroles executados pelos estabelecimentos registrados no SIF.([PECUÁRIA, 2025a](#))

De acordo com essa legislação, é obrigatória a manutenção de rastreabilidade e registros confiáveis das variáveis de processo, especialmente em casos que envolvem riscos à saúde pública e exigências de segurança alimentar. A ausência de registros históricos, como era o caso com as anotações manuais em planilhas físicas torna-se um problema, em auditorias ou em investigações de qualidade.

A arquitetura IIoT desenvolvida neste projeto atende com qualidade esse requisito, gerando um banco de dados com histórico completo de variáveis como temperatura, tempo e estados operacionais, conforme previsto nas normas do SIF e sob a fiscalização do MAPA/DIPOA. Com isso foi possível não apenas garantir a conformidade com normas de qualidade e segurança alimentar, mas também criar uma base de dados histórica valiosa para identificar tendências, prever falhas e otimizar o desempenho do processo produtivo.

2.3 IIOT

A Internet Industrial das Coisas (IIoT) refere-se à aplicação da Internet das Coisas (IoT) em ambientes industriais, conectando dispositivos, máquinas, sensores e sistemas para coletar, analisar e compartilhar dados em tempo real. O objetivo é otimizar processos, aumentar a eficiência, reduzir custos e melhorar a tomada de decisões em setores como manufatura, energia, logística, agricultura e saúde. A IIoT utiliza tecnologias como sensores, redes de comunicação, computação em nuvem, inteligência artificial e análise de big data para criar sistemas inteligentes e automatizados.([IBERDROLA, 2025](#))

A IIoT é um pilar essencial da Indústria 4.0, a integração de tecnologias avançadas para otimizar processos industriais, destaca-se por suas funcionalidades de conectividade, automação, monitoramento e análise de dados. A conectividade avançada permite que dispositivos, máquinas e sensores sejam interconectados por meio de redes robustas, possibilitando a troca de dados em tempo real com alta confiabilidade e baixa latência, essencial para ambientes industriais complexos.

A automação inteligente, por sua vez, viabiliza o controle automatizado de processos, como linhas de produção e cadeias logísticas, utilizando sistemas ciberfísicos e robôs colaborativos para reduzir a intervenção humana e aumentar a eficiência operacional. O monitoramento em tempo real e a coleta dos dados de forma contínua é crucial para a manutenção preditiva e a tomada de decisões. Além disso, a análise avançada de dados, utilizando inteligência artificial e machine learning, processa grandes volumes de informações para identificar padrões, prever falhas, otimizar operações e reforçar a segurança.

2.4 Sistemas de supervisão e monitoramento

Os sistemas supervisórios também conhecidos como sistemas SCADA (Supervisory Control and Data Acquisition) são sistemas de supervisão e controle de dados os quais permitem que sejam monitoradas e rastreadas informações de um processo produtivo, de um equipamento ou até mesmo de um ambiente. Tais informações são coletadas através de equipamentos de aquisição de dados (sensores), e posteriormente manipuladas, analisadas, armazenadas e apresentadas de forma visual e intuitiva para o operador. Basicamente a estrutura física de um sistema supervisório é composto por 3 camadas. A primeira sendo composta por sensores, atuadores, motores, válvulas etc. A segunda camada contempla a parte de comunicação entre o CLP e o microcomputador que executará o sistema supervisório. E por último a terceira camada, onde se encontra o microcomputador que será responsável de executar o sistema supervisório, armazenar os dados e realizar a interface com o operador.(CBR, 2025)

Tradicionalmente, os sistemas supervisórios foram desenvolvidos no modelo monolítico, no qual todas as funções essenciais — aquisição de dados, banco de dados, lógica de tratamento de informações e visualização — estão integradas em um único software. Essa abordagem apresenta como vantagens a centralização dos recursos e a facilidade de manutenção em ambientes menores. Contudo, o alto custo de licenciamento, a dependência de fornecedores específicos e a limitação de escalabilidade tornam-se desvantagens em cenários industriais complexos e em constante expansão.

Com o avanço das tecnologias de conectividade e virtualização, surgiram os sistemas distribuídos ou modulares, nos quais as funções clássicas de um SCADA são separadas em componentes independentes, que se comunicam entre si. Essa arquitetura possibilita maior flexibilidade na escolha das ferramentas, permitindo integrar soluções especializadas para aquisição de dados, bancos de dados, middleware e supervisão gráfica. Como principais vantagens destacam-se a escalabilidade, o menor custo de implantação e a possibilidade de adaptar a solução às necessidades específicas do processo. A principal desvantagem está na maior complexidade de integração e configuração inicial, exigindo maior conhecimento técnico para garantir a interoperabilidade entre os módulos.

Assim, a escolha entre um sistema supervisório monolítico ou modular depende do porte da aplicação, do orçamento disponível e da necessidade de flexibilidade futura, sendo que arquiteturas modulares têm ganhado espaço em função da tendência de digitalização e descentralização promovida pela Indústria 4.0.

2.4.1 KEPServerEX

O KEPServerEX é uma plataforma de software de comunicação criada pela Kepware e gerida pela Logitek a nível de espanhol e português. É um dos softwares mais utilizados

atualmente por todo o tipo de utilizadores, empresas e setores, já que permite uma troca de dados bidirecional, segura, vigorosa e fiável entre qualquer tipo de dispositivo e sistema de automatização, controle e gestão. KEPServerEX é capaz de adquirir informação de praticamente qualquer dispositivo industrial (PLCs, RTUs, BBDD, etc.), centralizar todos esses dados e dispô-los via OPC DA e OPC UA, os padrões de mercado ([KEPSERVEREX, 2020](#)).

A KepServerEx é um servidor OPC unificado, ao qual, para além de uma infindável lista de drivers, também podem ser acrescentados outros componentes que permitem dar valor acrescentado às soluções de comunicação e de automatização. Uma das características mais apreciadas da KEPServerEx é a sua escalabilidade. Pode ser criado um projeto para comunicar inicialmente com um único driver e, à medida que vão crescendo as necessidades, novos drivers de comunicação podem ser adicionados. Tudo isto a partir de uma única interface comum ([KEPSERVEREX, 2020](#)).

Utilizamos o KepServerEX devido à sua excelência em confiabilidade, compatibilidade e facilidade de integração com diversos dispositivos industriais. Sua arquitetura modular, baseada em drivers industriais, permite a comunicação com uma ampla gama de equipamentos e protocolos proprietários ou abertos. Além da compatibilidade com os principais fabricantes de CLP's também oferece uma interface altamente intuitiva para as devidas configurações.

2.4.2 Node-Red

O objetivo do Node-RED é permitir que qualquer pessoa crie aplicativos que colem, transformem e visualizem seus dados; criando fluxos que possam automatizar seu mundo. Sua natureza de baixo código o torna acessível a usuários de qualquer área, seja para automação residencial, sistemas de controle industrial ou qualquer área entre esses dois extremos. De amadores a operações industriais de larga escala, o Node-RED encontrou um lugar em uma ampla gama de aplicações, apoiado por uma comunidade ativa que construiu mais de 5.000 nós e fluxos para ajudar você a começar rapidamente. ([NODE-RED, 2013](#)).

O Node-RED é construído em Node.js, aproveitando ao máximo seu modelo não bloqueante e orientado a eventos. Isso o torna ideal para execução na borda da rede em hardware de baixo custo, como o Raspberry Pi, e também na nuvem. Os fluxos criados no Node-RED são armazenados usando JSON, que pode ser facilmente importado e exportado para compartilhamento com outras pessoas. Uma biblioteca de fluxos on-line permite que você compartilhe seus melhores fluxos com o mundo ([NODE-RED, 2013](#)).

Node-RED além de ser um software open-source, é baseado em fluxos visuais, que permite a construção de lógicas e automações complexas, sem a necessidade de escrever

grandes códigos ou criar softwares dedicados. A grande variedade de nós facilita a vida dos usuários, onde pode ser aplicado em múltiplos âmbitos e sua interface gráfica em ambiente web permite criar, editar e implantar fluxos, simplificando a manutenção ou alteração do sistema. Além disso como os dados são armazenados usando JSON, temos uma fácil integração com o banco de dados.

2.4.3 MQTT

O MQTT (Message Queuing Telemetry Transport) é um protocolo de mensagens leve e de código aberto, projetado para comunicação eficiente em aplicações de IoT (Internet das Coisas). Ele utiliza um modelo de publicação/assinatura, garantindo comunicação de baixa largura de banda e alta confiabilidade entre dispositivos. O MQTT é amplamente utilizado para troca de dados em tempo real, permitindo conectividade escalável e segura em casas inteligentes, automação industrial e outros sistemas conectados. Sua simplicidade e suporte a diversas camadas de transporte o tornam ideal para dispositivos com recursos limitados em ambientes de rede remotos ou não confiáveis. ([Cavli Wireless, 2025](#)).

2.4.4 PostgreSQL

O PostgreSQL é um poderoso sistema de banco de dados objeto-relacional de código aberto que utiliza e estende a linguagem SQL, combinada com diversos recursos que armazenam e escalam com segurança as cargas de trabalho de dados mais complexas. As origens do PostgreSQL remontam a 1986, como parte do projeto POSTGRES na Universidade da Califórnia em Berkeley, e conta com mais de 35 anos de desenvolvimento ativo na plataforma principal ([POSTGRESQL, 2025](#)).

O PostgreSQL oferece diversos recursos para ajudar desenvolvedores a criar aplicativos, administradores a proteger a integridade dos dados e criar ambientes tolerantes a falhas, além de ajudar você a gerenciar seus dados, independentemente do tamanho do conjunto de dados. Além de ser gratuito e de código aberto, o PostgreSQL é altamente extensível. Por exemplo, você pode definir seus próprios tipos de dados, criar funções personalizadas e até mesmo escrever código de diferentes linguagens de programação sem precisar recompilar seu banco de dados ([POSTGRESQL, 2025](#)).

O PostgreSQL busca se adequar ao padrão SQL, desde que tal conformidade não contradiga os recursos tradicionais ou possa levar a decisões arquitetônicas inadequadas. Muitos dos recursos exigidos pelo padrão SQL são suportados, embora às vezes com sintaxe ou função ligeiramente diferentes. Novas mudanças em direção à conformidade podem ser esperadas ao longo do tempo. Desde o lançamento da versão 16, em setembro de 2023, o PostgreSQL está em conformidade com pelo menos 170 dos 177 recursos obrigatórios para a conformidade com o SQL:2023 Core. Até o momento em que este documento foi

escrito, nenhum banco de dados relacional atende à conformidade total com este padrão. (POSTGRESQL, 2025).

A escolha do PostgreSQL como sistema de gerenciamento de banco de dados foi motivada por sua robustez, desempenho e ampla compatibilidade com ferramentas modernas de integração e visualização de dados. Sendo uma plataforma open-source amplamente consolidada no mercado, o PostgreSQL oferece recursos avançados como suporte a tipos de dados complexos, controle transacional, consultas SQL otimizadas e gerenciamento eficiente de grandes volumes de dados.

2.4.5 Grafana

Grafana Cloud é uma plataforma de observabilidade hospedada na nuvem totalmente gerenciada, ideal para ambientes nativos da nuvem. A Grafana Cloud é um arsenal integrado para métricas, logs e rastros unificados na melhor plataforma de painéis para visualização de dados, também está integrada com o Grafana k6 para testes de desempenho e Grafana Incident e Grafana OnCall para gerenciar seu fluxo de trabalho de resposta a incidentes, tornando-se o arsenal de observabilidade mais composto e abrangente do mundo (LABS, 2025).

O Grafana foi criado com base no princípio de que os dados devem ser acessíveis a todos na organização, não apenas ao profissional de operações. Ao democratizar os dados, a Grafana ajuda a facilitar uma cultura em que os dados podem ser facilmente usados e acessados pelas pessoas que precisam deles, ajudando a quebrar silos de dados e capacitar equipes. Os painéis do Grafana não apenas dão significado profundo aos dados coletados de diversas fontes, mas você também pode compartilhar os painéis criados com outros membros da equipe, permitindo que vocês explorem os dados juntos. (LABS, 2025).

No contexto deste trabalho, o Grafana foi empregado para apresentar, de maneira clara e organizada, as informações coletadas ao longo do processo de monitoramento das variáveis industriais. Os dados armazenados no banco PostgreSQL, oriundos do Node-RED e do servidor OPC (KepServerEX), foram integrados ao Grafana por meio de conexões diretas, permitindo a criação de painéis dinâmicos com indicadores de desempenho, gráficos históricos e alarmes configuráveis.

2.5 Sistemas de virtualização

A virtualização é um processo que permite a utilização mais eficiente do hardware físico do computador e é a base da cloud computing. A virtualização usa softwares para criar uma camada de abstração sobre o hardware do computador que permite que os elementos de hardware de um único computador, como processadores, memória, armazenamento e muito mais, sejam divididos em vários computadores virtuais, comumente chamados de

máquinas virtuais (VMs). Cada VM executa seu próprio sistema operacional (SO) e se comporta como um computador independente, apesar de estar em execução em apenas uma parte do efetivo hardware de computador subjacente, possibilitando uma utilização mais eficiente do hardware físico do computador e permite um maior retorno sobre o investimento da empresa em hardware.([IBM, 2025b](#))

Já o hipervisor é um programa que acrescenta recursos de computação, como processamento, memória e armazenamento, e os redireciona entre máquinas virtuais (VMs) Essa tecnologia possibilita a virtualização, isso significa que você pode criar e executar muitas VMs de uma única máquina física. Um hipervisor é chamado também de monitor de máquina virtual (VMM - Virtual Machine Monitor). Pense nele como um supervisor responsável por dispersar os componentes das VMs. Ele retém esses recursos do hardware físico e os oferece às diversas VMs de uma vez, permitindo a criação de novas e o gerenciamento de VMs existentes. O hardware físico é chamado de host quando utilizado como um hipervisor, e as VMs que usam os recursos do host são chamadas de guests.([HAT, 2025](#))

Além das máquinas virtuais tradicionais, a evolução da virtualização deu origem aos contêineres, que representam uma forma mais leve e eficiente de isolamento. Enquanto uma máquina virtual exige a instalação de um sistema operacional completo para cada instância, os contêineres compartilham o mesmo kernel do sistema hospedeiro, executando apenas as bibliotecas e dependências necessárias para cada aplicação.

As principais diferenças entre máquinas virtuais e contêineres podem ser resumidas da seguinte forma:

Máquinas virtuais: oferecem maior isolamento e segurança, pois cada instância possui seu próprio sistema operacional; no entanto, consomem mais recursos e demandam maior tempo de inicialização.

Contêineres: apresentam inicialização rápida, menor consumo de recursos e maior portabilidade entre ambientes, sendo ideais para aplicações distribuídas e escaláveis; contudo, possuem menor isolamento em comparação às máquinas virtuais.

Dessa forma, máquinas virtuais e contêineres não se excluem, mas se complementam: enquanto as primeiras são adequadas para cenários que demandam forte isolamento e independência de sistemas, os contêineres são amplamente utilizados para aplicações modulares, distribuídas e em arquiteturas de microsserviços.

2.5.1 Proxmox

A virtualização usa software para criar uma camada de abstração sobre o hardware do computador que permite que os elementos de hardware de um único computador, como processadores, memória, armazenamento e muito mais, sejam divididos em vários

computadores virtuais, comumente chamados de máquinas virtuais (VMs). Cada VM executa seu próprio sistema operacional (SO) e se comporta como um computador independente, apesar de estar em execução em apenas uma parte do efetivo hardware de computador subjacente (IBM, 2025a).

O Proxmox Virtual Environment é uma poderosa plataforma de virtualização de servidores de código aberto para gerenciar duas tecnologias de virtualização — KVM (Kernel-based Virtual Machine) para máquinas virtuais e LXC para contêineres — com uma única interface web. Ele também integra ferramentas prontas para uso para configurar alta disponibilidade entre servidores, armazenamento definido por software, rede e recuperação de desastres. O Ambiente Virtual Proxmox é baseado no Debian GNU/Linux e utiliza um kernel Linux personalizado. O código-fonte do Proxmox VE é gratuito, distribuído sob a Licença Pública Geral GNU Affero, v3 (GNU AGPLv3). Isso significa que você tem liberdade para usar o software, inspecionar o código-fonte a qualquer momento e contribuir com o projeto.(SMITH, 2023).

Em ambientes industriais, o uso de um hipervisor local como o Proxmox VE apresenta diversas vantagens. Primeiramente, permite o isolamento de serviços críticos em máquinas virtuais independentes, minimizando riscos de falha em cascata e facilitando a manutenção. Além disso, a centralização de todos os serviços em um único hardware reduz a complexidade de infraestrutura e os custos com equipamentos físicos adicionais. A virtualização também permite snapshots, backups simplificados e recuperação rápida em caso de falhas, o que é fundamental em processos industriais que exigem alta disponibilidade(SMITH, 2023).

O principal objetivo do Proxmox é garantir a concentração das máquinas utilizadas em um único local de forma virtualizada, segura e funcional com uma interface amigável e prática. Com o uso desse recurso conseguimos criar máquinas virtuais (VMs) de forma gratuita, dando liberdade de escolher o sistema operacional (SO) que o usuário preferir. Nesse projeto, a adoção do Proxmox VE como camada de virtualização foi essencial para garantir escalabilidade, flexibilidade e confiabilidade ao sistema de monitoramento industrial. A possibilidade de gerenciar múltiplos serviços (como o Kepserver, o Node-RED, o banco de dados e o Grafana), através de uma única máquina virtual (VMs), por meio de contêineres Docker e imagens personalizadas (ISOs), permitindo o provisionamento rápido e o gerenciamento eficiente dos componentes do sistema.

2.5.2 Docker

O Docker é uma plataforma aberta para desenvolvimento, distribuição e execução de aplicativos. O Docker permite separar seus aplicativos da infraestrutura para que você possa entregar software rapidamente. Com o Docker, você pode gerenciar sua infraestrutura da mesma forma que gerencia seus aplicativos. Ao aproveitar as metodologias do Docker

para distribuição, teste e implantação de código, você pode reduzir significativamente o atraso entre a escrita do código e sua execução em produção. O Docker oferece a capacidade de empacotar e executar um aplicativo em um ambiente fracamente isolado chamado contêiner. O isolamento e a segurança permitem executar vários contêineres simultaneamente em um determinado host. Os contêineres são leves e contêm tudo o que é necessário para executar o aplicativo, então você não precisa depender do que está instalado no host. Você pode compartilhar contêineres enquanto trabalha e garantir que todos com quem você compartilha recebam o mesmo contêiner que funciona da mesma maneira (DOCKER, 2025b).

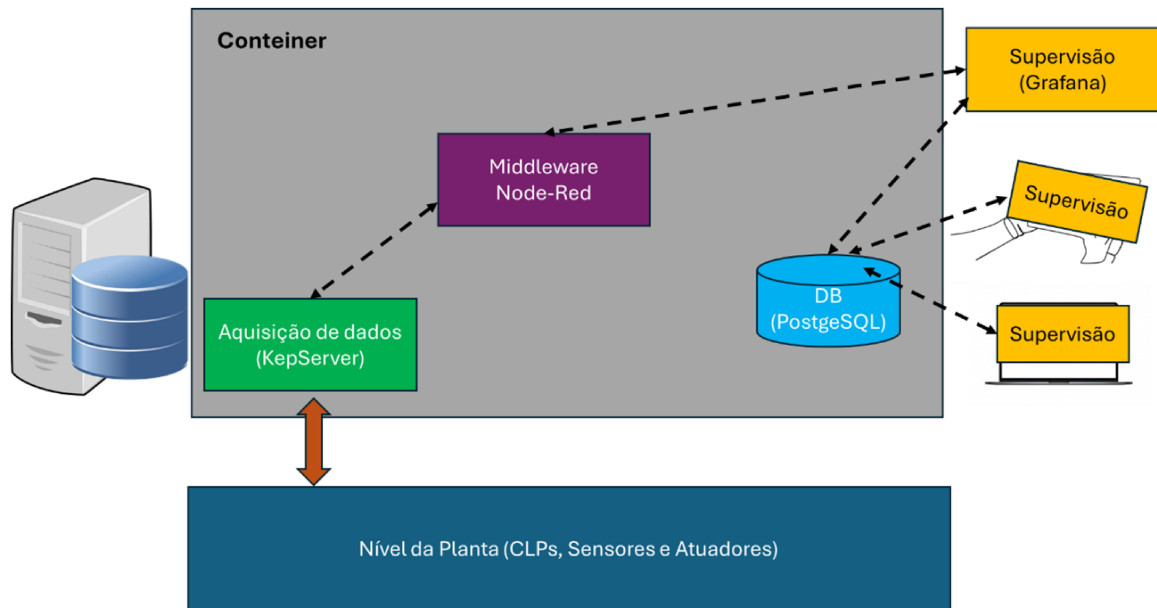
Ele oferece uma alternativa viável e econômica às máquinas virtuais baseadas em hipervisor, permitindo que você use mais da capacidade do seu servidor para atingir seus objetivos de negócios. O Docker é perfeito para ambientes de alta densidade e para implantações de pequeno e médio porte, onde você precisa fazer mais com menos recursos (DOCKER, 2025b).

Nessa aplicação utilizamos o Portainer.io um gerenciado de containers, que nos permite realizar esse gerenciamento de forma, simples pratica e visual, com isso colocamos todas as imagens necessarias dentro dele. (Node-Red, Grafana, MQTT, PostgreSQL).

3 Metodologia

Este projeto teve como objetivo desenvolver um sistema de monitoramento e registro histórico em tempo real do processo de cocção industrial, utilizando tecnologias de virtualização, aquisição de dados, tratamento lógico de variáveis e visualização gráfica. A Figura 1 apresenta o diagrama de blocos da arquitetura proposta.

Figura 1 – Esquema de funcionamento



Fonte: Autoria própria (2025).

Para a implementação, foi utilizada uma infraestrutura baseada em virtualização via Proxmox VE, permitindo a integração de diferentes ferramentas voltadas à automação industrial. O KEPServerEX foi configurado como servidor OPC, responsável pela comunicação com os CLPs e pela aquisição dos dados de processo. O Node-RED atuou como middleware, processando os dados e encaminhando-os ao banco de dados PostgreSQL, que assegurou o armazenamento estruturado das informações. Por fim, o Grafana foi utilizado para a visualização dos dados em tempo real, por meio de dashboards personalizados. Cada componente foi testado individualmente e de forma integrada, garantindo o funcionamento correto do sistema.

Nos subtópicos a seguir, será detalhado o processo metodológico adotado, utilizando a estufa01 como exemplo, sendo que os mesmos procedimentos foram aplicados às demais unidades de cocção.

3.1 Máquina do Servidor

Como o projeto foi desenvolvido em uma industria alimenticia real e multinacional, o computador que foi utilizado como servidor foi disponibilizado pela mesma. O computador é um OptiPlex 7020, as configurações são:

- 14^a geração Intel® Core™ i5-14500 vPro® (14 core, 20 threads, cache de 24MB, 2.6 GHz a 5.0 GHz)
- 1TB SSD
- 16Gb Memória Ram
- Placa gráfica: intel 770

Após termos o servidor em mãos é necessario realizar a formatação, para adicionarmos o Proxmox. Primeiro passo pra formatar o seu disco e remover todas suas partições.

- Abra o CMD como administrador.
- Digite *diskpart* e localize o disco.
- Em seguida selecione usando o comando *select disk x* (onde x é o numero do disco que deseja selecionar).
- Logo após digite *clean* para poder apagar todos dados e partições.

Agora com o SSD formatado podemos dar prosseguimento em nosso projeto.

3.2 Proxmox

Pra iniciarmos o nosso projeto de vez, precisamos instalar o sistema operacional, ou melhor dizendo plataforma de virtualização. Segundo (SMITH, 2023) O Proxmox VE é, de certa forma, um sistema operacional, mas talvez seja mais adequado tratá-lo como uma plataforma de virtualização, construída a partir uma distribuição baseada em Debian Linux.

3.2.1 Baixar ISOs e Criar mídia de instalação

O processo se inicia realizando o download da ISO do Proxmox VE no site Oficial SOLUTIONS (2025). Em seguida o software Rufus foi instalado para a criação de drives USB inicializáveis. O Rufus é uma ferramenta que facilita a formatação e a geração de

unidades USB inicializáveis, como pen drives, cartões de memória e outros dispositivos similares.[Rufus \(2025\)](#)

Rufus pode ser especialmente útil em casos onde:

- você precisa criar uma mídia de instalação USB a partir de ISOs inicializáveis (Windows, Linux, UEFI, etc.)
- você precisa trabalhar em um equipamento que não possui um SO (sistema operacional) instalado
- você precisa atualizar um BIOS ou outro firmware, a partir do DOS
- você deseja executar um utilitário de baixo nível
- Apesar de seu pequeno tamanho, o Rufus fornece tudo que você precisa!

Para criar um arquivo bootável do Proxmox, é necessário um pendrive com capacidade de 16GB. O dispositivo deve ser selecionado, seguido pela escolha da imagem do Proxmox. O esquema de partição GPT deve ser configurado, com UEFI como sistema de destino e FAT32 como sistema de arquivos. Após a configuração, o processo é finalizado.

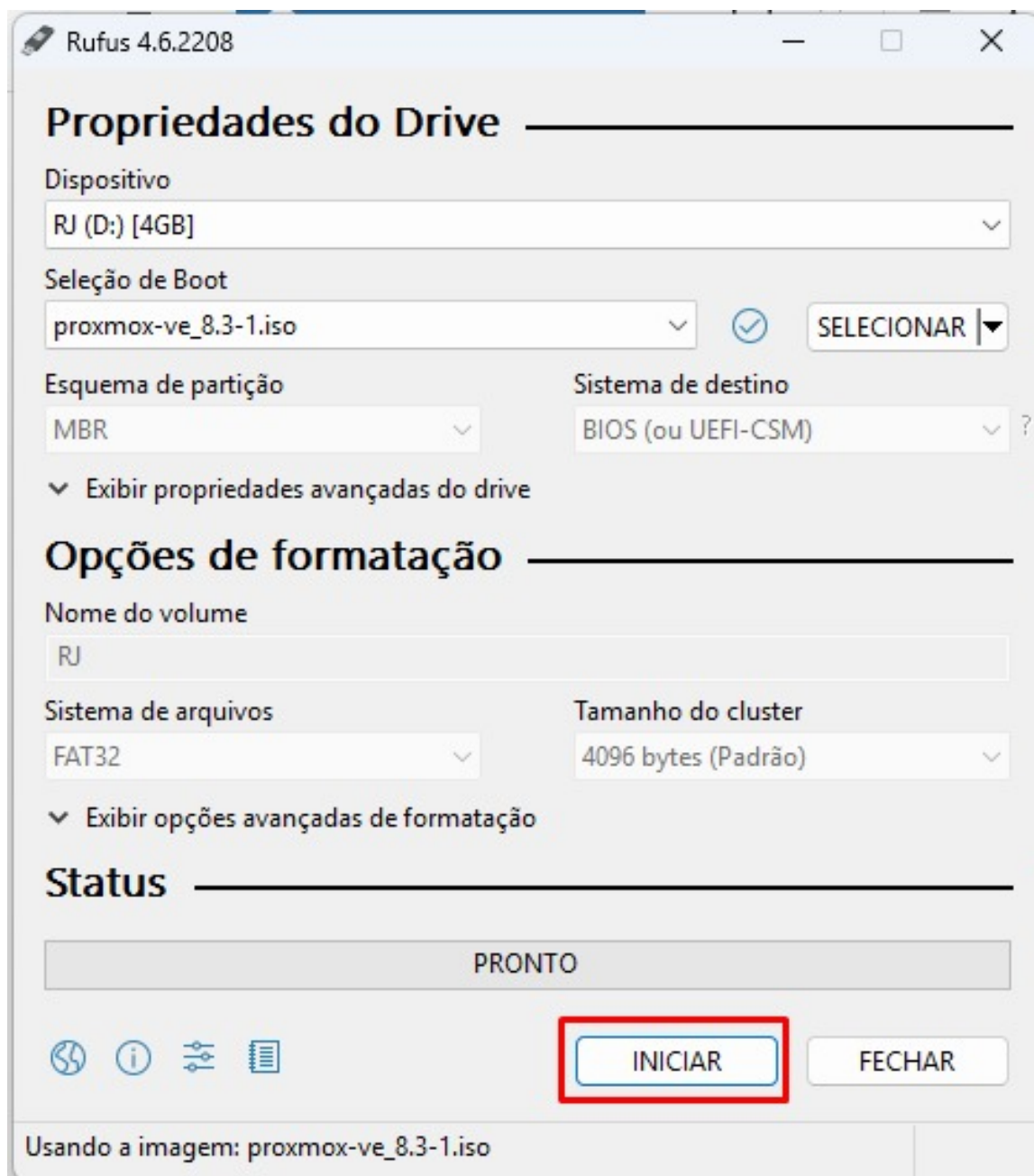
Explicando pontos escolhidos:

O GPT (GUID Partition Table) é o padrão moderno de particionamento, substituto do MBR. Ele suporta discos maiores que 2 TB, até 128 partições primárias e tem redundância das tabelas de partição (mais confiável). O GPT só funciona para boot em sistemas que usam UEFI. Se o hardware fosse mais antigo, com BIOS/Legacy, seria preciso escolher MBR.[\(Wikipedia, 2025\)](#).

O UEFI moderniza o processo de boot em relação ao BIOS e introduz melhores recursos como Secure Boot, interface gráfica e suporte a drives grandes. Ele foi concebido para trabalhar em conjunto com GPT.[\(Super User, 2025\)](#). Embora FAT32 seja amplamente suportado e recomendado para compatibilidade máxima com UEFI, o padrão UEFI não exige estritamente FAT32; com utilitários como o Rufus, é possível usar NTFS com UEFI:NTFS para contornar o limite de arquivos maiores que 4 GB. No entanto, por segurança, muitos guias ainda aderem ao uso do FAT32 para garantir que o pendrive será reconhecido por firmwares UEFI mais restritivos[\(RAKHESH, 2025\)](#).

Após isso é só seguir como mostra na figura 2:

Figura 2 – Configurações Ruffus.



Fonte: Autoria própria (2025).

O processo de instalação do Proxmox inicia-se com o boot do computador, acessando diretamente a tela de aceitação das licenças. Em seguida, a unidade de disco destinada à instalação é selecionada, juntamente com o idioma, fuso horário e localidade. Posteriormente, uma senha e um endereço de e-mail são configurados para acesso ao sistema; essas informações são restritas, considerando que o projeto é executado em uma empresa privada. Por fim, os parâmetros de rede, como endereço IP, gateway e DNS, são definidos conforme as especificações do ambiente.

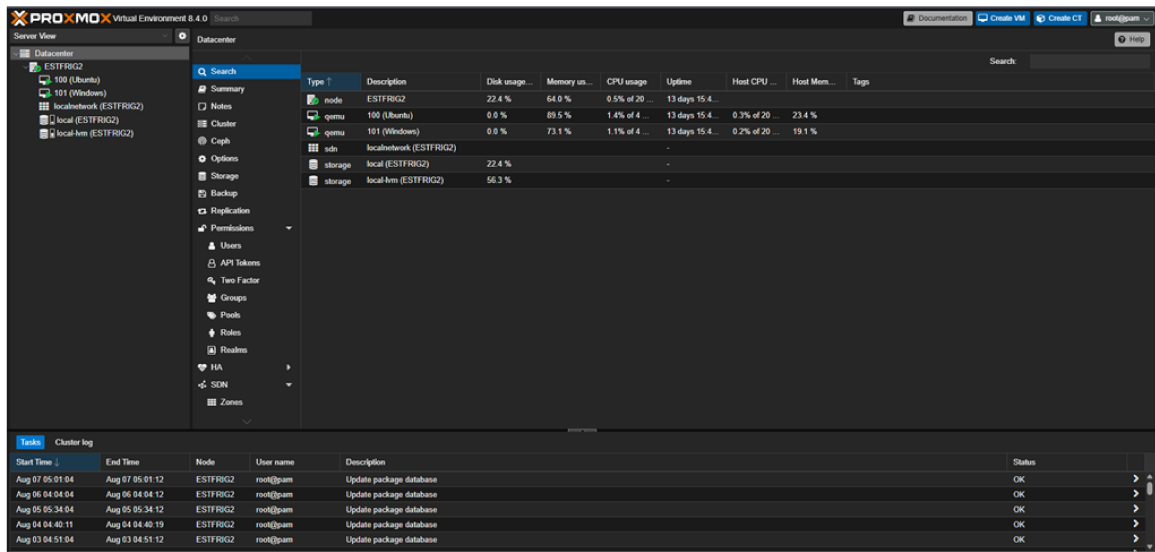
- IP: 172.168.30.45

- Gateway: 172.168.30.0
- DNS: 172.168.29.100

A seguir, será exibido um resumo com as informações configuradas. Clique em "Install" e aguarde a conclusão do processo de instalação. Ao final, o sistema exibirá uma mensagem confirmando a conclusão e solicitando a remoção do pendrive. No próximo boot, o sistema apresentará o endereço IP configurado inicialmente e a porta de acesso, permitindo a conexão a partir de outro dispositivo na mesma rede, utilizando a senha definida anteriormente. No primeiro acesso, o usuário padrão será "root".

Na tela inicial do Proxmox, conforme ilustrado na figura 3, prosseguiu-se com a configuração. Inicialmente, baixaram-se as imagens ISO necessárias: neste caso, utilizaram-se o Windows 10 e o Ubuntu Linux. Após o download, na interface principal do Proxmox, clicou-se com o botão direito e selecionou-se a opção "Criar VM". Em seguida, escolheu-se a ISO correspondente para iniciar o processo de criação da máquina virtual.

Figura 3 – Tela principal do Proxmox.



Fonte: Autoria própria (2025).

Na primeira máquina virtual (VM), baseada no sistema operacional Windows, foi realizado o download do software KEPServerEX. A configuração detalhada desse software será abordada na próxima seção, com um guia passo a passo para garantir a correta aquisição de dados. A segunda VM, baseada no sistema operacional Linux (Ubuntu), foi configurada para hospedar os serviços auxiliares do sistema de automação. Para isso, optou-se pela utilização do Portainer como interface de orquestração e gerenciamento de containers Docker, proporcionando maior eficiência e organização no controle dos serviços.

3.3 VM windows - configurando KEPServerEX

O software foi instalado em uma máquina virtual dedicada com sistema operacional Windows, devido à sua compatibilidade exclusiva com esse sistema, conforme especificações técnicas oficiais. Essa abordagem proporciona isolamento, estabilidade e escalabilidade para projetos futuros, além de garantir conformidade com o suporte técnico e o licenciamento da PTC (Power To Create). Conforme destacado por um usuário da comunidade PTC, identificado como AB_7478123, “o KEPServerEX é um aplicativo baseado em Windows, enquanto o TKE é compatível apenas com sistemas Linux. O TKE possui drivers para conexão com Ethernet Allen-Bradley ControlLogix, Ethernet Modbus e Ethernet Siemens TCP/IP, sendo incompatível com outros drivers” ([PTC Community, 2025](#)). Assim, optou-se por manter o KEPServerEX em uma VM dedicada para assegurar compatibilidade e robustez.

Para a configuração do KEPServerEX, foram adicionados os controladores lógicos programáveis (CLPs) e as tags necessárias para a aquisição de dados. Na seção de conectividade, foi selecionado o driver correspondente ao CLP, especificamente o Allen-Bradley ControlLogix Ethernet, compatível com o modelo MicroLogix 1100. Um canal foi criado com o nome “ESTUFAS_BACON”. Na configuração de dados, optou-se por utilizar apenas o valor mais recente para todas as tags, mantendo o Duty Cycle e as demais configurações padrão até a exibição da tela de configuração, conforme ilustrado na Figura 4.

Figura 4 – Configuração Device KEPServerEX.

← Add Device Wizard

Identification

Name

Device 1

Description

Driver

Allen-Bradley ControlLogix Ethernet

Model

MicroLogix 1100

Channel Assignment

Channel1

ID

<172.168.30.1>

Operating Mode

Data Collection

Enable

Simulated

No

Scan Mode

Scan Mode

Respect Client-Specified Scan Rate

Initial Updates from Cache

Disable

Communication Timeouts

Connect Timeout (s)

3

Request Timeout (ms)

1000

Attempts Before Timeout

3

Timing

Inter-Request Delay (ms)

0

Auto-Demotion

Demote on Failure

Disable

Tag Generation

On Device Startup

Do Not Generate on Startup

On Duplicate Tag

Delete on Create

Parent Group

Allow Automatically Generated Subgroups

Enable

EtherNet/IP

TCP/IP Port

44818

CIP

Logix

Protocol Options

Project Options

Database Import Method

Logix Database Options

Logix Database Filtering

Impose Array Limit

Disable

Array Count Upper Limit

2000

ENI DF1/DH+/CN Gtway Comm. Parameters

TCP/IP Port

44818

Request Size (bytes)

232

Allow Function File Block Writes

Disable

Logix

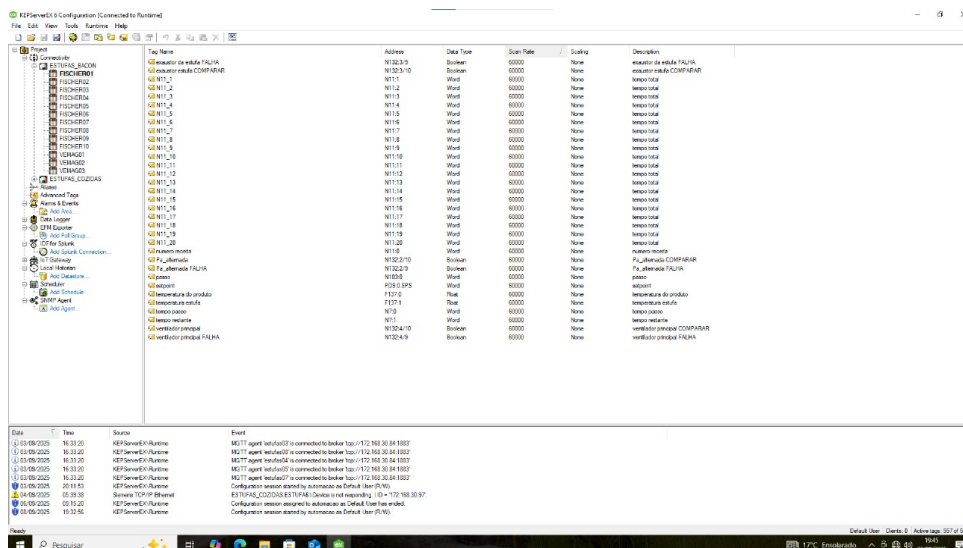
Concluir

Cancelar

Fonte: Autoria própria (2025).

Com a criação do canal de conectividade concluída, o próximo passo foi configurar o dispositivo. Abaixo do canal criado, um dispositivo foi adicionado com o nome “FISHER01”. O modelo do CLP foi selecionado, e o endereço IP foi definido como 172.168.30.1 para fins de demonstração. As configurações padrão foram mantidas, com a possibilidade de ajustes pontuais, se necessário. Após a configuração da conexão com o CLP, as tags desejadas foram adicionadas como mostra na Figura 5.

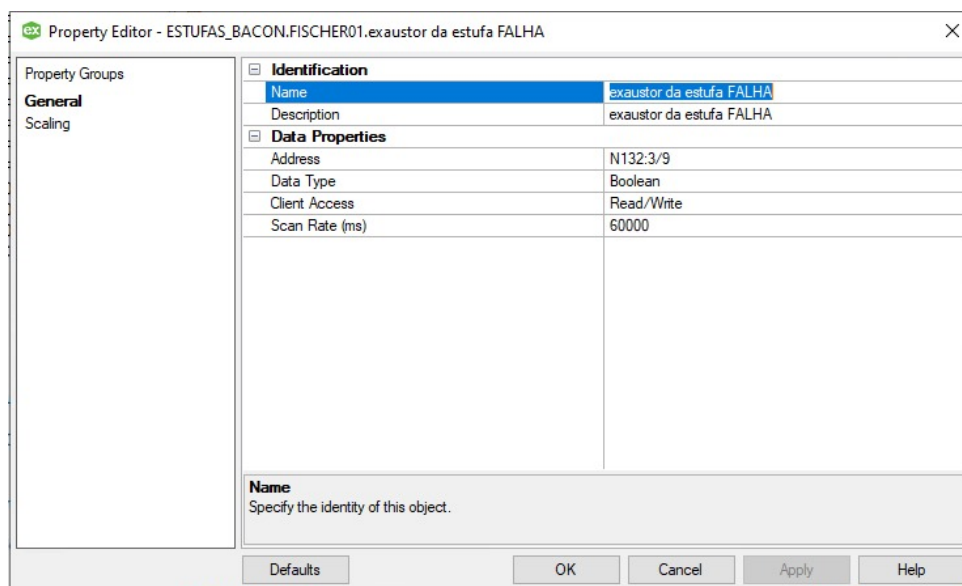
Figura 5 – Configuração KEPServerEX CLP.



Fonte: Autoria própria (2025).

Ao acessar a tela do dispositivo, uma nova tag foi criada clicando com o botão direito e selecionando “New Tag”. Com o software do CLP aberto, foram definidos o nome da tag, descrição, endereço, tipo de dado, permissões de leitura/escrita e a taxa de varredura (scan rate) em milissegundos. Um exemplo de tag configurada, “exaustor da estufa FALHA”, é ilustrado na Figura 6.

Figura 6 – Configuração de uma Tag no KepServerEx.

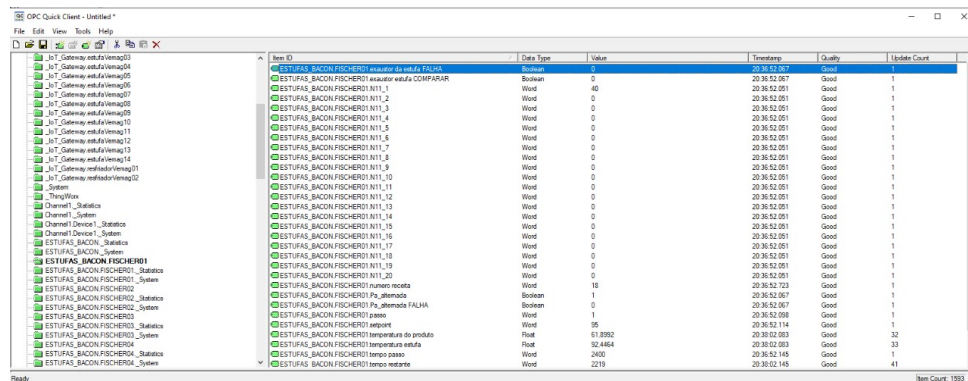


Fonte: Autoria própria (2025).

A comunicação via protocolos OPC UA e OPC DA, suportada pelo KEPServerEX, facilita a coleta de dados em tempo real, sendo compatível com a maioria dos hardwares industriais, o que simplifica a unificação de dados. Para verificar a conexão, foi utilizado

o recurso “Quick Client”, que permite comparar os dados com um backup do projeto, se disponível, ou verificar a qualidade da conexão pela coluna ”Quality”. Conforme mostrado na Figura 7, o status ”Good” indica uma conexão bem-sucedida.

Figura 7 – Quick Client.



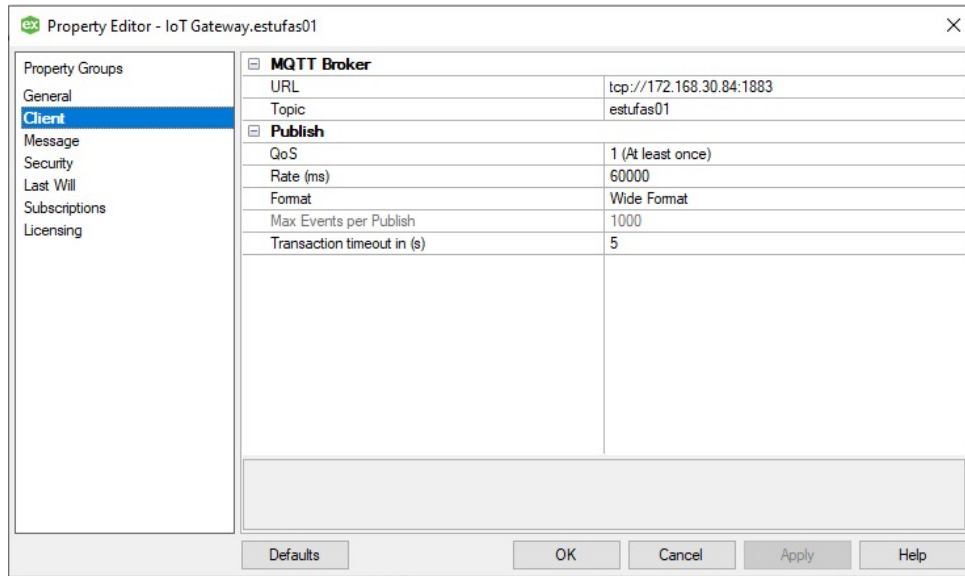
Item ID	Data Type	Value	Timestamp	Quality	Update Count
ESTUFAS_BACON_FISCHER01_sensador da estufa TALHA	Boolean	0	20/05/2021	Good	1
ESTUFAS_BACON_FISCHER01_sensador vazio COMPARAR	Boolean	0	20/05/2021	Good	1
ESTUFAS_BACON_FISCHER01_N11_1	Word	40	20/05/2021	Good	1
ESTUFAS_BACON_FISCHER01_N11_2	Word	0	20/05/2021	Good	1
ESTUFAS_BACON_FISCHER01_N11_3	Word	0	20/05/2021	Good	1
ESTUFAS_BACON_FISCHER01_N11_4	Word	0	20/05/2021	Good	1
ESTUFAS_BACON_FISCHER01_N11_5	Word	0	20/05/2021	Good	1
ESTUFAS_BACON_FISCHER01_N11_6	Word	0	20/05/2021	Good	1
ESTUFAS_BACON_FISCHER01_N11_7	Word	0	20/05/2021	Good	1
ESTUFAS_BACON_FISCHER01_N11_8	Word	0	20/05/2021	Good	1
ESTUFAS_BACON_FISCHER01_N11_9	Word	0	20/05/2021	Good	1
ESTUFAS_BACON_FISCHER01_N11_10	Word	0	20/05/2021	Good	1
ESTUFAS_BACON_FISCHER01_N11_11	Word	0	20/05/2021	Good	1
ESTUFAS_BACON_FISCHER01_N11_12	Word	0	20/05/2021	Good	1
ESTUFAS_BACON_FISCHER01_N11_13	Word	0	20/05/2021	Good	1
ESTUFAS_BACON_FISCHER01_N11_14	Word	0	20/05/2021	Good	1
ESTUFAS_BACON_FISCHER01_N11_15	Word	0	20/05/2021	Good	1
ESTUFAS_BACON_FISCHER01_N11_16	Word	0	20/05/2021	Good	1
ESTUFAS_BACON_FISCHER01_N11_17	Word	0	20/05/2021	Good	1
ESTUFAS_BACON_FISCHER01_N11_18	Word	0	20/05/2021	Good	1
ESTUFAS_BACON_FISCHER01_N11_19	Word	0	20/05/2021	Good	1
ESTUFAS_BACON_FISCHER01_N11_20	Word	0	20/05/2021	Good	1
ESTUFAS_BACON_FISCHER01_numero_reato	Word	18	20/05/2021	Good	1
ESTUFAS_BACON_FISCHER01_Pa_alternado	Boolean	1	20/05/2021	Good	1
ESTUFAS_BACON_FISCHER01_Pa_alternado TALHA	Boolean	0	20/05/2021	Good	1
ESTUFAS_BACON_FISCHER01_passeio	Word	1	20/05/2021	Good	1
ESTUFAS_BACON_FISCHER01_sensore	Word	99	20/05/2021	Good	1
ESTUFAS_BACON_FISCHER01_temperatura do produto	Float	51.8992	20/05/2021	Good	32
ESTUFAS_BACON_FISCHER01_temperatura estufa	Float	32.4664	20/05/2021	Good	33
ESTUFAS_BACON_FISCHER01_tempo passo	Word	2400	20/05/2021	Good	1
ESTUFAS_BACON_FISCHER01_tempo material	Word	2219	20/05/2021	Good	41

Fonte: Autoria própria (2025).

Agora já que temos as informações necessárias das tags, temos que envia-las pro NodeRed neste projeto resolvemos utilizar o protocolo MQTT, porém poderíamos usar qualquer outro protocolo como OPC, REST Client, REST Server, ThingWorx etc. A escolha do protocolo MQTT para a comunicação entre o KEPServerEX e o Node-RED fundamenta-se em três aspectos principais: leveza, desempenho em tempo real e flexibilidade de integração.

Para configurar a comunicação com o Node-RED via MQTT, o KEPServerEX foi configurado da seguinte forma: no menu lateral, clicou-se com o botão direito em “IoT Gateway” e selecionou-se “New Agent”. O tipo de agente escolhido foi “MQTT Client”, com o nome “estufas01”. As configurações foram realizadas conforme ilustrado na Figura 8.

Figura 8 – Configuração KEPServerEX MQTT.



Fonte: Autoria própria (2025).

- No campo URL, insira o endereço do broker MQTT. tcp://172.168.30.84:1883
- No campo Topic, insira o nome do tópico onde os dados serão publicados. estufas01.
- QoS (Quality of Service):1 (At least once) para garantir entrega.
- Rate (ms): tempo de varredura de publicação 60000 1x por minuto.
- Format: Wide format.
- Max Events per Publish: 1000
- Format: Wide format
- Transaction Timeout: 5

Desse modo finalizamos a configuração do KepsServerEX e podemos seguir com as demais etapas do Processo.

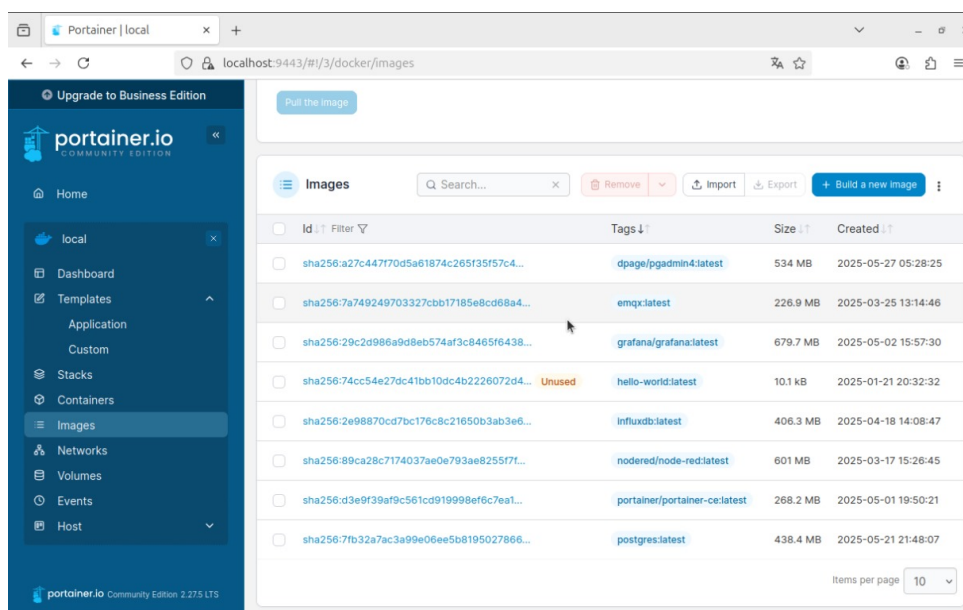
3.4 VM Linux - configurando Docker

Na máquina virtual Linux (Ubuntu), foram instalados todos os serviços necessários para o funcionamento do sistema. A instalação do Docker seguiu o passo a passo descrito no site oficial ([DOCKER, 2025a](#)). Para o Portainer, ferramenta que facilita a gestão de contêineres no Docker, a instalação foi realizada conforme as instruções do site oficial ([PORTAINER, 2025](#))

A escolha do Docker e Portainer para o gerenciamento das ferramentas, foi motivada pela necessidade de isolamento, escalabilidade e portabilidade dos serviços. As ferramentas foram containerizadas para simplificar manutenção, facilitar replicação do ambiente e assegurar reprodutibilidade dos experimentos, garantindo maior robustez e confiabilidade à arquitetura proposta. Diferente das máquinas virtuais tradicionais, que simulam um hardware completo, os contêineres compartilham o kernel do sistema operacional do host e isolam apenas o necessário para a aplicação. Na prática, isso significa que uma aplicação pode ser empacotada junto com todas as suas dependências em uma imagem, garantindo que será executada de forma idêntica em qualquer ambiente, as imagens são como modelos imutáveis que descrevem tudo o que uma aplicação precisa para rodar e os contêineres são como instâncias em execução dessas imagens. Cada contêiner é isolado, mas pode se comunicar com outros, e isso que torna essa ferramenta tão poderosa.

Após a instalação, o Portainer fica disponível em: <https://localhost:9443>, faça login com o usuário administrador configurado na instalação, na primeira tela clique em Add Environment, selecione a opção Docker Standalone > Agent/Local, escolha local para conectar ao Docker instalado na própria máquina e depois informe o socket padrão do Docker: `/var/run/docker.sock`, defina um nome para o ambiente nesse caso optamos por "local", clique em connect e concluímos a configuração. O próximo passo consistiu em adicionar todas as imagens utilizadas. No menu lateral, foi selecionada a opção "Images", podemos dar um Search e pesquisar as imagens que precisamos, na Figura 9 segue a lista das imagens que utilizamos.

Figura 9 – Imagens ISOS.

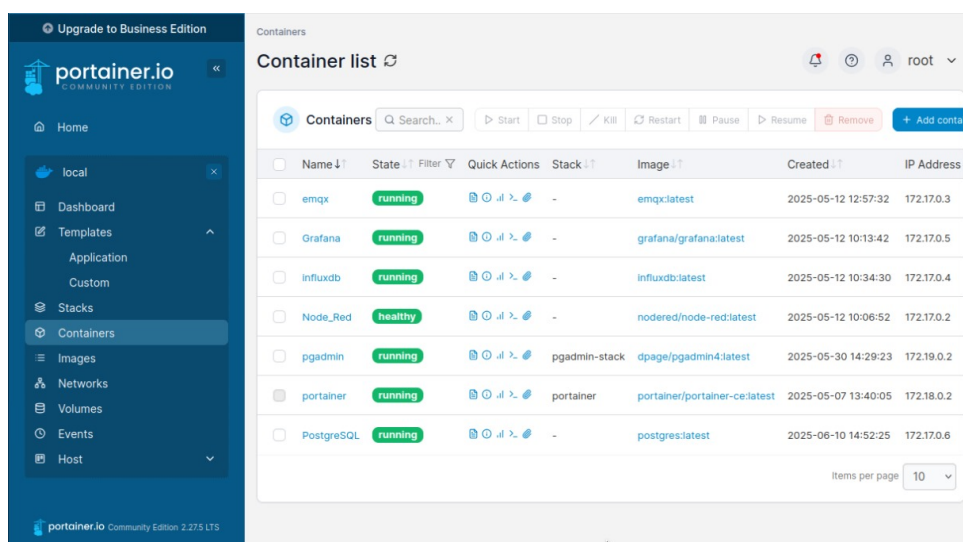


Fonte: Autoria própria (2025).

Após o download das imagens necessárias para o projeto, procedeu-se à criação dos

contêineres que executaram cada serviço no ambiente Docker. No menu lateral, selecionou-se a opção "Containers" e clicou-se em "Add container" para iniciar a configuração de um novo serviço. Definiu-se o nome do contêiner como "Node-RED" (o processo descrito aplica-se ao Node-RED, sendo análogo para os demais serviços). Selecionou-se a imagem correspondente (nodered/node-red:latest), configurando-se corretamente as portas, mantendo-se o padrão 1880:1880 para o Node-RED. Em "Advanced container settings", preservaram-se as configurações padrão, e na opção "Restart policy", selecionou-se "Always". Finalizada a configuração, executou-se o comando "Deploy". Para testar o serviço, acessou-se o navegador com o endereço <http://172.168.30.84:1880>, que abriu a interface do Node-RED. O mesmo processo foi replicado para as demais imagens. A Figura 10 ilustra os contêineres configurados.

Figura 10 – Containers.



Fonte: Autoria própria (2025).

3.5 Node-Red

O Node-RED foi utilizado como principal ferramenta de integração, tratamento e roteamento dos dados obtidos do CLP por meio do KepServerEX. Nele os dados chegam através do protocolo MQTT e foi configurado para receber individualmente de cada estufa. Após receber os dados, foi criada uma função para o tratamento onde se agrupa os dados de forma estruturada e realizamos o envio pro banco de dados. Como a configuração é idêntica para todas as estufas, adotaremos a Estufa01 como exemplo de referência, evitando repetições desnecessárias ao longo da explicação.

O primeiro nó a ser configurado é o MQTT in, por ele as informações chegam ao nosso fluxo. O KepeserEX é um MQTT client e o NodeRed também por isso quando criamos as imagens no Portainer, configuramos um broker MQTT, o EMQX. O EMQX (Eclipse MQTT eXtended) é um broker MQTT de alto desempenho, projetado para intermediar a

comunicação entre dispositivos, sistemas e aplicações. Seu papel é atuar como servidor central no protocolo MQTT (Message Queuing Telemetry Transport), garantindo a troca eficiente de mensagens publicadas por produtores (publishers) e consumidas por assinantes (subscribers). (TECHNOLOGIES, 2025)

O nó foi identificado com o nome "estufas01", garantindo que sua função no fluxo de dados fosse facilmente identificável. O endereço do broker MQTT foi definido como tcp://172.168.30.84. Um aspecto crítico da configuração foi a definição do tópico de assinatura (subscribe), para garantir a correta recepção dos dados provenientes do KepsServerEX, o tópico configurado no nó de entrada (estufas01), além disso, o parâmetro Quality of Service (QoS) foi configurado para o nível 1 (QoS 1). Esta escolha é fundamental para o funcionamento do sistema, pois garante que cada mensagem seja entregue ao menos uma vez ao destinatário, oferecendo um compromisso ideal entre confiabilidade e overhead na rede. A utilização do mesmo nível de QoS em ambos os endpoints da comunicação (KepsServerEX e Node-RED) é um requisito operacional indispensável para o correto funcionamento da fila de mensagens. Com esses parâmetros devidamente estabelecidos, a configuração do nó de entrada MQTT foi concluída, estabelecendo um canal de comunicação eficiente e confiável para a aquisição dos dados em tempo real, na figura 11 mostra como ficou a configuração do nó. Todos os códigos utilizados encontram-se em anexo.

Figura 11 – Nó MQTT.

Fonte: Autoria própria (2025).

Após a recepção dos dados, o fluxo é direcionado para um nó Function chamado Tratamento, responsável pela normalização e filtragem das informações recebidas. Nesse estágio, os estados dos dispositivos controlados, como a pá alternada, o ventilador principal e o exaustor da estufa, são convertidos em valores textuais de fácil compreensão, sendo classificados como "ligado", "desligado" ou "falha". Além disso, são realizados cálculos essenciais ao processo, como a soma das variáveis N11/1 a N11/20, que resulta no tempo

total de operação, e a conversão de tempos de processo, como tempo passo e tempo restante, de segundos para minutos. Por fim, o nó seleciona apenas as variáveis relevantes para a análise, incluindo número da receita, passo atual, setpoint e temperaturas, gerando um conjunto de dados organizado, limpo e padronizado.

Na sequência, os dados tratados passam por outro nó Function denominado Ajuste de Dados, responsável por adequar as informações ao formato exigido para inserção no banco de dados. Nesse estágio, cada variável é transformada em um registro individual, contendo o tópico de origem da mensagem, o identificador da tag e o valor correspondente. Esse processo garante granularidade e organização, permitindo que cada ponto de informação seja devidamente armazenado de forma independente e estruturada. Essa parte é essencial pois garante a organização de como os dados chegam no banco, garantindo qualidade e organização para retirada de relatórios e até mesmo nas visualizações em tempo real.

Por fim, os dados processados são encaminhados ao nó PostgreSQL, que realiza a execução de comandos SQL para inserção na tabela destinada ao histórico do processo (Os comandos serão explicados no tópico PostgreSQL). A cada nova mensagem recebida, as variáveis selecionadas são registradas acompanhadas do carimbo de tempo da inserção, do tópico de origem e de seus respectivos valores. Esse procedimento estabelece um sistema de historização contínua, assegurando a persistência e a rastreabilidade das informações. Essa etapa é essencial para possibilitar análises posteriores por meio de ferramentas de visualização, como o Grafana, que permitem o monitoramento em tempo real e a obtenção de indicadores de desempenho do processo.

A configuração deste nó exige a criação de um objeto de conexão, no qual são definidos os seguintes parâmetros:

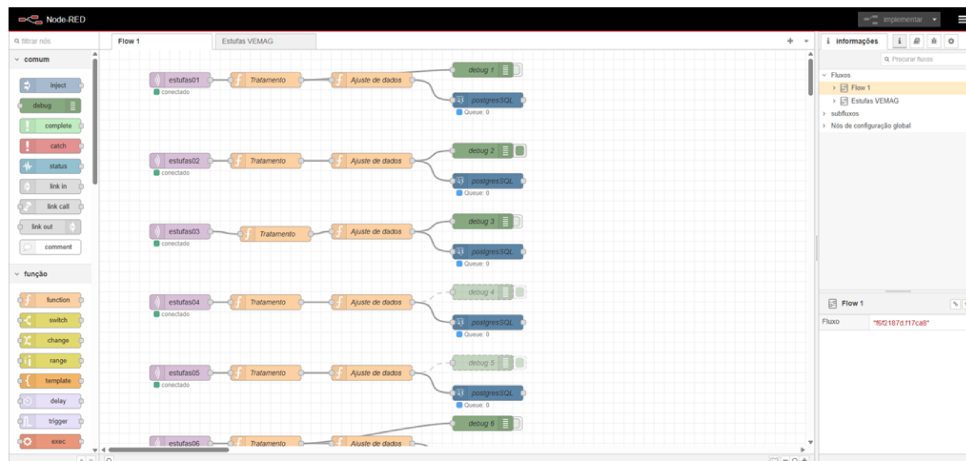
- Host: endereço IP do servidor onde o PostgreSQL está em execução (neste caso, 172.168.30.84).
- Porta: a porta padrão de comunicação do PostgreSQL, definida como 5432.
- Database: o nome do banco de dados criado para armazenar os registros, definido como postgres.
- Parâmetros adicionais: como número máximo de conexões simultâneas (max = 10), tempo limite de inatividade (idle = 1000 ms) e tempo limite de conexão (10000 ms).

No campo Query do nó, foi especificado o comando SQL responsável pela inserção dos dados.

```
INSERT INTO estufas (time, topico, tag, valor)
VALUES (NOW(), '{{{msg.topic}}}', '{{{msg.tag}}}', '{{{msg.payload}}});
```

Utilizamos o nó MQTT, dois blocos de função pra tratamento dos dados e organização e por último o nó PostgreSQL, na Figura 12 mostra o fluxo completo do NodeRed.

Figura 12 – Fluxo NodeRed



Fonte: Autoria própria (2025).

3.6 PostgreSQL

A modelagem do banco foi estruturada para registrar, de forma temporal, as variáveis monitoradas em cada estufa, incluindo temperaturas, estados de operação e eventos de processo. Essa estrutura garante que, em eventuais auditorias, seja possível recuperar informações precisas de períodos específicos, eliminando falhas comuns do registro manual.

A etapa inicial para estruturar o armazenamento de dados consistiu na criação do banco no PostgreSQL. Esse processo foi realizado utilizando o pgAdmin, ferramenta oficial de administração gráfica do PostgreSQL, que possibilita a gestão de usuários, permissões e tabelas de maneira intuitiva.

No pgAdmin, foi criada uma nova Base de Dados chamada postgres, dedicada ao projeto. Em seguida, configurou-se um usuário específico (brf) com senha própria, assegurando isolamento e segurança de acesso. Esse usuário recebeu privilégios de leitura e escrita apenas na base vinculada ao projeto, seguindo boas práticas de controle de permissões.

Posteriormente, foi criada a tabela responsável por armazenar os dados das estufas, denominada estufas, com a seguinte estrutura:

```
CREATE TABLE estufas (
  id SERIAL PRIMARY KEY,
  time TIMESTAMP NOT NULL,
```

```

topico VARCHAR(100) NOT NULL,
tag VARCHAR(100) NOT NULL,
valor FLOAT NOT NULL
);

```

- id: chave primária incremental que identifica unicamente cada registro.
- time: armazena o instante de inserção da informação.
- topico: campo que registra o tópico MQTT, qual estufa o dado pertence.
- tag: identifica a variável coletada (ex: temperatura, passo, tempo restante).
- valor: armazena o valor associado à variável em questão.

A estruturação da tabela foi planejada para otimizar a filtragem e a geração de relatórios, permitindo a seleção de estufas específicas, bem como a consulta por data e hora, o que facilita auditorias internas e externas. Outro aspecto relevante foi a adoção do pgAdmin, que simplificou a criação e a gestão das estruturas de dados, além de possibilitar o monitoramento em tempo real das inserções realizadas pelo Node-RED. Essa abordagem contribuiu significativamente para a validação do fluxo de dados e para a garantia da confiabilidade na persistência das informações. Na figura 13 evidencia a organização e a clareza na apresentação dos dados, facilitando sua compreensão e análise.

Figura 13 – Tabela PgAdmin.

id	time	topico	tag	valor
21	19209333 2025-09-08 14:13:20.521813	estufas08	passo	4
22	19209332 2025-09-08 14:13:20.521769	estufas08	numero_receita	2
23	19209331 2025-09-08 14:13:20.521727	estufas04	exaustor_da_estufa	desligado
24	19209330 2025-09-08 14:13:20.521682	estufas04	ventilador_principal	desligado
25	19209329 2025-09-08 14:13:20.521636	estufas04	pia_alternada	desligado
26	19209328 2025-09-08 14:13:20.521327	estufas04	tempo_total	480
27	19209327 2025-09-08 14:13:20.521284	estufas04	tempo_restante	1
28	19209326 2025-09-08 14:13:20.521241	estufas04	tempo_passo	1
29	19209325 2025-09-08 14:13:20.521187	estufas04	temperatura_do_produto	51.9333649
30	19209324 2025-09-08 14:13:20.521182	estufas04	temperatura_estufa	34.1878815
31	19209323 2025-09-08 14:13:20.520809	estufas04	setpoint	0

Fonte: Autoria própria (2025).

3.7 Grafana

No projeto, o Grafana foi configurado para se conectar diretamente ao banco de dados PostgreSQL, de onde obtém as informações processadas e armazenadas pelo Node-

RED. A partir dessa conexão, foram criados painéis personalizados que exibem em tempo real as variáveis críticas do processo de cocção, como temperaturas e status operacionais de cada estufa.

O processo de conexão é feito através da adição de uma nova data source no Grafana, utilizando o conector nativo para PostgreSQL. Primeiramente, acessa-se o painel administrativo do Grafana e, no menu lateral, seleciona-se a opção Connections > Data sources. Em seguida, escolhe-se Add data source e seleciona-se o tipo PostgreSQL. Dentro da tela de configuração, devem ser informados os parâmetros de conexão com o banco, como:

- name: grafana-postgresql-datasource
- Host: 172.168.30.84- endereço ip do banco
- Database name: postgres - nome do banco
- Authentication: Usuario e senha.

Após o preenchimento das informações, basta clicar em Save e Test para validar a conexão. Caso os parâmetros estejam corretos, o Grafana confirmará o acesso ao banco de dados.

Em cada visualização do dashboard foi definida uma consulta SQL (query) específica, configurada de forma a garantir que os dados sejam apresentados de maneira correta e contextualizada. Essas consultas foram ajustadas para cada tipo de painel, possibilitando a exibição de gráficos de tendência, valores instantâneos de temperatura e informações textuais, como o nome da receita atualmente em execução na estufa selecionada. Esse detalhamento na configuração das queries assegura que cada métrica exibida seja precisa, atualizada e diretamente relacionada ao processo monitorado.

Como o sistema conta com 13 estufas, a criação de dashboards individuais para cada uma delas resultaria em retrabalho e manutenção desnecessária. Para evitar essa repetição, foi implementada no Grafana uma variável dinâmica vinculada à coluna topico do banco de dados. Dessa forma, ao selecionar o número da estufa desejada no filtro do painel, o Grafana substitui automaticamente a variável na consulta SQL, retornando apenas os dados correspondentes àquela estufa. Os códigos de cada visualização serão inseridos em forma de anexo. A criação dessa variável é bastante simples: No menu do dashboard, acessa-se Settings > Variables. Cria-se uma nova variável do tipo Query e no campo de consulta, define-se:

```
SELECT DISTINCT topico FROM estufas ORDER BY topico;
```

Isso garante que todos os valores únicos da coluna *topico* (ex.: *estufa01*, *estufa02*, ..., *estufa13*) sejam listados. Com esse recurso, o mesmo dashboard atende a todas as estufas, bastando selecionar o número desejado no menu suspenso a esquerda. Além de reduzir o tempo de manutenção, essa configuração garante padronização das visualizações, e qualquer ajuste futuro é feito apenas uma vez e automaticamente refletido em todas as estufas. A Figura 14 apresenta o dashboard final, com destaque para a variável selecionada referente à respectiva estufa. Um aspecto importante a ser considerado é a utilização, em cada query, de uma cláusula de filtro no formato:

```
WHERE topico = '$topico'
```

As configurações foram atendidas conforme mencionado anteriormente. Os códigos de cada painel estão apresentados nos anexos.

Figura 14 – Seleção da Estufa Grafana.



Fonte: Autoria própria (2025).

4 Resultados e Discussões

A implementação da arquitetura IIoT para o monitoramento do processo de cocção industrial de bacon permitiu alcançar resultados relevantes no contexto fabril. Antes do desenvolvimento, o acompanhamento das variáveis era realizado de forma manual, por meio de anotações em planilhas físicas. Esse método apresentava limitações significativas, como suscetibilidade a falhas humanas, perda de registros e dificuldade de rastreabilidade em auditorias ou situações relacionadas à qualidade do produto.

Com a implantação do sistema, foi possível atender a um conjunto de requisitos definidos como metas do projeto, que podem ser apresentados em forma de checklist:

- **Aquisição automática de dados de processo:** as variáveis críticas (temperatura das estufas e do produto) passaram a ser coletadas em tempo real, eliminando a necessidade de registros manuais.
- **Historização em banco de dados relacional:** os dados coletados foram armazenados de forma estruturada, possibilitando rastrear lotes e analisar a evolução do processo ao longo do tempo.
- **Visualização por dashboards:** o sistema disponibilizou as informações em painéis gráficos acessíveis em qualquer ponto da planta, reduzindo a necessidade de deslocamento da equipe até o campo.
- **Rastreabilidade e auditoria:** os registros históricos passaram a estar organizados e disponíveis para consultas, facilitando auditorias e atendendo às exigências do SIF.
- **Eficiência operacional:** a centralização das informações e a visualização em tempo real reduziram o tempo de resposta a não conformidades, além de proporcionar suporte à manutenção preditiva.

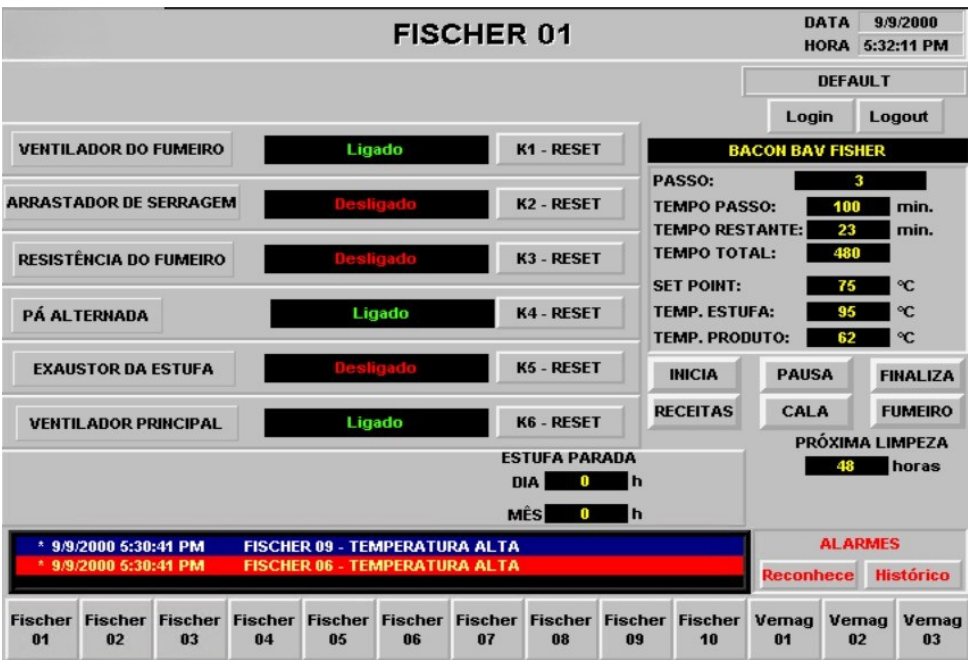
A partir desse checklist de funcionalidades implementadas, pode-se observar que o sistema trouxe impactos diretos na confiabilidade, na rastreabilidade e na eficiência operacional do processo de cocção industrial. A seguir, discute-se em maior detalhe os benefícios alcançados e os pontos críticos observados durante a aplicação da solução.

4.1 Comparativo entre a situação anterior e a solução proposta

4.1.1 Supervisão e monitoramento

Antes da implementação do projeto, o acompanhamento do processo de cocção era possível apenas por meio das duas IHMs instaladas em campo (Figura 15). A equipe de qualidade, para realizar a aferição da temperatura, precisava se deslocar fisicamente até o local para obter as informações.

Figura 15 – IHM Processo Bacon.



Fonte: Autoria própria (2025).

Após a implementação do sistema supervisorio, os dados passaram a ser disponibilizados em um dashboard central (Figura 16), permitindo seu acesso a partir de qualquer ponto da fábrica. Essa mudança trouxe maior agilidade e eficiência ao processo de monitoramento e reduziu a dependência de acesso físico às estufas.

Figura 16 – Dashborard Estufas Bacon.



Fonte: Autoria própria (2025).

4.1.2 Gestão operacional e suporte à decisão

Um segundo ponto de destaque foi a otimização da gestão operacional. O novo sistema passou a registrar automaticamente os dados de temperatura, tempo e ciclos de cocção em um banco de dados centralizado, permitindo a análise histórica e comparativa entre lotes. Isso possibilitou à equipe de produção e qualidade tomar decisões de forma mais rápida e embasada, com base em dados confiáveis e atualizados em tempo real.

4.1.3 Conformidade regulatória e rastreabilidade (SIF)

Um terceiro ponto, em relação às normas do Serviço de Inspeção Federal (SIF), que exigem a rastreabilidade completa dos processos térmicos aplicados a produtos de origem animal, a arquitetura IIoT desenvolvida neste projeto passou a atender plenamente a esse requisito. O sistema gera registros automáticos de todos os ciclos de cocção, vinculando-os aos lotes produzidos e garantindo sua rastreabilidade para fins de auditorias internas e externas.

4.1.4 Aplicação do Sistema ao Processo de Cocção de Linguiças

Por fim, um dos principais resultados do projeto foi mostrar que a arquitetura desenvolvida pode ser reaproveitada em outros processos. A mesma estrutura usada no cozimento do bacon foi aplicada em uma linha de produção de linguiças, comprovando que o sistema é flexível e funciona bem em diferentes situações dentro da indústria. Com isso, ficou visível que a solução não se limita a um único processo, mas pode ser adaptada de forma prática e eficiente, o que aumenta o valor do trabalho para empresas de grande porte que precisam de padronização e confiabilidade nas informações.

A Figura 17 mostra o dashboard utilizado no processo de cocção de linguiças cozidas, confirmando a replicação do sistema e os benefícios já alcançados, como rastreabilidade, acessibilidade e organização dos dados.

Figura 17 – Dashboard Estufas Cozidas



Fonte: Autoria própria (2025).

5 Conclusão

O presente trabalho teve como objetivo o desenvolvimento de uma solução para o historiamento e acompanhamento em tempo real do processo de cocção de bacon, utilizando um dashboard integrado a banco de dados. Essa proposta surgiu da necessidade de atender às exigências do Serviço de Inspeção Federal (SIF), bem como de otimizar a gestão de informações de produção, antes restritas às duas IHMs existentes no campo.

A solução implementada permitiu que a equipe de qualidade passasse a acessar as informações de forma remota, em qualquer ponto da fábrica, eliminando a necessidade de deslocamentos até a área de produção. Além disso, a inclusão do historiamento garante maior rastreabilidade de lotes e confiabilidade na verificação de eventuais não conformidades, tornando o processo mais seguro e aderente às normas vigentes.

Outro ponto relevante foi a replicação do sistema em uma linha de produção destinada ao cozimento de linguças, o que demonstrou a flexibilidade e escalabilidade da arquitetura desenvolvida. A mesma estrutura de dashboard e banco de dados mostrou-se facilmente adaptável a diferentes processos industriais, reforçando a aplicabilidade do trabalho em empresas do setor alimentício que demandam padronização, confiabilidade e rastreabilidade em seus processos.

Dessa forma, pode-se concluir que o projeto atingiu seus objetivos iniciais, ao fornecer uma solução de baixo custo, confiável e de alto impacto, alinhada tanto às necessidades de inspeção do SIF quanto às diretrizes internas da corporação. Além de evitar altos custos com retrofit, o sistema se mostrou inovador, escalável e viável para futuras aplicações em outras linhas de produção.

Como continuidade deste trabalho, sugere-se a implementação de novos módulos no dashboard, como relatórios customizáveis, alarmes inteligentes e análises estatísticas em tempo real, ampliando ainda mais a utilidade da solução. Tais evoluções possibilitarão ganhos adicionais de eficiência, rastreabilidade e suporte à tomada de decisão dentro do ambiente fabril.

Referências Bibliográficas

AINSWORTH, T. et al. A comprehensive survey of industry 4.0, iiot and areas of implementation. In: *IEEE SoutheastCon*. Charlotte: University of North Carolina at Charlotte, 2021. Disponível em: <<https://ieeexplore.ieee.org/document/9401860>>. Acesso em: 19 jun. 2025. Disponível em: <<https://ieeexplore.ieee.org/document/9401860>>. Citado na página 10.

Cavli Wireless. *What is the MQTT Protocol*. 2025. Disponível em: <<https://www.cavliwireless.com/blog/nerdiest-of-things/what-is-the-mqtt-protocol>>. Acesso em: 8 de setembro de 2025. Citado na página 17.

CBR, S. *Sistemas de supervisão*. 2025. Disponível em: <<https://www.sitecbr.com.br/produto/sistemas-de-supervisao>>. Acesso em: 4 out. 2025. Disponível em: <<https://www.sitecbr.com.br/produto/sistemas-de-supervisao>>. Citado na página 15.

DOCKER. *Install Docker Desktop on Ubuntu*. 2025. Disponível em: <<https://docs.docker.com/desktop/setup/install/linux/ubuntu/>>. Acesso em: 8 set. 2025. Disponível em: <<https://docs.docker.com/desktop/setup/install/linux/ubuntu/>>. Citado na página 31.

DOCKER. *Install Docker Engine on Ubuntu*. 2025. Disponível em: <<https://docs.docker.com/engine/install/ubuntu/>>. Acesso em: 10 ago. 2025. Disponível em: <<https://docs.docker.com/engine/install/ubuntu/>>. Citado na página 21.

HAT, R. *What is a hypervisor?* 2025. Disponível em: <<https://www.redhat.com/pt-br/topics/virtualization/what-is-a-hypervisor>>. Acesso em: 4 out. 2025. Disponível em: <<https://www.redhat.com/pt-br/topics/virtualization/what-is-a-hypervisor>>. Citado na página 19.

IBERDROLA. *O que é a Internet Industrial das Coisas (IIoT)?* 2025. Disponível em: <<https://www.iberdrola.com/quem-somos/nosso-modelo-inovacao/o-que-e-iiot>>. Acesso em: 4 out. 2025. Disponível em: <<https://www.iberdrola.com/quem-somos/nosso-modelo-inovacao/o-que-e-iiot>>. Citado na página 14.

IBM. *O que é virtualização?* 2025. Disponível em: <<https://www.ibm.com/br-pt/topics/virtualization>>. Acesso em: 10 ago. 2025. Disponível em: <<https://www.ibm.com/br-pt/topics/virtualization>>. Citado na página 20.

IBM. *Virtualização: conceito e aplicações*. 2025. Disponível em: <<https://www.ibm.com/br-pt/think/topics/virtualization>>. Acesso em: 4 out. 2025. Disponível em: <<https://www.ibm.com/br-pt/think/topics/virtualization>>. Citado na página 19.

KEPSERVEREX. *Conecte, aceda e controle seus dados industriais*. 2020. Disponível em: <<https://www.kepserverex.pt/caracteristicas-do-kepserverex/>>. Acesso em: 18 jun. 2025. Disponível em: <<https://www.kepserverex.pt/caracteristicas-do-kepserverex/>>. Citado na página 16.

KUMARESAN, K. et al. Iiot based remote monitoring and control of bottle filling process using plc and node-red. In: *2023 4th International Conference on Communication, Computing and Industry 6.0 (C2I6)*. IEEE, 2023. Disponível em: <<https://ieeexplore.ieee.org/document/10430762>>. Acesso em: 19 jun. 2025. Disponível em: <<https://ieeexplore.ieee.org/document/10430762>>. Citado na página 10.

LABS, G. *Grafana: dashboard anything. Observe everything*. 2025. Disponível em: <<https://grafana.com/grafana/>>. Acesso em: 19 jun. 2025. Disponível em: <<https://grafana.com/grafana/>>. Citado na página 18.

MATKE, M.; KUMAR, S.; SINGH, U. An empirical evaluation of machine learning algorithms for intrusion detection in iiot networks. In: *2023 IEEE 20th India Council International Conference (INDICON)*. IEEE, 2023. Disponível em: <<https://ieeexplore.ieee.org/document/10440779>>. Acesso em: 19 jun. 2025. Disponível em: <<https://ieeexplore.ieee.org/document/10440779>>. Citado na página 10.

NODE-RED. *Programação de baixo código para aplicações orientadas a eventos*. 2013. Disponível em: <<https://nodered.org/>>. Acesso em: 18 jun. 2025. Disponível em: <<https://nodered.org/>>. Citado na página 16.

PECUÁRIA, M. D. A. E. *Legislação de inspeção de produtos de origem animal*. 2025. Disponível em: <<https://www.gov.br/agricultura/pt-br/assuntos/inspecao/produtos-animal/legislacao>>. Acesso em: 8 set. 2025. Disponível em: <<https://www.gov.br/agricultura/pt-br/assuntos/inspecao/produtos-animal/legislacao>>. Citado na página 14.

PECUÁRIA, M. D. A. E. *Serviço de Inspeção Federal (SIF)*. 2025. Disponível em: <<https://www.gov.br/agricultura/pt-br/assuntos/inspecao/produtos-animal/sif>>. Acesso em: 9 set. 2025. Disponível em: <<https://www.gov.br/agricultura/pt-br/assuntos/inspecao/produtos-animal/sif>>. Citado na página 13.

PORTAINER. *Install Portainer CE with Docker on Linux*. 2025. Disponível em: <<https://docs.portainer.io/start/install-ce/server/docker/linux>>. Acesso em: 8 set. 2025. Disponível em: <<https://docs.portainer.io/start/install-ce/server/docker/linux>>. Citado na página 31.

POSTGRESQL. *PostgreSQL*. 2025. Disponível em: <<https://www.postgresql.org/>>. Acesso em: 10 ago. 2025. Disponível em: <<https://www.postgresql.org/>>. Citado 2 vezes nas páginas 17 e 18.

PTC Community. *PTC Community Discussion*. 2025. Disponível em: <<<https://community.ptc.com/>>>. Acesso em: 08 set. 2025. Citado na página 27.

RAKHESH. *Notes on UEFI, GPT, UEFI boot process, disk partitions, and Hyper-V differencing disks with a Generation 2 VM*. 2025. Disponível em: <<https://l1nq.com/46KkH>>. Acesso em: 8 set. 2025. Disponível em: <<https://l1nq.com/46KkH>>. Citado na página 24.

Rufus. *Rufus – criar unidades USB inicializáveis*. 2025. Disponível em: <<https://rufus.ie/pt_BR/>>. Acesso em: 08 set. 2025. Citado na página 24.

SMITH, L. *Proxmox VE 7.4 agora disponível: novos recursos*. 2023. Disponível em: <<https://www.storagereview.com/pt/news/proxmox-ve-7-4-now-available>>.

Acesso em: 18 jun. 2025. Disponível em: <<https://www.storagereview.com/pt/news/proxmox-ve-7-4-now-available>>. Citado 2 vezes nas páginas 20 e 23.

SOLUTIONS, P. S. *Proxmox VE 8.4 ISO installer*. 2025. Disponível em: <<https://www.proxmox.com/en/downloads/proxmox-virtual-environment/iso/proxmox-ve-8-4-iso-installer>>. Acesso em: 8 set. 2025. Disponível em: <<https://www.proxmox.com/en/downloads/proxmox-virtual-environment/iso/proxmox-ve-8-4-iso-installer>>. Citado na página 23.

Super User. *What is a UEFI bootable USB flash drive and can I boot it on legacy BIOS?* 2025. Disponível em: <<<https://superuser.com/questions/1128669/what-is-a-uefi-bootable-usb-flash-drive-and-can-i-boot-it-on-legacy-bios>>>. Acesso em: 08 set. 2025. Citado na página 24.

TECHNOLOGIES, E. *EMQX: The world's most scalable MQTT broker*. 2025. Disponível em: <<https://www.emqx.com/en>>. Acesso em: 7 set. 2025. Disponível em: <<https://www.emqx.com/en>>. Citado na página 34.

Wikipedia. *GUID Partition Table*. 2025. Disponível em: <<https://en.wikipedia.org/wiki/GUID_Partition_Table>>. Acesso em: 08 set. 2025. Citado na página 24.

6 Jason Grafana

Listing 6.1 – Configuração do Dashboard em JSON

```
1 {
2   "annotations": {
3     "list": [
4       {
5         "builtIn": 1,
6         "datasource": {
7           "type": "grafana",
8           "uid": "-- Grafana --"
9         },
10        "enable": true,
11        "hide": true,
12        "iconColor": "rgba(0, 211, 255, 1)",
13        "name": "Annotations & Alerts",
14        "target": {
15          "limit": 100,
16          "matchAny": false,
17          "tags": [],
18          "type": "dashboard"
19        },
20        "type": "dashboard"
21      }
22    ]
23  },
24  "editable": true,
25  "fiscalYearStartMonth": 0,
26  "graphTooltip": 0,
27  "id": 14,
28  "links": [],
29  "panels": [
30    {
31      "datasource": {
32        "type": "grafana-postgresql-datasource",
33        "uid": "beqivwgripam8c"
34      },
35      "fieldConfig": {
36        "defaults": {
```

```
37     "color": {
38         "fixedColor": "orange",
39         "mode": "fixed"
40     },
41     "mappings": [],
42     "thresholds": {
43         "mode": "absolute",
44         "steps": [
45             {
46                 "color": "green",
47                 "value": null
48             },
49             {
50                 "color": "red",
51                 "value": 80
52             }
53         ]
54     },
55     "overrides": []
56 },
57 "gridPos": {
58     "h": 9,
59     "w": 8,
60     "x": 0,
61     "y": 0
62 },
63 "id": 1,
64 "options": {
65     "colorMode": "value",
66     "graphMode": "none",
67     "justifyMode": "center",
68     "orientation": "auto",
69     "percentChangeColorMode": "standard",
70     "reduceOptions": {
71         "calcs": [
72             "lastNotNull"
73         ],
74         "fields": "/^Receita Atual$/",
75         "values": false
76     },
77     "showPercentChange": false,
```

```

79     "text": {
80         "valueSize": 80
81     },
82     "textMode": "auto",
83     "wideLayout": true
84 },
85 "pluginVersion": "11.4.0",
86 "targets": [
87     {
88         "datasource": {
89             "type": "grafana-postgresql-datasource",
90             "uid": "beqivwgripam8c"
91         },
92         "editorMode": "code",
93         "format": "table",
94         "rawQuery": true,
95         "rawSql": "SELECT \r\n    time AS \"1 - Data e Hora\", \r\n
                    valor::integer AS numero_receita, \r\n    CASE (valor
                    ::integer) \r\n        WHEN 0 THEN 'BACON BAV' \r\n        WHEN
                    1 THEN 'BACON SADIA PASSO 1' \r\n        WHEN 2 THEN '
                    BACON SADIA PASSO 2' \r\n        WHEN 3 THEN 'BACON SADIA
                    PASSO 3' \r\n        WHEN 4 THEN 'BACON SADIA PASSO 4' \r\n
                    WHEN 5 THEN 'BACON SADIA PASSO 5' \r\n        WHEN 6
                    THEN 'BACON SADIA PASSO 6' \r\n        WHEN 7 THEN 'BAF
                    PASSO 1' \r\n        WHEN 8 THEN 'BAF PASSO 2' \r\n        WHEN
                    9 THEN 'BAF PASSO 3' \r\n        WHEN 10 THEN 'BAF PASSO
                    4' \r\n        WHEN 11 THEN 'BAF PASSO 5' \r\n        WHEN 12
                    THEN 'BAF PASSO 6' \r\n        WHEN 13 THEN 'BAF PASSO 7' \r\n
                    WHEN 14 THEN 'BAF PASSO 8' \r\n        WHEN 15 THEN
                    'BAF PASSO 9' \r\n        WHEN 16 THEN 'BAF PASSO 10' \r\n
                    WHEN 17 THEN 'BAF PASSO 11' \r\n        WHEN 18 THEN '
                    HIGIENIZA 0' \r\n        WHEN 19 THEN 'MEDI AO' \r\n
                    WHEN 20 THEN 'BAKN PASSO 1' \r\n        WHEN 21 THEN '
                    BAKN PASSO 2' \r\n        WHEN 22 THEN 'BAKN PASSO 3' \r\n
                    WHEN 23 THEN 'BAKN PASSO 4' \r\n        WHEN 24 THEN '
                    BAKN PASSO 5' \r\n        WHEN 25 THEN 'BAKN PASSO 6' \r\n
                    WHEN 26 THEN 'BAKN PASSO 7' \r\n        WHEN 27 THEN '
                    BAKN PASSO 8' \r\n        WHEN 28 THEN 'BAKN PASSO 9' \r\n
                    WHEN 29 THEN 'BAKN PASSO 10' \r\n        WHEN 30 THEN '
                    BAKN PASSO 11' \r\n        WHEN 31 THEN 'BACON TESTE
                    28/02/2024' \r\n        WHEN 32 THEN 'CAVACO 05/03/2025' \r\n
                    ELSE 'Receita Desconhecida' \r\n    END AS \"

```

```

Receita Atual\"r\nFROM r\n  estufas\r\nWHERE\r\n
topico = '${estufas}' AND\r\n  tag = 'numero_receita'
AND\r\n  valor ~ '^\\d+$' AND\r\n  $__timeFilter(
time)\r\nORDER BY r\n  time DESC\r\nLIMIT 1;\r\n",
96  "refId": "A",
97  "sql": {
98    "columns": [
99      {
100        "parameters": [],
101        "type": "function"
102      }
103    ],
104    "groupBy": [
105      {
106        "property": {
107          "type": "string"
108        },
109        "type": "groupBy"
110      }
111    ],
112    "limit": 50
113  }
114 },
115 ],
116 "title": "",
117 "transparent": true,
118 "type": "stat"
119 },
120 {
121   "datasource": {
122     "type": "grafana-postgresql-datasource",
123     "uid": "beqivwgripam8c"
124   },
125   "fieldConfig": {
126     "defaults": {
127       "color": {
128         "mode": "palette-classic"
129       },
130       "displayName": "Setpoint",
131       "fieldMinMax": false,
132       "mappings": [],
133       "thresholds": {

```

```
134         "mode": "absolute",
135         "steps": [
136             {
137                 "color": "green",
138                 "value": null
139             },
140             {
141                 "color": "red",
142                 "value": 80
143             }
144         ]
145     },
146     "unit": "celsius"
147 },
148 "overrides": []
149 },
150 "gridPos": {
151     "h": 12,
152     "w": 4,
153     "x": 8,
154     "y": 0
155 },
156 "id": 7,
157 "options": {
158     "minVizHeight": 75,
159     "minVizWidth": 75,
160     "orientation": "auto",
161     "reduceOptions": {
162         "calcs": [
163             "lastNotNull"
164         ],
165         "fields": "/^Setpoint$/",
166         "values": true
167     },
168     "showThresholdLabels": false,
169     "showThresholdMarkers": false,
170     "sizing": "auto"
171 },
172 "pluginVersion": "11.4.0",
173 "targets": [
174     {
175         "datasource": {
```

```

176         "type": "grafana-postgresql-datasource",
177         "uid": "beqivwgripam8c"
178     },
179     "editorMode": "code",
180     "format": "table",
181     "rawQuery": true,
182     "rawSql": "SELECT \r\n  valor AS \"Setpoint\" \r\nFROM \r\n
      \r\n  estufas\r\nWHERE \r\n  topico = '${estufas}' AND
      \r\n  tag = 'setpoint' AND\r\n  $__timeFilter(time)\r\n
      \r\nORDER BY \r\n  time DESC\r\nLIMIT 1;\r\n",
183     "refId": "A",
184     "sql": {
185         "columns": [
186             {
187                 "parameters": [],
188                 "type": "function"
189             }
190         ],
191         "groupBy": [
192             {
193                 "property": {
194                     "type": "string"
195                 },
196                 "type": "groupBy"
197             }
198         ],
199         "limit": 50
200     }
201 },
202 ],
203 "title": "",
204 "transparent": true,
205 "type": "gauge"
206 },
207 {
208     "datasource": {
209         "type": "grafana-postgresql-datasource",
210         "uid": "beqivwgripam8c"
211     },
212     "fieldConfig": {
213         "defaults": {
214             "color": {

```

```
215         "mode": "palette-classic"
216     },
217     "custom": {
218         "neutral": 2
219     },
220     "decimals": 1,
221     "displayName": "Tempo total",
222     "fieldMinMax": false,
223     "mappings": [],
224     "thresholds": {
225         "mode": "absolute",
226         "steps": [
227             {
228                 "color": "green",
229                 "value": null
230             },
231             {
232                 "color": "red",
233                 "value": 80
234             }
235         ]
236     },
237     "unit": "m"
238 },
239 "overrides": []
240 },
241 "gridPos": {
242     "h": 12,
243     "w": 4,
244     "x": 12,
245     "y": 0
246 },
247 "id": 5,
248 "options": {
249     "minVizHeight": 0,
250     "minVizWidth": 75,
251     "orientation": "auto",
252     "reduceOptions": {
253         "calcs": [
254             "lastNotNull"
255         ],
256         "fields": "/^Tempo total$/",
```

```
257     "values": true
258   },
259   "showThresholdLabels": false,
260   "showThresholdMarkers": false,
261   "sizing": "auto"
262 },
263 "pluginVersion": "11.4.0",
264 "targets": [
265   {
266     "datasource": {
267       "type": "grafana-postgresql-datasource",
268       "uid": "beqivwgripam8c"
269     },
270     "editorMode": "code",
271     "format": "table",
272     "rawQuery": true,
273     "rawSql": "SELECT \r\n  valor AS \"Tempo total\" \r\nFROM
                \r\n  estufas \r\nWHERE \r\n  topico = '${estufas}'
                AND \r\n  tag = 'tempo_total' AND \r\n  $__timeFilter(
                time) \r\nORDER BY \r\n  time DESC \r\nLIMIT 1 \r\n",
274     "refId": "A",
275     "sql": {
276       "columns": [
277         {
278           "parameters": [],
279           "type": "function"
280         }
281       ],
282       "groupBy": [
283         {
284           "property": {
285             "type": "string"
286           },
287           "type": "groupBy"
288         }
289       ],
290       "limit": 50
291     }
292   }
293 ],
294 "title": "",
295 "transparent": true,
```

```
296     "type": "gauge"
297   },
298   {
299     "datasource": {
300       "type": "grafana-postgresql-datasource",
301       "uid": "beqivwgripam8c"
302     },
303     "fieldConfig": {
304       "defaults": {
305         "color": {
306           "mode": "thresholds"
307         },
308         "displayName": "Tempo passo",
309         "mappings": [],
310         "thresholds": {
311           "mode": "absolute",
312           "steps": [
313             {
314               "color": "green",
315               "value": null
316             },
317             {
318               "color": "red",
319               "value": 80
320             }
321           ]
322         },
323         "unit": "m"
324       },
325       "overrides": []
326     },
327     "gridPos": {
328       "h": 12,
329       "w": 4,
330       "x": 16,
331       "y": 0
332     },
333     "id": 9,
334     "options": {
335       "minVizHeight": 75,
336       "minVizWidth": 75,
337       "orientation": "auto",
```

```
338     "reduceOptions": {
339         "calcs": [
340             "lastNotNull"
341         ],
342         "fields": "/^Tempo passo$/",
343         "values": false
344     },
345     "showThresholdLabels": false,
346     "showThresholdMarkers": true,
347     "sizing": "auto"
348 },
349 "pluginVersion": "11.4.0",
350 "targets": [
351     {
352         "datasource": {
353             "type": "grafana-postgresql-datasource",
354             "uid": "beqivwgripam8c"
355         },
356         "editorMode": "code",
357         "format": "table",
358         "rawQuery": true,
359         "rawSql": "SELECT \r\n  valor AS \"Tempo passo\" \r\nFROM
\r\n  estufas \r\nWHERE \r\n  topico = '${estufas}'
AND \r\n  $__timeFilter(time) AND \r\n  tag = '
tempo_passo' \r\nORDER BY \r\n  time DESC \r\nLIMIT 1",
360         "refId": "A",
361         "sql": {
362             "columns": [
363                 {
364                     "parameters": [],
365                     "type": "function"
366                 }
367             ],
368             "groupBy": [
369                 {
370                     "property": {
371                         "type": "string"
372                     },
373                     "type": "groupBy"
374                 }
375             ],
376             "limit": 50
```

```
377     }
378   }
379 ],
380   "title": "",
381   "transparent": true,
382   "type": "gauge"
383 },
384 {
385   "datasource": {
386     "type": "grafana-postgresql-datasource",
387     "uid": "beqivwgripam8c"
388   },
389   "fieldConfig": {
390     "defaults": {
391       "color": {
392         "mode": "thresholds"
393       },
394       "displayName": "Tempo restante",
395       "mappings": [],
396       "thresholds": {
397         "mode": "absolute",
398         "steps": [
399           {
400             "color": "green",
401             "value": null
402           },
403           {
404             "color": "red",
405             "value": 80
406           }
407         ]
408       },
409       "unit": "m"
410     },
411     "overrides": []
412   },
413   "gridPos": {
414     "h": 12,
415     "w": 4,
416     "x": 20,
417     "y": 0
418   },
419 }
```

```
419     "id": 8,
420     "options": {
421       "minVizHeight": 75,
422       "minVizWidth": 75,
423       "orientation": "auto",
424       "reduceOptions": {
425         "calcs": [
426           "lastNotNull"
427         ],
428         "fields": "/^Tempo restante$/",
429         "values": false
430       },
431       "showThresholdLabels": false,
432       "showThresholdMarkers": true,
433       "sizing": "auto"
434     },
435     "pluginVersion": "11.4.0",
436     "targets": [
437       {
438         "datasource": {
439           "type": "grafana-postgresql-datasource",
440           "uid": "beqivwgripam8c"
441         },
442         "editorMode": "code",
443         "format": "table",
444         "rawQuery": true,
445         "rawSql": "SELECT \r\n  valor AS \"Tempo restante\"\r\n
                    nFROM \r\n  estufas\r\nWHERE\r\n  topico = '${estufas
                    }' AND\r\n  $__timeFilter(time) AND\r\n  tag = '
                    tempo_restante' \r\nORDER BY \r\n  time DESC\r\n
                    nLIMIT 1\r\n",
446         "refId": "A",
447         "sql": {
448           "columns": [
449             {
450               "parameters": [],
451               "type": "function"
452             }
453           ],
454           "groupBy": [
455             {
456               "property": {
```

```
457         "type": "string"
458     },
459     "type": "groupBy"
460 }
461 ],
462 "limit": 50
463 }
464 }
465 ],
466 "title": "",
467 "transparent": true,
468 "type": "gauge"
469 },
470 {
471     "datasource": {
472         "type": "grafana-postgresql-datasource",
473         "uid": "beqivwgripam8c"
474     },
475     "fieldConfig": {
476         "defaults": {
477             "color": {
478                 "fixedColor": "orange",
479                 "mode": "fixed"
480             },
481             "mappings": [],
482             "thresholds": {
483                 "mode": "absolute",
484                 "steps": [
485                     {
486                         "color": "#0d0d0f",
487                         "value": null
488                     },
489                     {
490                         "color": "#EAB839",
491                         "value": 0
492                     }
493                 ]
494             }
495         },
496         "overrides": []
497     },
498     "gridPos": {
```

```
499     "h": 3,
500     "w": 8,
501     "x": 0,
502     "y": 9
503 },
504 "id": 6,
505 "options": {
506     "colorMode": "value",
507     "graphMode": "area",
508     "justifyMode": "auto",
509     "orientation": "auto",
510     "percentChangeColorMode": "standard",
511     "reduceOptions": {
512         "calcs": [
513             "lastNotNull"
514         ],
515         "fields": "/^passo$/",
516         "values": false
517     },
518     "showPercentChange": false,
519     "text": {},
520     "textMode": "value_and_name",
521     "wideLayout": true
522 },
523 "pluginVersion": "11.4.0",
524 "targets": [
525     {
526         "datasource": {
527             "type": "grafana-postgresql-datasource",
528             "uid": "beqivwgripam8c"
529         },
530         "editorMode": "code",
531         "format": "table",
532         "rawQuery": true,
533         "rawSql": "SELECT \r\n  time AS \"1 - Data e Hora\", \r\n
                    valor AS \"passo\" \r\nFROM \r\n  estufas \r\nWHERE \r\n
                    \n  topico = '${estufas}' AND \r\n  $__timeFilter(time
                    ) AND \r\n  tag = 'passo' \r\nORDER BY \r\n  time DESC \r\n
                    \r\nLIMIT 1 \r\n",
534         "refId": "A",
535         "sql": {
536             "columns": [
```

```
537         {
538             "parameters": [],
539             "type": "function"
540         }
541     ],
542     "groupBy": [
543         {
544             "property": {
545                 "type": "string"
546             },
547             "type": "groupBy"
548         }
549     ],
550     "limit": 50
551 }
552 }
553 ],
554 "title": "",
555 "transparent": true,
556 "type": "stat"
557 },
558 {
559     "datasource": {
560         "type": "grafana-postgresql-datasource",
561         "uid": "beqivwgripam8c"
562     },
563     "fieldConfig": {
564         "defaults": {
565             "color": {
566                 "mode": "palette-classic"
567             },
568             "mappings": [
569                 {
570                     "options": {
571                         "desligado": {
572                             "color": "red",
573                             "index": 0
574                         },
575                         "falha": {
576                             "color": "yellow",
577                             "index": 2
578                         },
```

```
579         "ligado": {
580             "color": "green",
581             "index": 1
582         }
583     },
584     "type": "value"
585 }
586 ],
587 "thresholds": {
588     "mode": "absolute",
589     "steps": [
590         {
591             "color": "green",
592             "value": null
593         },
594         {
595             "color": "red",
596             "value": 80
597         }
598     ]
599 }
600 },
601 "overrides": []
602 },
603 "gridPos": {
604     "h": 4,
605     "w": 8,
606     "x": 0,
607     "y": 12
608 },
609 "id": 2,
610 "options": {
611     "colorMode": "value",
612     "graphMode": "area",
613     "justifyMode": "center",
614     "orientation": "auto",
615     "percentChangeColorMode": "standard",
616     "reduceOptions": {
617         "calcs": [
618             "last"
619         ],
620         "fields": "/^ P Alternada$/",
```

```
621         "values": false
622     },
623     "showPercentChange": false,
624     "textMode": "value_and_name",
625     "wideLayout": true
626 },
627 "pluginVersion": "11.4.0",
628 "targets": [
629     {
630         "datasource": {
631             "type": "grafana-postgresql-datasource",
632             "uid": "beqivwgripam8c"
633         },
634         "editorMode": "code",
635         "format": "table",
636         "rawQuery": true,
637         "rawSql": "SELECT \r\n  time AS \"1 - Data e Hora\", \r\n
        valor AS \"P Alternada\" \r\nFROM \r\n  estufas \r\n
        WHERE \r\n  topico = '${estufas}' AND \r\n
        $__timeFilter(time) AND \r\n  tag = 'pa_alternada' \r\n
        ORDER BY \r\n  time DESC \r\nLIMIT 1 \r\n",
638         "refId": "A",
639         "sql": {
640             "columns": [
641                 {
642                     "parameters": [],
643                     "type": "function"
644                 }
645             ],
646             "groupBy": [
647                 {
648                     "property": {
649                         "type": "string"
650                     },
651                     "type": "groupBy"
652                 }
653             ],
654             "limit": 50
655         }
656     }
657 ],
658 "title": "",
```

```
659     "transparent": true,
660     "type": "stat"
661 },
662 {
663     "datasource": {
664         "type": "grafana-postgresql-datasource",
665         "uid": "beqivwgripam8c"
666     },
667     "fieldConfig": {
668         "defaults": {
669             "color": {
670                 "mode": "palette-classic"
671             },
672             "mappings": [
673                 {
674                     "options": {
675                         "desligado": {
676                             "color": "red",
677                             "index": 0
678                         },
679                         "falha": {
680                             "color": "yellow",
681                             "index": 2
682                         },
683                         "ligado": {
684                             "color": "green",
685                             "index": 1
686                         }
687                     },
688                     "type": "value"
689                 }
690             ],
691             "thresholds": {
692                 "mode": "absolute",
693                 "steps": [
694                     {
695                         "color": "green",
696                         "value": null
697                     },
698                     {
699                         "color": "red",
700                         "value": 80
```



```
        estufas\r\nWHERE\r\n    topico = '${estufas}'AND\r\n    $__timeFilter(time) AND\r\n    tag = 'exaustor_da_estufa'\r\nORDER BY \r\n    time DESC\r\nLIMIT 1\r\n",
742     "refId": "A",
743     "sql": {
744         "columns": [
745             {
746                 "parameters": [],
747                 "type": "function"
748             }
749         ],
750         "groupBy": [
751             {
752                 "property": {
753                     "type": "string"
754                 },
755                 "type": "groupBy"
756             }
757         ],
758         "limit": 50
759     }
760 ],
761     "title": "",
762     "transparent": true,
763     "type": "stat"
764 },
765 {
766     "datasource": {
767         "type": "grafana-postgresql-datasource",
768         "uid": "beqivwgripam8c"
769     },
770     "fieldConfig": {
771         "defaults": {
772             "color": {
773                 "mode": "palette-classic"
774             },
775             "mappings": [
776                 {
777                     "options": {
778                         "desligado": {
```

```
780         "color": "red",
781         "index": 0
782     },
783     "falha": {
784         "color": "yellow",
785         "index": 2
786     },
787     "ligado": {
788         "color": "green",
789         "index": 1
790     }
791 },
792 "type": "value"
793 }
794 ],
795 "thresholds": {
796     "mode": "absolute",
797     "steps": [
798         {
799             "color": "green",
800             "value": null
801         },
802         {
803             "color": "red",
804             "value": 80
805         }
806     ]
807 },
808 },
809 "overrides": []
810 },
811 "gridPos": {
812     "h": 4,
813     "w": 8,
814     "x": 16,
815     "y": 12
816 },
817 "id": 4,
818 "options": {
819     "colorMode": "value",
820     "graphMode": "area",
821     "justifyMode": "center",
```

```
822     "orientation": "auto",
823     "percentChangeColorMode": "standard",
824     "reduceOptions": {
825         "calcs": [
826             "lastNotNull"
827         ],
828         "fields": "/^Ventilador Principal$/",
829         "values": false
830     },
831     "showPercentChange": false,
832     "textMode": "value_and_name",
833     "wideLayout": true
834 },
835 "pluginVersion": "11.4.0",
836 "targets": [
837     {
838         "datasource": {
839             "type": "grafana-postgresql-datasource",
840             "uid": "beqivwgripam8c"
841         },
842         "editorMode": "code",
843         "format": "table",
844         "rawQuery": true,
845         "rawSql": "SELECT \r\n    time AS \"1 - Data e Hora\", \r\n
            valor AS \"Ventilador Principal\" \r\nFROM \r\n
            estufas \r\nWHERE \r\n    topico = '${estufas}' AND \r\n
            $__timeFilter(time) AND \r\n    tag = '
            ventilador_principal' \r\nORDER BY \r\n    time DESC \r\n
            nLIMIT 1 \r\n",
846         "refId": "A",
847         "sql": {
848             "columns": [
849                 {
850                     "parameters": [],
851                     "type": "function"
852                 }
853             ],
854             "groupBy": [
855                 {
856                     "property": {
857                         "type": "string"
858                     },
```

```
859         "type": "groupBy"
860     }
861 ],
862     "limit": 50
863 }
864 }
865 ],
866     "title": "",
867     "transparent": true,
868     "type": "stat"
869 },
870 {
871     "datasource": {
872         "type": "grafana-postgresql-datasource",
873         "uid": "beqivwgripam8c"
874     },
875     "fieldConfig": {
876         "defaults": {
877             "color": {
878                 "mode": "palette-classic"
879             },
880             "custom": {
881                 "axisBorderShow": false,
882                 "axisCenteredZero": false,
883                 "axisColorMode": "text",
884                 "axisGridShow": true,
885                 "axisLabel": "",
886                 "axisPlacement": "auto",
887                 "barAlignment": 1,
888                 "barWidthFactor": 0.6,
889                 "drawStyle": "line",
890                 "fillOpacity": 0,
891                 "gradientMode": "none",
892                 "hideFrom": {
893                     "legend": false,
894                     "tooltip": false,
895                     "viz": false
896                 },
897                 "insertNulls": false,
898                 "lineInterpolation": "linear",
899                 "lineStyle": {
900                     "fill": "solid"
```

```
901         },
902         "lineWidth": 2,
903         "pointSize": 2,
904         "scaleDistribution": {
905             "type": "linear"
906         },
907         "showPoints": "never",
908         "spanNulls": true,
909         "stacking": {
910             "group": "A",
911             "mode": "none"
912         },
913         "thresholdsStyle": {
914             "mode": "off"
915         }
916     },
917     "mappings": [],
918     "thresholds": {
919         "mode": "absolute",
920         "steps": [
921             {
922                 "color": "green",
923                 "value": null
924             },
925             {
926                 "color": "red",
927                 "value": 80
928             }
929         ]
930     }
931 },
932 "overrides": [
933     {
934         "matcher": {
935             "id": "byName",
936             "options": "temperatura_estufa"
937         },
938         "properties": []
939     },
940     {
941         "matcher": {
942             "id": "byName",
```

```
943         "options": "temperatura_do_produto"
944     },
945     "properties": []
946 }
947 ]
948 },
949 "gridPos": {
950     "h": 13,
951     "w": 24,
952     "x": 0,
953     "y": 16
954 },
955 "id": 10,
956 "options": {
957     "legend": {
958         "calcs": [],
959         "displayMode": "list",
960         "placement": "bottom",
961         "showLegend": true
962     },
963     "timezone": [
964         "browser"
965     ],
966     "tooltip": {
967         "hideZeros": false,
968         "mode": "single",
969         "sort": "asc"
970     }
971 },
972 "pluginVersion": "11.4.0",
973 "targets": [
974     {
975         "datasource": {
976             "type": "grafana-postgresql-datasource",
977             "uid": "beqivwgripam8c"
978         },
979         "editorMode": "code",
980         "format": "table",
981         "rawQuery": true,
982         "rawSql": "SELECT\r\n  time,\r\n  MAX(CASE \r\n\n    WHEN tag = 'temperatura_do_produto' AND valor ~ '^\\d\n\n    +((\\.\\d+)?$'\\r\n\n    THEN CAST(valor AS DOUBLE
```

```

PRECISION)\r\n      END) AS temperatura_do_produto,\r\n
\r\n MAX(CASE \r\n      WHEN tag = '
temperatura_estufa' AND valor ~ '^\\d+(\\.\\d+)?$'\r\n
\r\n      THEN CAST(valor AS DOUBLE PRECISION)\r\n
      END) AS temperatura_estufa\r\nFROM\r\n  estufas\r\nWHERE\r\n  topico = '${estufas}' AND\r\n  $__timeFilter(time) AND\r\n  tag IN ('
temperatura_do_produto', 'temperatura_estufa')\r\n
\r\nGROUP BY\r\n  topico, time\r\nORDER BY\r\n  time\r\n
",
983 "refId": "A",
984 "sql": {
985   "columns": [
986     {
987       "parameters": [],
988       "type": "function"
989     }
990   ],
991   "groupBy": [
992     {
993       "property": {
994         "type": "string"
995       },
996       "type": "groupBy"
997     }
998   ],
999   "limit": 50
1000 }
1001 }
1002 ],
1003 "title": "",
1004 "transparent": true,
1005 "type": "timeseries"
1006 },
1007 {
1008   "datasource": {
1009     "type": "grafana-postgresql-datasource",
1010     "uid": "beqivwgripam8c"
1011   },
1012   "fieldConfig": {
1013     "defaults": {
1014       "color": {

```

```
1015         "fixedColor": "yellow",
1016         "mode": "shades"
1017     },
1018     "mappings": [],
1019     "thresholds": {
1020         "mode": "absolute",
1021         "steps": [
1022             {
1023                 "color": "green"
1024             },
1025             {
1026                 "color": "red",
1027                 "value": 80
1028             }
1029         ]
1030     },
1031     "unit": "celsius"
1032 },
1033 "overrides": []
1034 },
1035 "gridPos": {
1036     "h": 3,
1037     "w": 6,
1038     "x": 0,
1039     "y": 29
1040 },
1041 "id": 15,
1042 "options": {
1043     "colorMode": "value",
1044     "graphMode": "none",
1045     "justifyMode": "auto",
1046     "orientation": "auto",
1047     "percentChangeColorMode": "standard",
1048     "reduceOptions": {
1049         "calcs": [
1050             "lastNotNull"
1051         ],
1052         "fields": "/^Temperatura da Estufa$/",
1053         "values": false
1054     },
1055     "showPercentChange": false,
1056     "textMode": "auto",
```

```
1057     "wideLayout": true
1058   },
1059   "pluginVersion": "11.4.0",
1060   "targets": [
1061     {
1062       "datasource": {
1063         "type": "grafana-postgresql-datasource",
1064         "uid": "beqivwgripam8c"
1065       },
1066       "editorMode": "code",
1067       "format": "table",
1068       "rawQuery": true,
1069       "rawSql": "SELECT \r\n  time AS \"1 - Data e Hora\", \r\n
                valor AS \"Temperatura da Estufa\" \r\nFROM \r\n
                estufas \r\nWHERE \r\n  topico = '${estufas}' \r\n
                AND $__timeFilter(time) \r\n  AND tag = '
                temperatura_estufa' \r\n  \r\nORDER BY \r\n  time
                DESC \r\nLIMIT 1; \r\n",
1070       "refId": "A",
1071       "sql": {
1072         "columns": [
1073           {
1074             "parameters": [],
1075             "type": "function"
1076           }
1077         ],
1078         "groupBy": [
1079           {
1080             "property": {
1081               "type": "string"
1082             },
1083             "type": "groupBy"
1084           }
1085         ],
1086         "limit": 50
1087       }
1088     }
1089   ],
1090   "title": "",
1091   "transparent": true,
1092   "type": "stat"
1093 },
```

```
1094 {
1095   "datasource": {
1096     "type": "grafana-postgresql-datasource",
1097     "uid": "beqivwgripam8c"
1098   },
1099   "fieldConfig": {
1100     "defaults": {
1101       "color": {
1102         "fixedColor": "yellow",
1103         "mode": "shades"
1104       },
1105       "displayName": "M nima",
1106       "fieldMinMax": true,
1107       "mappings": [],
1108       "thresholds": {
1109         "mode": "absolute",
1110         "steps": [
1111           {
1112             "color": "green"
1113           },
1114           {
1115             "color": "red",
1116             "value": 80
1117           }
1118         ]
1119       },
1120       "unit": "celsius"
1121     },
1122     "overrides": []
1123   },
1124   "gridPos": {
1125     "h": 3,
1126     "w": 6,
1127     "x": 6,
1128     "y": 29
1129   },
1130   "id": 11,
1131   "options": {
1132     "colorMode": "value",
1133     "graphMode": "none",
1134     "justifyMode": "auto",
1135     "orientation": "auto",
```

```
1136     "percentChangeColorMode": "standard",
1137     "reduceOptions": {
1138         "calcs": [
1139             "min"
1140         ],
1141         "fields": "",
1142         "values": false
1143     },
1144     "showPercentChange": false,
1145     "textMode": "value_and_name",
1146     "wideLayout": true
1147 },
1148 "pluginVersion": "11.4.0",
1149 "targets": [
1150     {
1151         "datasource": {
1152             "type": "grafana-postgresql-datasource",
1153             "uid": "beqivwgripam8c"
1154         },
1155         "editorMode": "code",
1156         "format": "table",
1157         "rawQuery": true,
1158         "rawSql": "SELECT\r\n  time,\r\n  CAST(valor AS DOUBLE
                    PRECISION) AS temperatura_minima_estufa\r\nFROM \r\n
                    estufas\r\nWHERE \r\n  topico = '${estufas}' AND\r\n
                    $__timeFilter(time) AND\r\n  tag = '
                    temperatura_estufa'\r\n",
1159         "refId": "A",
1160         "sql": {
1161             "columns": [
1162                 {
1163                     "parameters": [],
1164                     "type": "function"
1165                 }
1166             ],
1167             "groupBy": [
1168                 {
1169                     "property": {
1170                         "type": "string"
1171                     },
1172                     "type": "groupBy"
1173                 }
1174             ]
1175         }
1176     }
1177 ]
```

```
1174         ],
1175         "limit": 50
1176     }
1177 }
1178 ],
1179 "title": "",
1180 "transparent": true,
1181 "type": "stat"
1182 },
1183 {
1184     "datasource": {
1185         "type": "grafana-postgresql-datasource",
1186         "uid": "beqivwgripam8c"
1187     },
1188     "fieldConfig": {
1189         "defaults": {
1190             "color": {
1191                 "fixedColor": "yellow",
1192                 "mode": "shades"
1193             },
1194             "displayName": "M dia",
1195             "mappings": [],
1196             "thresholds": {
1197                 "mode": "absolute",
1198                 "steps": [
1199                     {
1200                         "color": "green"
1201                     },
1202                     {
1203                         "color": "red",
1204                         "value": 80
1205                     }
1206                 ]
1207             },
1208             "unit": "celsius"
1209         },
1210         "overrides": []
1211     },
1212     "gridPos": {
1213         "h": 3,
1214         "w": 6,
1215         "x": 12,
```

```
1216     "y": 29
1217   },
1218   "id": 12,
1219   "options": {
1220     "colorMode": "value",
1221     "graphMode": "none",
1222     "justifyMode": "auto",
1223     "orientation": "auto",
1224     "percentChangeColorMode": "standard",
1225     "reduceOptions": {
1226       "calcs": [
1227         "lastNotNull"
1228       ],
1229       "fields": "",
1230       "values": false
1231     },
1232     "showPercentChange": false,
1233     "textMode": "value_and_name",
1234     "wideLayout": true
1235   },
1236   "pluginVersion": "11.4.0",
1237   "targets": [
1238     {
1239       "datasource": {
1240         "type": "grafana-postgresql-datasource",
1241         "uid": "beqivwgripam8c"
1242       },
1243       "editorMode": "code",
1244       "format": "table",
1245       "hide": false,
1246       "rawQuery": true,
1247       "rawSql": "SELECT \r\n  time,\r\n  CAST(valor AS DOUBLE
        PRECISION) AS temperatura_media_do_produto\r\nFROM \r
        \n  estufas\r\nWHERE \r\n  topico = '${estufas}' AND\
        \r\n  $__timeFilter(time) AND\r\n  tag = '
        temperatura_estufa'\r\n\r\n",
1248       "refId": "A",
1249       "sql": {
1250         "columns": [
1251           {
1252             "parameters": [],
1253             "type": "function"
```

```
1254         }
1255     ],
1256     "groupBy": [
1257         {
1258             "property": {
1259                 "type": "string"
1260             },
1261             "type": "groupBy"
1262         }
1263     ],
1264     "limit": 50
1265 }
1266 }
1267 ],
1268 "title": "",
1269 "transparent": true,
1270 "type": "stat"
1271 },
1272 {
1273     "datasource": {
1274         "type": "grafana-postgresql-datasource",
1275         "uid": "beqivwgripam8c"
1276     },
1277     "fieldConfig": {
1278         "defaults": {
1279             "color": {
1280                 "fixedColor": "yellow",
1281                 "mode": "shades"
1282             },
1283             "displayName": "M xima",
1284             "fieldMinMax": true,
1285             "mappings": [],
1286             "thresholds": {
1287                 "mode": "absolute",
1288                 "steps": [
1289                     {
1290                         "color": "green"
1291                     },
1292                     {
1293                         "color": "red",
1294                         "value": 80
1295                     }
1296                 ]
1297             }
1298         }
1299     }
1300 }
```

```
1296         ]
1297     },
1298     "unit": "celsius"
1299 },
1300     "overrides": []
1301 },
1302     "gridPos": {
1303         "h": 3,
1304         "w": 6,
1305         "x": 18,
1306         "y": 29
1307     },
1308     "id": 13,
1309     "options": {
1310         "colorMode": "value",
1311         "graphMode": "none",
1312         "justifyMode": "auto",
1313         "orientation": "auto",
1314         "percentChangeColorMode": "standard",
1315         "reduceOptions": {
1316             "calcs": [
1317                 "max"
1318             ],
1319             "fields": "",
1320             "values": false
1321         },
1322         "showPercentChange": false,
1323         "textMode": "value_and_name",
1324         "wideLayout": true
1325     },
1326     "pluginVersion": "11.4.0",
1327     "targets": [
1328         {
1329             "datasource": {
1330                 "type": "grafana-postgresql-datasource",
1331                 "uid": "beqivwgripam8c"
1332             },
1333             "editorMode": "code",
1334             "format": "table",
1335             "rawQuery": true,
1336             "rawSql": "SELECT \r\n  time,\r\n  CAST(valor AS DOUBLE
                        PRECISION) AS temperatura_media_do_produto\r\nFROM \r"
```



```
1376         {
1377             "color": "green"
1378         },
1379         {
1380             "color": "red",
1381             "value": 80
1382         }
1383     ],
1384     },
1385     "unit": "celsius"
1386 },
1387 "overrides": []
1388 },
1389 "gridPos": {
1390     "h": 3,
1391     "w": 6,
1392     "x": 0,
1393     "y": 32
1394 },
1395 "id": 14,
1396 "options": {
1397     "colorMode": "value",
1398     "graphMode": "area",
1399     "justifyMode": "auto",
1400     "orientation": "auto",
1401     "percentChangeColorMode": "standard",
1402     "reduceOptions": {
1403         "calcs": [
1404             "lastNotNull"
1405         ],
1406         "fields": "/^Temperatura do produto$/",
1407         "values": false
1408     },
1409     "showPercentChange": false,
1410     "textMode": "auto",
1411     "wideLayout": true
1412 },
1413 "pluginVersion": "11.4.0",
1414 "targets": [
1415     {
1416         "datasource": {
1417             "type": "grafana-postgresql-datasource",
```

```

1418         "uid": "beqivwgripam8c"
1419     },
1420     "editorMode": "code",
1421     "format": "table",
1422     "rawQuery": true,
1423     "rawSql": "SELECT\r\n    time AS \"1 - Data e Hora\", \r\n
        valor AS \"Temperatura do produto\" \r\n
        FROM \r\n
        estufas \r\n
        WHERE \r\n
        topico = '${estufas}' AND \r\n
        $__timeFilter(time) AND \r\n
        tag = '
        temperatura_do_produto' \r\n
        ORDER BY \r\n
        time
        DESC \r\n
        LIMIT 1;",
1424     "refId": "A",
1425     "sql": {
1426         "columns": [
1427             {
1428                 "parameters": [],
1429                 "type": "function"
1430             }
1431         ],
1432         "groupBy": [
1433             {
1434                 "property": {
1435                     "type": "string"
1436                 },
1437                 "type": "groupBy"
1438             }
1439         ],
1440         "limit": 50
1441     }
1442 },
1443 ],
1444 "title": "",
1445 "transparent": true,
1446 "type": "stat"
1447 },
1448 {
1449     "datasource": {
1450         "type": "grafana-postgresql-datasource",
1451         "uid": "beqivwgripam8c"
1452     },
1453     "fieldConfig": {
1454         "defaults": {

```

```
1455     "color": {
1456         "fixedColor": "green",
1457         "mode": "shades"
1458     },
1459     "displayName": "M nima",
1460     "mappings": [],
1461     "thresholds": {
1462         "mode": "absolute",
1463         "steps": [
1464             {
1465                 "color": "green"
1466             },
1467             {
1468                 "color": "red",
1469                 "value": 80
1470             }
1471         ]
1472     },
1473     "unit": "celsius"
1474 },
1475 "overrides": []
1476 },
1477 "gridPos": {
1478     "h": 3,
1479     "w": 6,
1480     "x": 6,
1481     "y": 32
1482 },
1483 "id": 16,
1484 "options": {
1485     "colorMode": "value",
1486     "graphMode": "none",
1487     "justifyMode": "auto",
1488     "orientation": "auto",
1489     "percentChangeColorMode": "standard",
1490     "reduceOptions": {
1491         "calcs": [
1492             "min"
1493         ],
1494         "fields": "",
1495         "values": false
1496     },
```

```
1497     "showPercentChange": false,
1498     "textMode": "value_and_name",
1499     "wideLayout": true
1500 },
1501 "pluginVersion": "11.4.0",
1502 "targets": [
1503     {
1504         "datasource": {
1505             "type": "grafana-postgresql-datasource",
1506             "uid": "beqivwgripam8c"
1507         },
1508         "editorMode": "code",
1509         "format": "table",
1510         "rawQuery": true,
1511         "rawSql": "SELECT \r\n  time,\r\n  CAST(valor AS DOUBLE
                    PRECISION) AS temperatura_minima_do_produto\r\nFROM \r\n
                    \r\n  estufas\r\nWHERE \r\n  topico = '${estufas}' AND
                    \r\n  $__timeFilter(time) AND\r\n  tag = '
                    temperatura_do_produto'\r\n",
1512         "refId": "A",
1513         "sql": {
1514             "columns": [
1515                 {
1516                     "parameters": [],
1517                     "type": "function"
1518                 }
1519             ],
1520             "groupBy": [
1521                 {
1522                     "property": {
1523                         "type": "string"
1524                     },
1525                     "type": "groupBy"
1526                 }
1527             ],
1528             "limit": 50
1529         }
1530     }
1531 ],
1532 "title": "",
1533 "transparent": true,
1534 "type": "stat"
```

```
1535     },
1536     {
1537         "datasource": {
1538             "type": "grafana-postgresql-datasource",
1539             "uid": "beqivwgripam8c"
1540         },
1541         "fieldConfig": {
1542             "defaults": {
1543                 "color": {
1544                     "fixedColor": "green",
1545                     "mode": "shades"
1546                 },
1547                 "displayName": "M dia",
1548                 "mappings": [],
1549                 "thresholds": {
1550                     "mode": "absolute",
1551                     "steps": [
1552                         {
1553                             "color": "green"
1554                         },
1555                         {
1556                             "color": "red",
1557                             "value": 80
1558                         }
1559                     ]
1560                 },
1561                 "unit": "celsius"
1562             },
1563             "overrides": []
1564         },
1565         "gridPos": {
1566             "h": 3,
1567             "w": 6,
1568             "x": 12,
1569             "y": 32
1570         },
1571         "id": 19,
1572         "options": {
1573             "colorMode": "value",
1574             "graphMode": "none",
1575             "justifyMode": "auto",
1576             "orientation": "auto",
```

```
1577     "percentChangeColorMode": "standard",
1578     "reduceOptions": {
1579       "calcs": [
1580         "lastNotNull"
1581       ],
1582       "fields": "",
1583       "values": false
1584     },
1585     "showPercentChange": false,
1586     "textMode": "value_and_name",
1587     "wideLayout": true
1588   },
1589   "pluginVersion": "11.4.0",
1590   "targets": [
1591     {
1592       "datasource": {
1593         "type": "grafana-postgresql-datasource",
1594         "uid": "beqivwgripam8c"
1595       },
1596       "editorMode": "code",
1597       "format": "table",
1598       "hide": false,
1599       "rawQuery": true,
1600       "rawSql": "SELECT \r\n  time,\r\n  CAST(valor AS DOUBLE
        PRECISION) AS temperatura_media_do_produto\r\nFROM \r
        \n  estufas\r\nWHERE \r\n  topico = '${estufas}' AND\r
        \r\n  $__timeFilter(time) AND\r\n  tag = '
        temperatura_do_produto'\r\n\r\n",
1601       "refId": "A",
1602       "sql": {
1603         "columns": [
1604           {
1605             "parameters": [],
1606             "type": "function"
1607           }
1608         ],
1609         "groupBy": [
1610           {
1611             "property": {
1612               "type": "string"
1613             },
1614             "type": "groupBy"
```

```
1615         }
1616     ],
1617     "limit": 50
1618 }
1619 }
1620 ],
1621 "title": "",
1622 "transparent": true,
1623 "type": "stat"
1624 },
1625 {
1626     "datasource": {
1627         "type": "grafana-postgresql-datasource",
1628         "uid": "beqivwgripam8c"
1629     },
1630     "fieldConfig": {
1631         "defaults": {
1632             "color": {
1633                 "fixedColor": "green",
1634                 "mode": "shades"
1635             },
1636             "displayName": "M xima",
1637             "mappings": [],
1638             "thresholds": {
1639                 "mode": "absolute",
1640                 "steps": [
1641                     {
1642                         "color": "green"
1643                     },
1644                     {
1645                         "color": "red",
1646                         "value": 80
1647                     }
1648                 ]
1649             },
1650             "unit": "celsius"
1651         },
1652         "overrides": []
1653     },
1654     "gridPos": {
1655         "h": 3,
1656         "w": 6,
```

```
1657     "x": 18,
1658     "y": 32
1659 },
1660 "id": 18,
1661 "options": {
1662     "colorMode": "value",
1663     "graphMode": "none",
1664     "justifyMode": "auto",
1665     "orientation": "auto",
1666     "percentChangeColorMode": "standard",
1667     "reduceOptions": {
1668         "calcs": [
1669             "max"
1670         ],
1671         "fields": "",
1672         "values": false
1673     },
1674     "showPercentChange": false,
1675     "textMode": "value_and_name",
1676     "wideLayout": true
1677 },
1678 "pluginVersion": "11.4.0",
1679 "targets": [
1680     {
1681         "datasource": {
1682             "type": "grafana-postgresql-datasource",
1683             "uid": "beqivwgripam8c"
1684         },
1685         "editorMode": "code",
1686         "format": "table",
1687         "rawQuery": true,
1688         "rawSql": "SELECT \r\n  time,\r\n  CAST(valor AS DOUBLE
                    PRECISION) AS temperatura_mmaxima_do_produto\r\nFROM
                    \r\n  estufas\r\nWHERE \r\n  topico = '${estufas}'
                    AND\r\n  $__timeFilter(time) AND\r\n  tag = '
                    temperatura_do_produto' \r\n",
1689         "refId": "A",
1690         "sql": {
1691             "columns": [
1692                 {
1693                     "parameters": [],
1694                     "type": "function"
```

```
1695         }
1696     ],
1697     "groupBy": [
1698         {
1699             "property": {
1700                 "type": "string"
1701             },
1702             "type": "groupBy"
1703         }
1704     ],
1705     "limit": 50
1706 }
1707 }
1708 ],
1709 "title": "",
1710 "transparent": true,
1711 "type": "stat"
1712 }
1713 ],
1714 "preload": false,
1715 "refresh": "30s",
1716 "schemaVersion": 40,
1717 "tags": [],
1718 "templating": {
1719     "list": [
1720         {
1721             "current": {
1722                 "text": "estufas04",
1723                 "value": "estufas04"
1724             },
1725             "datasource": {
1726                 "type": "grafana-postgresql-datasource",
1727                 "uid": "beqivwgripam8c"
1728             },
1729             "definition": "SELECT DISTINCT topico FROM estufas ",
1730             "description": "Escolhas a estufa que deseja vizualizar os
1731                             dados",
1732             "label": "estufas",
1733             "name": "estufas",
1734             "options": [],
1735             "query": "SELECT DISTINCT topico FROM estufas ",
1736             "refresh": 1,
```

```
1736         "regex": "",
1737         "type": "query"
1738     }
1739 ]
1740 },
1741 "time": {
1742     "from": "now-24h",
1743     "to": "now"
1744 },
1745 "timepicker": {},
1746 "timezone": "browser",
1747 "title": "Estufas do Bacon",
1748 "uid": "dac6b97a-1476-410c-a14b-6b4b414c8b6c",
1749 "version": 16,
1750 "weekStart": ""
1751 }
```

7 Jason NodeRed

Listing 7.1 – Configuração Flow NodeRed estufa01 JSON

```
1  [
2    {
3      "id": "f6f2187d.f17ca8",
4      "type": "tab",
5      "label": "Estufas do Bacon",
6      "disabled": false,
7      "info": ""
8    },
9    {
10     "id": "ea546fe6ccfb3c33",
11     "type": "mqtt in",
12     "z": "f6f2187d.f17ca8",
13     "name": "",
14     "topic": "estufas01",
15     "qos": "1",
16     "datatype": "auto-detect",
17     "broker": "2588a161d52b99a8",
18     "nl": false,
19     "rap": true,
20     "rh": 0,
21     "inputs": 0,
22     "x": 160,
23     "y": 60,
24     "wires": [
25       [
26         "5529ca1fe0b3db24"
27       ]
28     ]
29   },
30   {
31     "id": "a852c8ae0175c1a7",
32     "type": "debug",
33     "z": "f6f2187d.f17ca8",
34     "name": "debug 1",
35     "active": false,
36     "tosidebar": true,
37     "console": false,
```

```

38     "tostatus": false,
39     "complete": "payload",
40     "targetType": "msg",
41     "statusVal": "",
42     "statusType": "auto",
43     "x": 760,
44     "y": 40,
45     "wires": []
46 },
47 {
48     "id": "5529ca1fe0b3db24",
49     "type": "function",
50     "z": "f6f2187d.f17ca8",
51     "name": "Tratamento",
52     "func": "// Função para retornar o estado em texto\
nfunction determinarEstadoTexto(componente) {\n    if (
componente.falha) return \"falha\";\n    if (componente
.ligado) return \"ligado\";\n    return \"desligado\";\n
}\n\n// Verifica se msg.payload.values é lido\nif
(!msg.payload || !Array.isArray(msg.payload.values)) {\n
    node.warn(\"Erro: msg.payload.values inválido.\")
;\n    return { payload: { error: \"Dados de entrada
inválidos.\" } };\n}\n\nlet dados = msg.payload.values
;\nlet resultado = {};\nlet tempoTotal = 0;\n\n//
Variáveis de estado dos componentes\nlet paExaustor =
{ falha: false, ligado: false };\nlet
ventiladorPrincipal = { falha: false, ligado: false };\n
let exaustorEstufa = { falha: false, ligado: false };\n
\n// Processa os dados\nfor (let item of dados) {\n
    let id = item.id;\n    let valor = item.v;\n\n    if
(!id || valor === undefined) continue;\n\n    //
Somatório das tags N11_1 a N11_20\n    let matchN11 =
id.match(/^ESTUFAS_BACON\\.FISCHER01\\.N11_(\\d+)$/);\n
    if (matchN11) {\n        let num = parseInt(
matchN11[1]);\n        if (num >= 1 && num <= 20) {\n
            tempoTotal += valor;\n        }\n
continue;\n    }\n\n    // Estado dos dispositivos\n
    switch (id) {\n        case \"ESTUFAS_BACON.
FISCHER01.Pa_alternada FALHA\":\n            paExaustor
.falha = valor === true;\n            continue;\n
        case \"ESTUFAS_BACON.FISCHER01.Pa_alternada\":\n
            paExaustor.ligado = valor === true;\n

```

```

        continue;\n        case \"ESTUFAS_BACON.
FISCHER01.ventilador principal FALHA\":\n
ventiladorPrincipal.falha = valor === true;\n
        continue;\n        case \"ESTUFAS_BACON.
FISCHER01.ventilador principal\":\n
ventiladorPrincipal.ligado = valor === true;\n
        continue;\n        case \"ESTUFAS_BACON.
FISCHER01.exaustor da estufa FALHA\":\n
exaustorEstufa.falha = valor === true;\n
continue;\n        case \"ESTUFAS_BACON.FISCHER01.
exaustor da estufa\":\n                exaustorEstufa.
ligado = valor === true;\n                continue;\n
        case \"ESTUFAS_BACON.FISCHER01.exaustor estufa
COMPARAR\":\n                continue; // ignorar\n    }\n\n
    // Campos desejados (em letras min sculas e com _
    no lugar de espa o)\n    let chave = id.split(\".\").
pop().toLowerCase().replace(/\\s+/g, \"_\").normalize('
NFD').replace(/[\\u0300-\\u036f]/g, \"\");\n\n    if (
chave === \"tempo_passo\" || chave === \"tempo_restante
\") {\n        valor = valor / 60; // converte para
minutos\n    }\n\n    // Adiciona ao resultado apenas
se for uma das chaves desejadas\n    if ([\n        \"
numero_receita\", \n        \"passo\", \n        \"
setpoint\", \n        \"temperatura_do_produto\", \n
        \"temperatura_estufa\", \n        \"tempo_passo
\", \n        \"tempo_restante\"\n    ].includes(chave))
    {\n        resultado[chave] = valor;\n    }\n\n\n//
Adiciona os campos calculados finais\nresultado[\"
tempo_total\"] = tempoTotal;\nresultado[\"pa_alternada
\"] = determinarEstadoTexto(paExaustor);\nresultado[\"
ventilador_principal\"] = determinarEstadoTexto(
ventiladorPrincipal);\nresultado[\"exaustor_da_estufa
\"] = determinarEstadoTexto(exaustorEstufa);\n\n\n//
Retorno final apenas com os campos desejados\nreturn {\n
    n    payload: resultado, \n    topic: msg.topic\n};\n\",
53     \"outputs\": 1,
54     \"timeout\": 0,
55     \"noerr\": 0,
56     \"initialize\": \"\",
57     \"finalize\": \"\",
58     \"libs\": [],
59     \"x\": 330,

```

```
60         "y": 60,  
61         "wires": [  
62             [  
63                 "320b4e6e4f9f0aab",  
64                 "a852c8ae0175c1a7"  
65             ]  
66         ]  
67     },  
68     {  
69         "id": "68ae6e8101b776f7",  
70         "type": "postgresql",  
71         "z": "f6f2187d.f17ca8",  
72         "name": "postgresSQL",  
73         "query": "INSERT INTO estufas (time, topico, tag, valor)\  
        nVALUES (NOW(), '{{msg.topic}}', '{{msg.tag}}', '{{  
        msg.payload}}')";\n",  
74         "postgresSQLConfig": "6236a9773eca422d",  
75         "split": false,  
76         "rowsPerMsg": 1,  
77         "outputs": 1,  
78         "x": 770,  
79         "y": 100,  
80         "wires": [  
81             []  
82         ]  
83     }  
84 ]
```