INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DE GOIÁS
CAMPUS ITUMBIARA
BACHARELADO EM ENGENHARIA ELÉTRICA

Breno Henrique Marques Jorge

Desenvolvimento de uma Plataforma IoT para Monitoramento de Dióxido de Carbono e Amônia

ITUMBIARA
2026

INSTITUTO FEDERAL
Goiás

MINISTÉRIO DA EDUCAÇÃO
SECRETARIA DE EDUCAÇÃO PROFISSIONAL E TECNOLÓGICA
INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
PRÓ-REITORIA DE PESQUISA E PÓS-GRADUAÇÃO
SISTEMA INTEGRADO DE BIBLIOTECAS

TERMO DE AUTORIZAÇÃO PARA DISPONIBILIZAÇÃO NO REPOSITÓRIO DIGITAL DO IFG - ReDi IFG

Com base no disposto na Lei Federal nº 9.610/98, AUTORIZO o Instituto Federal de Educação, Ciência e Tecnologia de Goiás, a disponibilizar gratuitamente o documento no Repositório Digital (ReDi IFG), sem ressarcimento de direitos autorais, conforme permissão assinada abaixo, em formato digital para fins de leitura, download e impressão, a título de divulgação da produção técnico-científica no IFG.

Identificação da Produção Técnico-Científica

[ ] Tese
[ ] Dissertação
[ ] Monografia – Especialização
[X] TCC - Graduação
[ ] Produto Técnico e Educacional - Tipo: _____
[ ] Artigo Científico
[ ] Capítulo de Livro
[ ] Livro
[ ] Trabalho Apresentado em Evento

Nome Completo do Autor: Breno Henrique Marques Jorge
Matrícula: 20202040070067
Título do Trabalho: Desenvolvimento de uma Plataforma IoT para Monitoramento de Dióxido de Carbono e Amônia

Autorização - Marque uma das opções

1. (X) Autorizo disponibilizar meu trabalho no Repositório Digital do IFG (acesso aberto);
2. ( ) Autorizo disponibilizar meu trabalho no Repositório Digital do IFG somente após a data ___/___/_____ (Embargo);
3. ( ) Não autorizo disponibilizar meu trabalho no Repositório Digital do IFG (acesso restrito).

Ao indicar a opção **2 ou 3**, marque a justificativa:

( ) O documento está sujeito a registro de patente.
( ) O documento pode vir a ser publicado como livro, capítulo de livro ou artigo.
( ) Outra justificativa: _____

DECLARAÇÃO DE DISTRIBUIÇÃO NÃO-EXCLUSIVA

O/A referido/a autor/a declara que:

i. o documento é seu trabalho original, detém os direitos autorais da produção técnico-científica e não infringe os direitos de qualquer outra pessoa ou entidade;
ii. obteve autorização de quaisquer materiais inclusos no documento do qual não detém os direitos de autor/a, para conceder ao Instituto Federal de Educação, Ciência e Tecnologia de Goiás os direitos requeridos e que este material cujos direitos autorais são de terceiros, estão claramente identificados e reconhecidos no texto ou conteúdo do documento entregue;
iii. cumpriu quaisquer obrigações exigidas por contrato ou acordo, caso o documento entregue seja baseado em trabalho financiado ou apoiado por outra instituição que não o Instituto Federal de Educação, Ciência e Tecnologia de Goiás.

Itumbiara-GO, 15/07/2026.
Local Data

Breno Henrique Marques Jorge
Assinatura do Autor e/ou Detentor dos Direitos Autorais

Breno Henrique Marques Jorge

Desenvolvimento de uma Plataforma IoT para Monitoramento de Dióxido de Carbono e Amônia

Trabalho de conclusão de curso apresentado ao Instituto Federal De Educação, Ciência e Tecnologia de Goiás, como parte dos requisitos para a obtenção do título de Bacharel em Engenharia Elétrica.

Orientador: Prof. Dr. Eric Nery Chaves

Coorientador: Prof. Dr. Giovani Aud Lourenço

ITUMBIARA

2026

Dados Internacionais de Catalogação na Publicação (CIP)

J82d Jorge, Breno Henrique Marques
Desenvolvimento de uma plataforma IoT para monitoramento de dióxido de carbono e amônia. / Breno Henrique Marques Jorge. – Itumbiara, 2026.
120 p.: il.

Trabalho de conclusão do curso (Graduação em Engenharia Elétrica) - Instituto Federal de Educação, Ciência e Tecnologia de Goiás, Câmpus Itumbiara, 2026.

Orientador: Prof. Dr. Eric Nery Chaves.

1. Internet das coisas. 2. Monitoramento ambiental. 3. Dióxido de carbono. I. Título. II. Chaves, Eric Nery. III. Instituto Federal de Educação, Ciência e Tecnologia de Goiás.

CDD 629.8

Instituto Federal de Educação, Ciência e Tecnologia de Goiás
Biblioteca Maria Gabriela Pacheco Pardey / Câmpus Itumbiara
Bibliotecário: Rosiane Gonçalves de Lima CRB1/1684

MINISTÉRIO DA EDUCAÇÃO
SECRETARIA DE EDUCAÇÃO PROFISSIONAL E TECNOLÓGICA
INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DE GOIÁS
CÂMPUS ITUMBIARA

Termo de Aprovação

Breno Henrique Marques Jorge

Desenvolvimento de uma Plataforma IoT para Monitoramento de Dióxido de Carbono e Amônia

Trabalho de Conclusão de Curso apresentado à Coordenação do Curso de Engenharia Elétrica do Instituto Federal de Educação, Ciência e Tecnologia de Goiás – Campus Itumbiara, como parte dos requisitos para a obtenção do título de Bacharel em Engenharia Elétrica.

Trabalho de Conclusão de Curso defendido e aprovado, em 14 de julho de 2026, pela banca examinadora constituída pelos seguintes membros:

BANCA EXAMINADORA

Prof. Dr. Eric Nery Chaves - Orientador
Instituto Federal de Goiás, Campus Itumbiara
(Assinado Eletronicamente).

Prof. Dr. Giovani Aud Lourenço - Membro interno
Instituto Federal de Goiás, Campus Itumbiara
(Assinado Eletronicamente).

Prof. Dr. Victor Régis Bernadeli - Membro interno
Instituto Federal de Goiás, Campus Itumbiara
(Assinado Eletronicamente).

Itumbiara – GO
2026

Documento assinado eletronicamente por:

- **Giovani Aud Lourenco, PROFESSOR ENS BASICO TECN TECNOLOGICO**, em 14/07/2026 11:29:34.
- **Victor Regis Bernardeli, CHEFE - CD4 - ITU-DAA**, em 14/07/2026 08:38:34.
- **Eric Nery Chaves, PROFESSOR ENS BASICO TECN TECNOLOGICO**, em 13/07/2026 10:50:25.

Este documento foi emitido pelo SUAP em 10/07/2026. Para comprovar sua autenticidade, faça a leitura do QRCode ao lado ou acesse https://suap.ifg.edu.br/autenticar-documento/ e forneça os dados abaixo:

Código Verificador: 798845
Código de Autenticação: b6b23f82f6

DEDICATÓRIA

Dedico este trabalho para minha avó materna, Mariana. Que foi o pilar que tornou toda minha graduação possível.

AGRADECIMENTOS

Agradeço a todos meus familiares pelo apoio durante toda a graduação.

Agradeço a meus orientadores, Eric Nery Chaves e Giovani Aud Lourenço por toda a ajuda e materiais fornecidos para a confecção deste trabalho.

Agradeço a todos docentes da área de química que ajudaram de forma direta ou indiretamente no trabalho.

“Uma vez que você tenha experimentado voar, você caminhará para sempre na terra com seus olhos voltados para o céu, pois lá você esteve e para lá você sempre desejará retornar."

Leonardo Da Vinci

RESUMO

Este trabalho apresenta o desenvolvimento, a implementação e a validação de uma plataforma baseada na Internet das Coisas (IoT) para o monitoramento contínuo e em tempo real das concentrações de dióxido de carbono (CO_2) e amônia (NH_3), com potencial aplicação em sistemas de agricultura vertical em contêineres. A arquitetura desenvolvida utiliza o microcontrolador ESP32 como nó de aquisição e transmissão de dados e o Raspberry Pi 3B como servidor local, responsável pelo gerenciamento das informações por meio do protocolo MQTT. A ferramenta Node-RED foi empregada para o processamento dos dados, desenvolvimento de dashboards para visualização em tempo real e armazenamento local do histórico de medições em arquivos CSV. Para validar o sistema, foi desenvolvida uma metodologia experimental baseada na liberação controlada de amônia a partir de soluções de hidróxido de amônio com concentrações de 0,077; 0,153 e 0,306 $mol \cdot L^{-1}$, permitindo avaliar a resposta dinâmica da plataforma durante o processo de transferência de massa entre as fases líquida e gasosa. Os dados experimentais foram ajustados por um modelo dinâmico de primeira ordem, possibilitando a estimativa dos parâmetros aparentes associados à dinâmica de liberação de NH_3. Os coeficientes de correlação obtidos (R = 0,9808; 0,9831 e 0,9565) demonstraram boa concordância entre o modelo e os dados experimentais, evidenciando a elevada sensibilidade da plataforma desenvolvida para acompanhar a evolução temporal da concentração do gás. Os resultados também demonstraram desempenho consistente do sensor infravermelho não dispersivo (NDIR) na medição de CO_2, apresentando estabilidade e precisão durante os ensaios. A plataforma demonstrou elevada confiabilidade na aquisição, transmissão, armazenamento e visualização dos dados, evidenciando seu potencial para aplicações em contêineres de cultivo de plantas e em outras situações que requeiram monitoramento ambiental.

Palavras-chaves: Internet das Coisas. ESP32. Raspberry Pi. Monitoramento de gases. Node-RED.

ABSTRACT

This work presents the development, implementation, and validation of an Internet of Things (IoT)-based platform for the continuous, real-time monitoring of carbon dioxide (CO_2) and ammonia (NH_3) concentrations, with potential applications in container-based vertical farming systems. The proposed architecture employs an ESP32 microcontroller as the data acquisition and transmission node and a Raspberry Pi 3B as the local server, responsible for data management through the MQTT protocol. Node-RED was used for data processing, the development of real-time visualization dashboards, and the local storage of measurement records in CSV files. To validate the system, an experimental methodology was developed based on the controlled release of ammonia from ammonium hydroxide solutions with concentrations of 0.077, 0.153, and 0.306 $mol \cdot L^{-1}$, enabling the evaluation of the platform's dynamic response during the mass transfer process between the liquid and gas phases. The experimental data were fitted using a first-order dynamic model, allowing the estimation of the apparent parameters associated with NH_3 release dynamics. The correlation coefficients obtained (R = 0.9808, 0.9831, and 0.9565) indicated excellent agreement between the model and the experimental data, demonstrating the high sensitivity of the proposed platform in tracking the temporal evolution of gas concentration. The results also showed consistent performance of the non-dispersive infrared (NDIR) sensor for CO_2 measurements, exhibiting stability and accuracy throughout the experiments. The developed platform demonstrated high reliability in data acquisition, transmission, storage, and visualization, highlighting its potential for applications in container-based plant cultivation systems and other scenarios requiring continuous environmental monitoring.

Key-words: Internet of Things. ESP32. Raspberry Pi. Gas monitoring. Node-RED.

LISTA DE FIGURAS

LISTA DE SIGLAS E ABREVIATURAS

AP Access Point (Ponto de Acesso)

CSV Comma-Separated Values (Valores Separados por Vírgulas)

CO_2 Dióxido de Carbono (fórmula química)

CPU Central Processing Unit (Unidade Central de Processamento)

DNS Domain Name System (Sistema de Nomes de Domínio)

ESP32 Microcontrolador com Wi-Fi e Bluetooth integrado (Espressif Systems)

GPIO General Purpose Input/Output (Entrada/Saída de Propósito Geral)

IDE Integrated Development Environment (Ambiente de Desenvolvimento Integrado)

IoT Internet of Things (Internet das Coisas)

IP Internet Protocol (Protocolo de Internet)

MQTT Message Queuing Telemetry Transport (Transporte de Telemetria de Fila de Mensagens)

MOS Metal Oxide Semiconductor (Semicondutor de Óxido Metálico)

NDIR Non-Dispersive Infrared (Infravermelho Não Dispersivo)

NH_3 Amônia (fórmula química)

RAM Random Access Memory (Memória de Acesso Aleatório)

SMD Surface Mounted Device (Dispositivo Montado em Superfície)

STA Station Mode (Modo Estação)

TCP/IP Transmission Control Protocol / Internet Protocol (Protocolo de Controle de Transmissão / Protocolo de Internet)

UDP User Datagram Protocol (Protocolo de Datagrama do Usuário)

PPM Partes por milhão (unidade de concentração de gases)

LISTA DE SÍMBOLOS

ºC	Grau Celsius (unidade de temperatura)
%	Percentagem
h	Hora (unidade de tempo)
mL	Mililitro (unidade de volume)
V	Volt (unidade de tensão elétrica)
A	Ampère
mA	Miliampère
W	Watt
Ω	Ohm
Ppm	Partes por milhão
ppm·s⁻¹	Taxa de variação da concentração
mol·L⁻¹	Concentração molar
mL	Mililitro
g	Grama
m²	Metro quadrado
µm	Micrômetro
k	Constante aparente de velocidade (s^{-1})
C	Concentração do gás na fase gasosa
C∞	Concentração de equilíbrio da fase gasosa
C_0	Concentração inicial
$J_{app,0}$	Fluxo aparente inicial de transferência de massa
J	Fluxo de transferência de massa

$g \cdot m^{-2} \cdot s^{-1}$ Fluxo mássico superficial

Hz Hertz (unidade de frequência)

B Byte (unidade de armazenamento de dados / fluxo de informação)

µm Micrómetro (unidade de comprimento/comprimento de onda, comum no estudo de sensores NDIR)

SUMÁRIO

1 INTRODUÇÃO

O crescimento populacional, aliado às limitações de áreas cultiváveis, às mudanças climáticas e à necessidade de maior eficiência nos sistemas produtivos, tem impulsionado o desenvolvimento de novas técnicas de produção agrícola. Com isso, a agricultura vertical se destaca como uma alternativa sustentável e eficiente, pois permite o cultivo de plantas em ambientes controlados, com redução do consumo de água e maior previsibilidade da produção [1]. Esta solução é relevante em grandes centros urbanos, onde a disponibilidade de solo é limitada e a demanda por alimentos frescos é crescente.

A agricultura vertical caracteriza-se pelo cultivo em camadas sobrepostas, geralmente em locais fechados, como estufas e contêineres. Nesses ambientes, diversos parâmetros precisam ser continuamente monitorados e controlados para garantir o desenvolvimento adequado das plantas. Entre esses parâmetros, destacam-se a temperatura, a umidade, a iluminação e, principalmente, a composição gasosa do ambiente. A concentração de dióxido de carbono (CO_2) exerce papel fundamental no processo de fotossíntese, pois o mesmo está intrinsecamente relacionado aos níveis de luz e água disponíveis. Sendo diretamente responsável pelo crescimento e produtividade das culturas. Níveis inadequados de CO_2 podem limitar o desenvolvimento das plantas, enquanto concentrações elevadas podem causar estresse fisiológico [2].

Um outro gás que possuí um papel fundamental nos espaços de cultivo é a amônia (NH_3), normalmente relacionada à decomposição de materiais orgânicos e à aplicação de adubos ricos em nitrogênio. A presença de amônia em concentrações elevadas pode indicar falhas no controle, ventilação inadequada ou excesso de compostos nitrogenados, além de ser prejudicial tanto às plantas quanto as pessoas [3]. Dessa forma, o monitoramento contínuo dos níveis de CO_2 e NH_3 torna-se essencial para garantir a qualidade do ambiente de cultivo, a segurança dos operadores e a eficiência produtiva.

Em sistemas de cultivo em contêineres, como ocorre na agricultura vertical, o controle do ambiente torna-se ainda mais crítico, uma vez que esses espaços são espaços confinados, com pouca circulação de ar e totalmente dependentes de sistemas artificiais de ventilação e renovação do ar. A ausência de um sistema automatizado de monitoramento de gases pode comprometer a produtividade e

aumentar os riscos à saúde humana. Logo, a implementação de soluções tecnológicas acessíveis e confiáveis para a supervisão contínua desses parâmetros torna-se necessário.

Em paralelo, observa-se o avanço da Internet das Coisas (IoT), que possibilita a interligação de sensores e dispositivos por meio de redes de comunicação, permitindo a coleta, o armazenamento e a visualização de dados em tempo real. No setor agrícola, essa tecnologia tem sido amplamente aplicada em sistemas de monitoramento ambiental, estufas inteligentes, controle de irrigação e automação de processos.

Com isso, dispositivos como o ESP32 e o Raspberry Pi são referencias por apresentarem baixo custo, elevada capacidade de processamento e ampla compatibilidade com aplicações de monitoramento e automação. O ESP32 é amplamente utilizado como nó de aquisição e transmissão de dados por meio de redes Wi-Fi, enquanto o Raspberry Pi atua como servidor local, responsável pelo recebimento, armazenamento e gerenciamento das informações coletadas [4]. A comunicação entre esses dispositivos pode ser realizada por meio de protocolos próprios para aplicações IoT, como o MQTT (Message Queuing Telemetry Transport), que se destaca por sua leveza, confiabilidade e eficiência.

Apesar do avanço das tecnologias disponíveis, ainda é comum o uso de métodos manuais ou sistemas isolados no monitoramento de cultivos protegidos, o que dificulta a análise contínua dos dados e compromete a eficiência na tomada de decisões. Neste contexto, destaca-se a necessidade de soluções integradas que possibilitem a aquisição, transmissão e visualização das informações em tempo real.

Além da visualização instantânea, o armazenamento contínuo dos dados, como o de CO_2 e NH_3 destacados, possibilita a realização de análise do comportamento temporal desses gases em ambientes confinados, como é o caso do cultivo em contêiners. As curvas geradas pela plataforma podem ser utilizadas para avaliar tendências de acúmulo, resposta dinâmica dos sensores e possíveis relações entre as concentrações medidas e processos de cunho biológico ou físico-químico. Assim, o sistema proposto apresenta potencial não apenas como ferramenta de monitoramento para agricultura vertical, mas também como base para estudos futuros de dinâmica de gases.

Portanto, este trabalho tem como objetivo o desenvolvimento de um sistema de monitoramento de CO_2 e NH_3 com possibilidade de aplicação em plantação vertical

instalada em contêiner, utilizando um ESP32 como unidade de aquisição de dados e um Raspberry Pi como servidor central, com comunicação via rede Wi-Fi por meio do protocolo MQTT. O sistema proposto tem o propósito de permitir o acompanhamento em tempo real das concentrações desses gases, contribuindo para a melhoria das condições ambientais do cultivo, o aumento da produtividade e confiabilidade do ambiente aplicado.

2 CAPÍTULO 1 – HARDWARE

Neste capítulo serão apresentados os principais componentes de hardware empregados no desenvolvimento do sistema proposto. Inicialmente, serão abordados os diferentes tipos de sensores utilizados para a detecção de CO_2 e NH_3, destacando seus princípios de funcionamento, vantagens e limitações. Em seguida, serão apresentados os dispositivos responsáveis pelo processamento e gerenciamento das informações, como os microcontroladores e minicomputadores, enfatizando suas características e a importância de sua aplicação no sistema de monitoramento desenvolvido.

2.1 Sensores

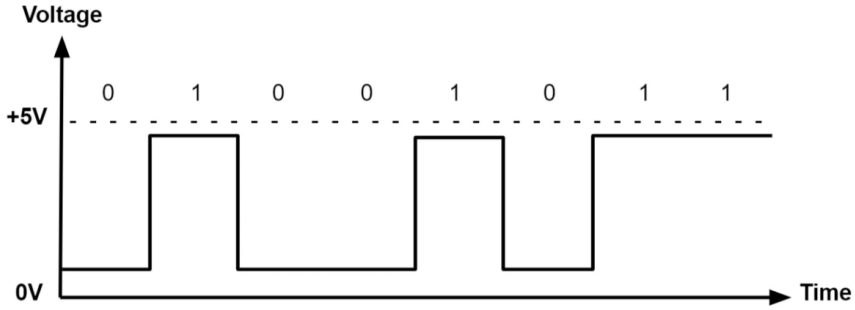
Sensor, derivado de sentir (do latim sentire), significa perceber, detectar ou captar algo. São elementos que reagem a determinados estímulos desencadeando reações especificas a partir disso.

Estes por si só, são componentes sensíveis que reagem a determinada forma de energia (térmica, química, luminosa, elétrica, mecânica, etc.) convertendo para sua saída, sinais elétricos [5]. Ou seja: funcionam como transdutores. Seus sinais de saída podem ser classificados como digitais ou analógicos.

2.1.1 Sensor digital

Sensores digitais ou binários, fornecem somente dois estados ao longo do tempo: verdadeiro ou falso, correspondendo aos valores lógicos 1 e 0. Como nenhuma grandeza física assume apenas esses dois valores de forma natural, esse tipo de sensor apenas identifica se determinada grandeza física alcançou ou não um valor previamente definido, como mostrado na Figura 1.

Figura 1 - Sinal digital

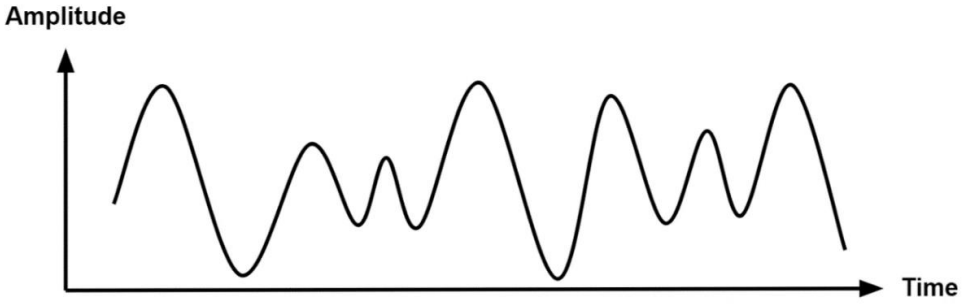


Fonte: [6]

2.1.2 Sensor Analógico

Já os sensores analógicos fornecem valores que variam ao longo do tempo, podendo representar infinitas amostras dentro deste intervalo. Ou seja: trata-se de um sinal analógico e contínuo (Figura 2). Frequentemente são utilizados para responder a mudanças de luz, som, temperatura, posição, pressão ou outros fenômenos físicos. Normalmente a saída desses sensores são padronizados em (4-20) miliamperes

Figura 2 - Sinal analógico



Fonte: [6]

2.2 Sensores de dióxido de carbono (CO_2)

O dióxido de carbono é um gás incolor e inodoro, sendo assim, a única maneira de medir-se a concentração do mesmo é utilizando sensores eletrônicos [7]. A partir do mesmo, é possível obter a quantidade de partícula por milhão (ppm) de CO_2 presente no ar. Com isso, há três tipos de sensores mais comuns de serem utilizados: NDIR, condutores térmicos e semicondutor de óxido metálico (MOS).

2.2.1 Sensor NDIR (infravermelho não dispersivo)

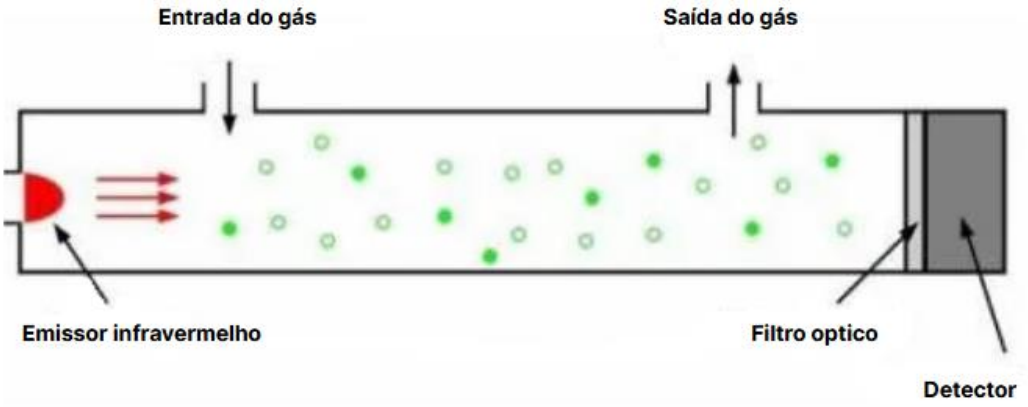
Diferentes materiais presentes no planeta terra absorvem tipos específicos de luz. Materiais de cores vermelhas por exemplo: refletem as luzes relacionadas a cor vermelha e absorvem todas as outras. Quando se analisa esse fenômeno a nível atômico e molecular, é possível identificar quais comprimentos de onda cada material absorve, inclusive as faixas que são invisíveis ao olho humano, como a infravermelho. É exatamente nessa ideia que os sensores NDIR se baseiam.

O funcionamento dos sensores do tipo infravermelho não dispersivo (Figura 3) é baseado no princípio de que cada gás absorve luz em comprimentos de onda diferente. O sensor utiliza a faixa do infravermelho, pois o CO_2 apresenta forte sensibilidade de absorção nela [7]. Com isso, o equipamento emite as ondas de infravermelho, que passarão pela câmara onde o gás do ambiente estará presente e em sequência, as ondas atingirão o detector de infravermelho, como mostrado na Figura 4.

Figura 3 - Exemplo de sensor DNIR

Fonte: Nenvitech (2025)

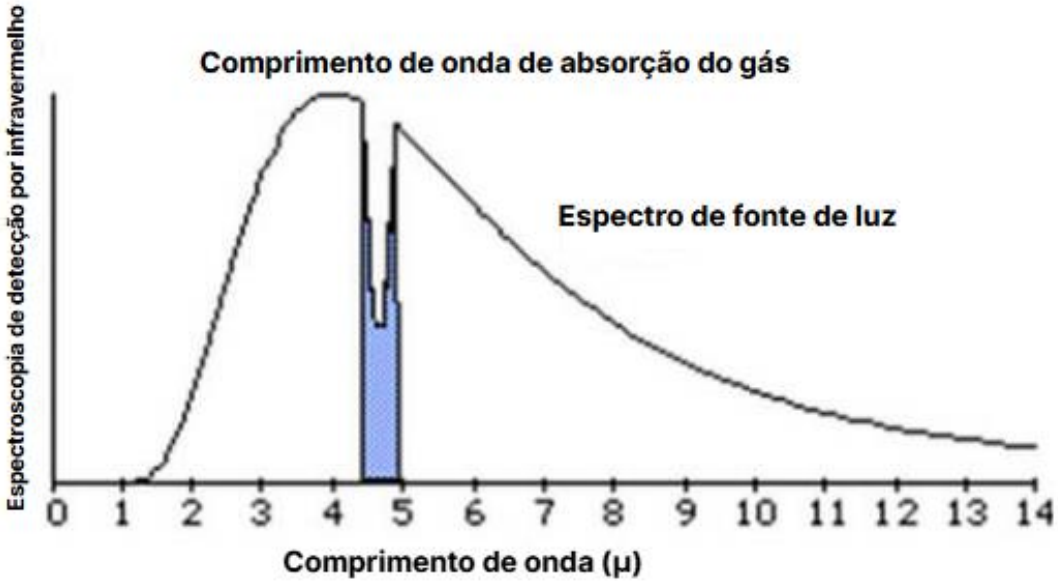
Figura 4 - Funcionamento do sensor NDIR



Fonte: [8]

A concentração de dióxido de carbono será obtida a partir da diferença de intensidade da emissão e recepção dessa onda previamente conhecida, como visto na Figura 5.

Figura 5 - Relação entre a onda emitida e recebida



Fonte: [7]

Sensores tipo NDIR são referências em quesito de vida útil, além de possuírem ampla faixa de medição. É o tipo de sensor mais popular [7].

2.2.2 Sensor semicondutores de óxido metálico (MOS)

Este tipo de sensor se utiliza do princípio de oxidação ou redução (oxirredução) em um eletrodo para medir a corrente elétrica e obter a concentração de CO_2 [9]. Estes elementos utilizam materiais sensíveis a gases de oxido metálico, como o dióxido de estanho (SnO_2), para que quando o dióxido de carbono entrar em contato com o mesmo, ocorra a reação química nas superfícies do material, alterando a resistência elétrica do mesmo.

Quando se conhece a resistência elétrica do eletrodo sem a presença de CO_2 no ambiente, torna-se possível determinar a faixa de medição, pois a alteração da resistência elétrica é diretamente proporcional a concentração do dióxido de carbono

presente. A saída linear é uma das vantagens dos sensores eletroquímicos em relação as outras tecnologias, além de serem menos vulneráveis a variação de temperatura e umidade [7]. Um sensor MOS pode ser visto na figura a seguir:

Figura 6 - Exemplo de sensor MOS

Fonte: Winsen (2025)

2.2.3 Sensor do tipo condução térmica

Diferentes compostos gasosos apresentam condutividade térmica diferentes entre sí, isto quer dizer que corpos em exposição a estes gases irão perder calor em quantidades diferentes ao longo do tempo [10]. O princípio de funcionamento de um sensor do tipo térmico se baseia nisso.

Os sensores de CO_2 do tipo condução térmica medem a concentração do gás baseado na relação entre a condutividade térmica do dióxido de carbono e sua concentração no ambiente. O mesmo é composto por uma fonte de energia térmica (geralmente um termopar) mantida a uma temperatura constante, juntamente com um sensor de temperatura para servir de referência. Quando o dióxido de carbono flui através do sensor, parte do calor é retirado do elemento térmico, resultando em uma diferença de temperatura e alterando a resistência elétrica do termopar. O termopar é ligado a uma ponte de wheatstone afim de medir a resistência elétrica a partir da

corrente, para então retornar um feedback para calcular-se a concentração de CO_2 [9]. Trata-se de um sensor muito susceptível a erros devido a variações de temperatura e outros gases, principalmente a amônia (NH_3). Um exemplo de um sensor smd(Dispositivo de Montagem em Superfície) do tipo térmico pode ser visto na Figura 7

Figura 7 - Exemplo de sensor térmico SMD

Fonte: [11]

2.3 Sensores de amônia (NH_3)

A amônia é um gás incolor e de odor muito forte, é altamente irritante a pele e aos mucos do corpo humano, além disso, exposições a altos níveis de concentração de amônia pode levar a complicações as vias respiratórias [12]. Tornando o monitoramento e controle deste composto por sensores, essencial.

Com isso, há quatro principais tipos de sensores de amônia presentes no mercado: semicondutores (MOS), infravermelhos (NDIR), catalíticos e fotoionização. O princípio de funcionamento dos semicondutores (MOS) e infravermelhos (NDIR) já foram explicados anteriormente.

2.3.1 Sensor catalítico

O princípio de funcionamento deste tipo de sensor é feito através de reação de oxidação da amônia em um catalisador (acelerador de reação química por meio de contato mecânico), geralmente de platina. O resultado desta reação é uma quantidade de energia térmica gerada proporcional a quantidade de concentração de amônia presente. Esta energia é transferida a uma bobina que altera sua resistência elétrica de acordo com a temperatura da mesma [13]. Ou seja: o sistema atua como um transdutor químico-físico-elétrico.

As principais características desse tipo de sensor se dão a alta sensibilidade para medições a baixas concentrações de amônia, alta robustez contra choques mecânicos e contaminações de outros gases. Apesar do baixo custo, apresenta vida útil muito menor se comparado a outras tecnologias de sensores de NH_3 [14].

O sensor seccionado em questão pode ser visto na Figura 8:

Figura 8 - Exemplo de sensor catalítico de amônia

Fonte: draeger (2025)

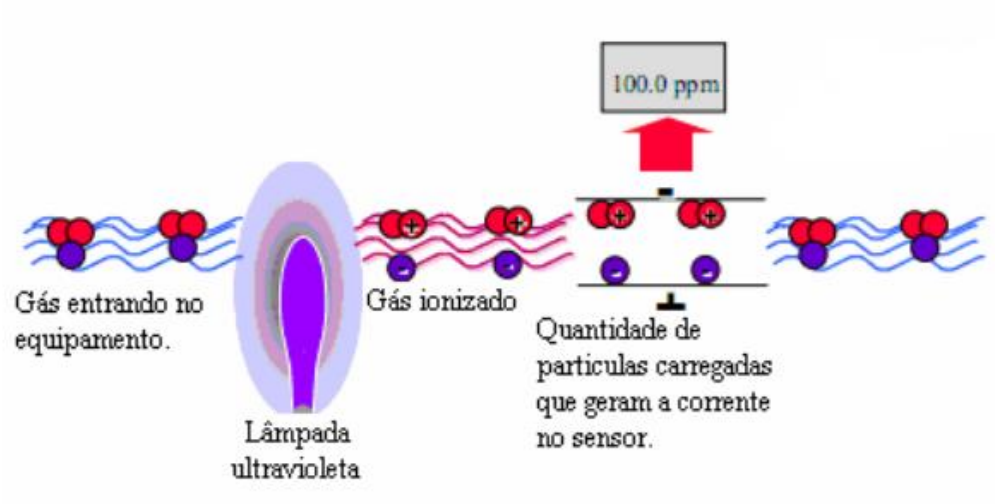
2.3.2 Sensor de fotoionização

Trata-se de um dos sensores com princípio de funcionamento mais interessantes. O sensor por fotoionização utiliza uma lâmpada ultravioleta para emitir fótons com alta energia para quando estes entrarem em contato com as moléculas de

amônia, “arrancarem” um elétron da banda de condução, gerando um íon positivo (ionização). Consequentemente o gás de amônia ficará carregado eletricamente.

Em seguida, o gás passará por eletrodos coletores, que irão receber a carga elétrica do composto. Gerando um sinal proporcional a concentração de amônia presente [15]. A Figura 9 ilustra o princípio de funcionamento deste sensor.

Figura 9 - Princípio de funcionamento do sensor de fotoionização



Fonte: [15]

Este tipo de sensor também possui alta sensibilidade para captações de baixas concentrações de amônia, além disso, possuem leituras rápidas em tempo real. Em contrapartida, seu custo é alto em relação aos outros tipos de sensores [16].

Figura 10 - Exemplo de sensor fotoionizador

Fonte: Shenzhen Nuoan (2025)

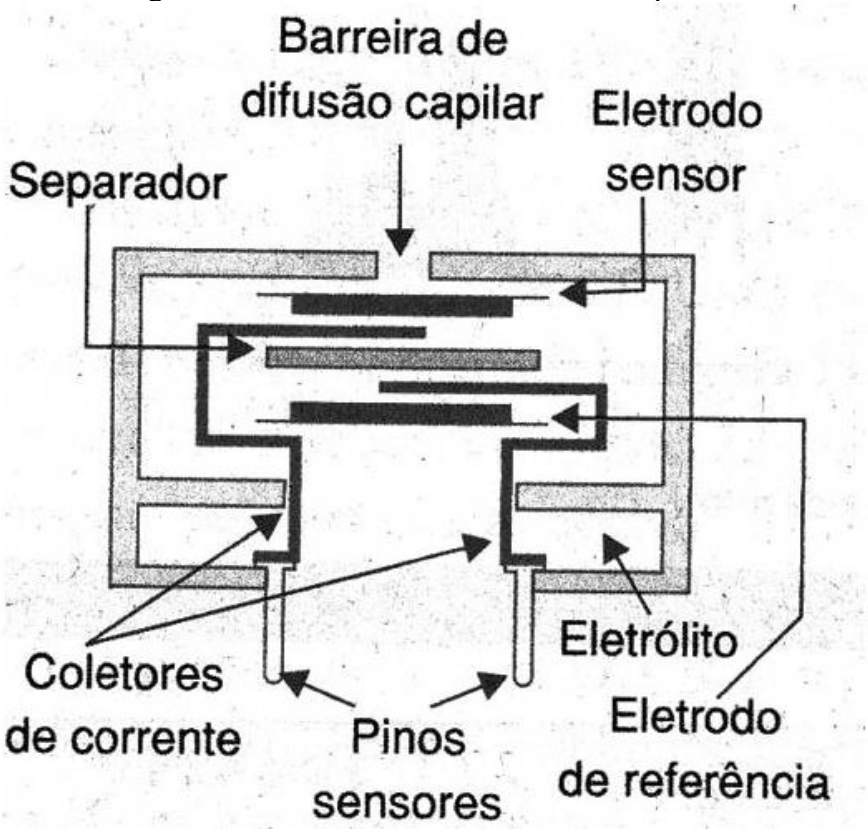
2.3.3 Sensor eletroquímico

Estes tipos de sensores utilizam uma reação química de redução ou oxidação de um eletrodo para medir a concentração de um gás especifico em determinado ambiente.

Esta reação ocorre quando o gás alvo penetra uma membrana porosa que bloqueia partículas indesejadas, como poeira e pequenas quantidades de líquidos, em seguida, o mesmo entra em contato com o eletrodo sensor (feito de materiais mais refinados como ouro e platina para potencializar e sensibilizar a leitura) que irá sofrer uma oxidação ou redução (dependendo do gás), com isso, haverá a perda ou ganho de elétrons livres na superfície do eletrodo.

Para que um circuito elétrico seja formado, há outro eletrodo de referência isolado por um separador que garante a estabilidade química do mesmo [34]. A estrutura descrita pode ser visualizada na Figura 11.

Figura 11 - Estrutura de um sensor eletroquímico



Fonte: Newtoncbraga (2026)

Este tipo de sensor possui uma sensibilidade muito alta, porém é extremamente sensível a temperatura e umidade, visto que são variáveis que influenciam diretamente na reação do eletrodo, e justamente por isso que a maioria dos sensores presentes no mercado possuem compensação por temperatura e umidade já inclusa no aparato. A Figura 12 mostra um sensor com tais compensações presentes.

Figura 12 - Sensor de amônia com compensador

Fonte: ComWinTop(2026)

2.4 Microcontroladores

Microcontroladores (Figura 13) são “minis computadores”, compostos por um chip de circuito integrado único, contendo memória RAM, memoria PROM, núcleo de processador central (CPU) e capacidade de comportar periféricos de entrada e saída para serem programados (entradas I/O).

A principal vantagem dos microcontroladores são seu baixo custo e baixo consumo elétrico, bastando apenas uma pequena fonte DC externa para manter seu funcionamento.

Figura 13 - Microcontrolador

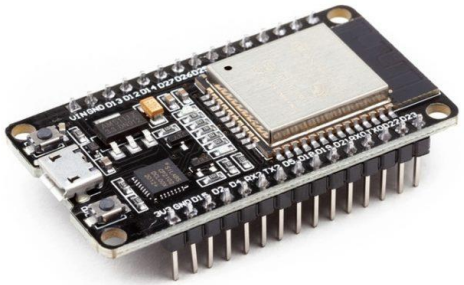
Fonte: STM (2025)

Existem diversos tipos de microcontroladores no mercado, com cada fabricante mantendo determinado foco em seus produtos finais, variando desta forma, tamanho, formato, alimentação, complexidade e aplicação [17].

2.4.1 Microcontrolador ESP32

O ESP32 (Figura 14) é um microcontrolador de baixo custo, criado pela empresa de tecnologia Espressif Systems, desenvolvido principalmente para projetos IOT's, ou seja: para aqueles que não possuem fio, como bluetooth e wi-fi[18]. Com isso, o mesmo possui ambos módulos de comunicação integrado. Além disso, o mesmo possui uma versatilidade de poder ser alimento a partir de uma conexão USB, dispensando uma fonte externa dedicada a isso.

Figura 14 - Microcontrolador ESP32



Fonte: [18]

Ele pode ser programado em diversas linguagens de programação, como C/C++, Micropython, JavaScript e Lua. Além disso, o mesmo é composto por um processador de 32 bits, com clock de até 240MHZ; com uma memória RAM e FLASH

respectivamente de 80KB e 520KB. Características que suprem muito bem as aplicações projetadas para o mesmo.

2.5 Minicomputadores

Minicomputadores são computadores compactos, com baixo nível de processamento e consequentemente, baixo nível de consumo energético.

Eles diferem dos computadores convencionais por possuírem recursos específicos para integração direta com algum hardware eletrônico especifico, como pinos de entrada e saída de uso geral (GPIO). Além de terem um custo muito menor frente aos computadores convencionais, os tornando ideais para projetos de eletrônica [19].

Figura 15 - Exemplo de minicomputador

Fonte: [20]

2.5.1 Minicomputador Raspberry Pi

Raspberry Pi é uma linha de computadores modulares de placa única, criada pela Raspberry Pi Foundation. “Sua origem está por volta de 2006, na Universidade de Cambridge, quando professores perceberam o declínio do interesse dos alunos em programação” (LOJA PM, 2024). A solução foi acessível e que pudesse servir como plataforma de ensino e experimentação técnica: o minicomputador raspberry pi.

Após anos de testes e protótipos, o primeiro modelo: Raspberry Pi Model B, foi lançado em 2012, como mostrado na Figura 16. Posteriormente, foram lançados modelos mais diversos, como model A, série zero, etc.

Figura 16 - Primeiro modelo Raspberry Pi B

Fonte: [21]

Suas principais características estão voltadas ao alto nível de modularidade, baixíssimo consumo energético, a compactabilidade, presença de interface GPIO para controle de hardware (como sensores e atuadores) [22].

3 CAPÍTULO 2 – ELEMENTOS VIRTUAIS

3.1 Linguagem de programação e framework

Uma linguagem de programação é a forma do homem instruir a máquina. Ou seja, é um sistema formal com regras e sintaxes (relação entre comandos e organização das estruturas), de como as instruções são passadas. Em resumo, é a sequência lógica de passos para resolver-se um problema.

Já o framework é uma estrutura previamente desenvolvida com base em uma linguagem de programação já existente, ele é extremamente útil para projetos de engenharia, pois assim, não se perde tempo desenvolvendo algo que já padronizaram para se utilizar.

Assim que o código fonte (algoritmo utilizando a linguagem de programação) é previamente escrito, este passará por um compilador para converter a forma textual em código binário, permitindo desta maneira, que o dispositivo “entenda” as instruções [23].

3.2 Editor de código fonte

Os editores de código fonte são programas destinados ao auxílio do desenvolvimento de um código fonte, possuindo diversos recursos destinados a isso, como preenchimento automático, formatação padronizada de código, atalhos de teclado, entre outros [24].

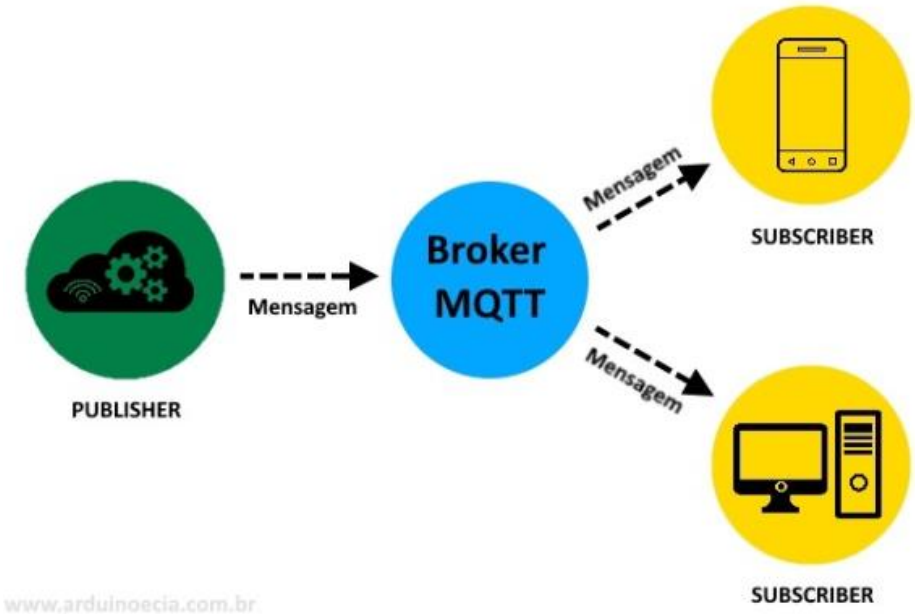
O editor de código fonte mais conhecido e utilizado é o Visual Studio Code, onde o mesmo é de código aberto e gratuito, desenvolvido pela Microsoft Studio.

3.3 Protocolo MQTT

O protocolo MQTT (*Message Queunig Telemetry Transport*), é uma forma padronizada de comunicação entre dispositivos que o utilizam e que compartilham do mesmo IP, ou seja: estejam na mesma rede.

Ele foi projetado para ser um transportador de mensagens de publicadores (envios de dados, geralmente sensores) e assinantes (receptores, geralmente computadores e celulares), de forma a ser extremamente leve, e ocupar uma banda de rede muito pequena [25]. Os envios e os recebimentos são gerenciados por um elemento intermediário: denominado broker, o gerenciador dos dados. É ele que garantirá a entrega e o recebimento das mensagens. A Figura 17 exemplifica a estrutura do MQTT.

Figura 17 - Estrutura do MQTT



Fonte: Arduino & Cia (2025)

Além disso, o MQTT oferece três níveis de qualidade de serviço (QoS), recurso este, que permite gerenciar o nível de confiabilidade da entrega dos dados;

Para QoS 0, a mensagem é entregue no máximo uma vez, sem confirmação. Ideal para aplicações onde a perda ocasional de mensagens é aceitável.

QoS 1: A mensagem é entregue pelo menos uma vez, com confirmação. Esse nível é utilizado quando é importante garantir que a mensagem chegue ao destino, mesmo que ocorra duplicação.

Já para QoS 2: A mensagem é entregue exatamente uma vez, utilizando um processo de confirmação mais complexo. Este é o nível mais seguro, adequado para aplicações críticas onde a duplicação não é aceitável [26].

3.4 Node-RED

Inicialmente desenvolvida pela IBM, o Node-RED é uma ferramenta de desenvolvimento baseada em fluxos construídos através de nós ou nodes. Estes por sua vez, interagem com os dados de entrada respondendo ao próximo nó do fluxo. A mesma é baseada em navegador [27].

A principal característica desta ferramenta é simplificar o problema a partir de uma representação visual do mesmo, pois evita grandes linhas de códigos ou em alguns casos, a necessidade de entender o código fonte de cada node. O Node-RED “abraçou” a maioria dos projetos IoT (internet das coisas) pois possui alta capacidade de trabalhar com dados em tempo real em baixa latência e baixo consumo de recursos de hardware [28].

Com isso, seu suporte nativo é voltado para protocolos de comunicação mais leves, como o MQTT.

4 CAPÍTULO 3 – MATERIAIS E METODOS

4.1 Softwares

4.1.1 Visual Studio Code e PlatformIO

Para a edição dos códigos para o ESP32, foi utilizado o Visual Studio Code, da Microsoft. Também seria possível utilizar o software Arduino IDE, porém o mesmo apresenta menos recursos disponíveis.

Já o platformIO será um extensor instalado dentro do próprio Visual Studio Code, de forma a servir como ferramenta de acesso aos frameworks a serem utilizados no ESP32, ou seja: ele será o responsável por identificar o microcontrolador, além de fornecer bibliotecas prontas para o uso.

4.1.2 Mosquitto e Node-RED

Neste trabalho, será utilizado o broker Mosquitto, que implementa o protocolo de comunicação MQTT, sendo responsável pelo gerenciamento da troca de mensagens entre os dispositivos do sistema.

Já para o gerenciamento dos dados fornecidos pelos sensores e criação da interface gráfica para armazenamento e visualização das informações, como já dito anteriormente, foi utilizado o Node-RED.

4.2 Materiais

Para se realizar o monitoramento e armazenamento dos níveis de amônia e dióxido de carbono, os componentes adiante foram utilizados.

4.2.1 Microcontrolador ESP32

Componente responsável em coletar os níveis das saídas dos sensores e converte-los em valores de concentração dos gases medidos, além realizar a comunicação com o minicomputador.

Figura 18 - ESP32

Fonte: Mamute eletrônica (2026)

4.2.2 Minicomputador Raspberry Pi 3B

Para recepção dos dados do ESP32, armazenamento dos níveis dos gases e execução do broker, foi utilizado o mini computador Raspberry pi 3B.

Figura 19 - Raspberry Pi 3B

Fonte: Autoria Própria

4.2.3 Sensor de CO_2

O sensor de concentração de dióxido de carbono a ser utilizado foi o modelo chines de código CWT-CO2-2K-INA-I como visto na Figura 20 . O mesmo possui uma faixa de medição de 0-2000ppm, faixa de alimentação de 10-30v contínuos com saída 4-20mA. Seu princípio de funcionamento é baseado em NDIR (já explicado anteriormente). Dentre suas principais vantagens, se encontra a alta precisão de leitura e vida útil maior em relação aos outros tipos.

No entanto, variações bruscas de temperatura e umidade causam pequenas flutuações de leitura, e é por isso, que este sensor possuí um sistema integrado de alto calibração baseado nessas variáveis.

Figura 20 - Sensor de dióxido de carbono utilizado

Fonte: ComWinTop(2026)

4.2.4 Sensor de NH_3

Já para mensurar a concentração de amônia, foi utilizado um sensor eletroquímico de amônia (Figura 21) com faixa de medição de 0 a 100 ppm e saída analógica de 0 a 5 V, comercializado para aplicações de monitoramento ambiental e automação industrial. O mesmo também possui um hardware interno de compensação por variação de temperatura e umidade. Uma de suas principais desvantagens é alta sensibilidade a umidade. Sendo que o mesmo não pode ser exposto a ambientes com umidade acima de 95%.

Figura 21 - Sensor de amônia utilizado

Fonte: ComWinTop(2026)

4.2.5 Fonte de alimentação

Visando robustez no projeto, optou-se por utilizar uma fonte chaveada modelo T60-C (fabricante Mean Well) para alimentar os sensores e o microcontrolador ESP32. A mesma pode ser alimentada tanto em 110v, quanto em 220v. Suas saídas contam com tensões de alimentação de 5v,15v e -15v contínuos. Sendo que a potência entregue entre elas é de 60w.

Figura 22 - Fonte utilizada no projeto

Fonte: Lemaqs (2026)

4.2.6 Resistores

Para que a saída (4-20mA) do sensor de CO_2 seja convertida em tensão, foi utilizado um resistor shunt de 150Ω ,1/4W em seus terminais de saída.

Além disso, três resistores de 10kΩ serão utilizados para servirem de divisores de tensão, de forma a se obter uma faixa de 0v a 3.33v na saída do sensor de NH_3.

4.2.7 Componentes para confecção da placa de circuito impresso

4.2.7.1 Placa de fenolite

Para que todos os componentes eletrônicos sejam fixados em uma estrutura mais confiável, é necessário a produção de uma placa eletrônica. Para tal, foi necessária uma placa de fenolite (dimensões dependerão do projeto), como visto na Figura 23.

Figura 23 - Placa de Fenolite

Fonte: Mamute eletrônica (2026)

4.2.7.2 Papel glossy ou fotográfico

Para a impressão das trilhas, utilizando o método a quente, foi necessário o papel fotográfico com gramatura entre 95g/m² a 150g/m². Papeis com gramatura acima dessa faixa não são recomendadas, pois o calor necessário para transferir a tinta é maior do que a placa de fenolite suporta.

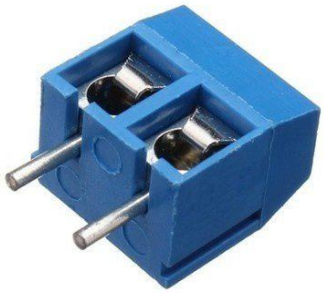
4.2.7.3 Percloreto de ferro

Para que as trilhas sejam formadas e posteriormente o cobre não necessário seja corroído na placa de fenolite, a utilização da solução de percloreto de ferro é recomendada. Para este trabalho, foi adquirido o produto em pó e posteriormente o mesmo foi diluído em água. No entanto há opções prontas para o uso no mercado.

4.2.7.4 Conectores tipo borne

Para conectar os cabos de alimentação, foi utilizado conectores bornes de 2 vias (Figura 24). No entanto, o mesmo poderia ser feito soldando diretamente na placa.

Figura 24 - Conector borne 2 vias



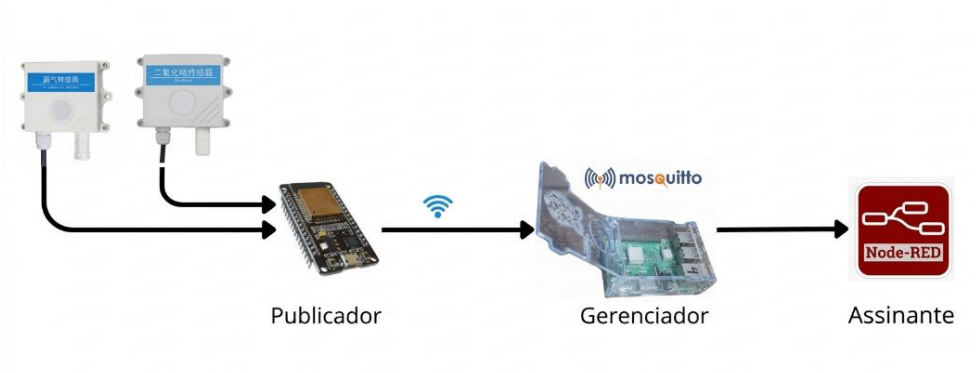
Fonte: Eletrogate (2026).

5 CAPÍTULO 4 – METODOLOGIA DE DESENVOLVIMENTO DO SISTEMA

Para desenvolver o trabalho, foi utilizado dois tipos de sensores: o de detecção de nível de concentração de amônia e de dióxido de carbono. Os valores de tensão entregues destes, são coletados pelo microcontrolador ESP32, onde estes dados serão tratados, de forma que sejam convertidos em dados legíveis.

Posteriormente, o ESP32 irá transmitir ambas informações dos sensores em formato. json (ambos valores separados por vírgula em uma mesma mensagem) para o broker Mosquitto executado no mini computador raspberry pi 3b por meio de uma rede wi-fi. Ou seja: o ESP32 atuará como um publicador de mensagens, enquanto o broker Mosquitto (funcionando no raspberry) atuará como gerenciador dessas mensagens, podendo controlar elementos publicadores e assinantes. Em seguida, terá o Node-RED executando dentro do mini computador. Este, atuará como um elemento assinante dos dados recebidos. O diagrama do processo descrito pode ser visto na Figura 25.

Figura 25 - Diagrama da comunicação entre o publicador e o assinante



Fonte: Autoria própria

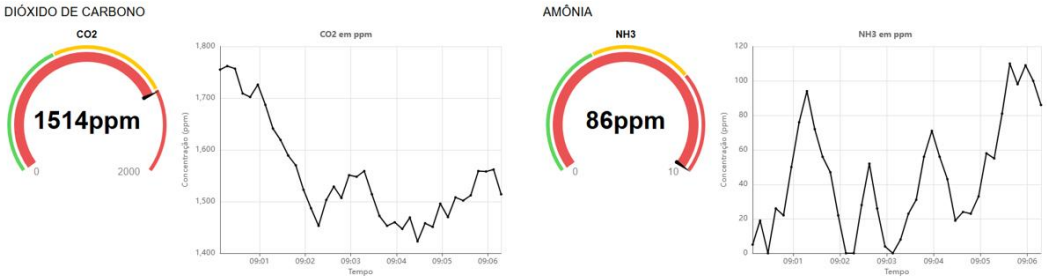
A partir do Node-RED, os dados serão recepcionados e divididos em duas vertentes (duas series de nodes): A primeira voltada para uma visualização em tempo real e outra para armazenamento local dos dados históricos. O fluxograma pode ser visto na Figura 26.

Figura 26 - Fluxograma do Node-RED

Fonte: Autoria própria

Com todos os nodes séries concluídos, a interface dos dados em tempo real foi exposta de forma a mostrar um medidor analógico para as medições instantâneas e um pequeno gráfico dinâmico das últimas 8 horas de medição registradas, como visto na Figura 27:

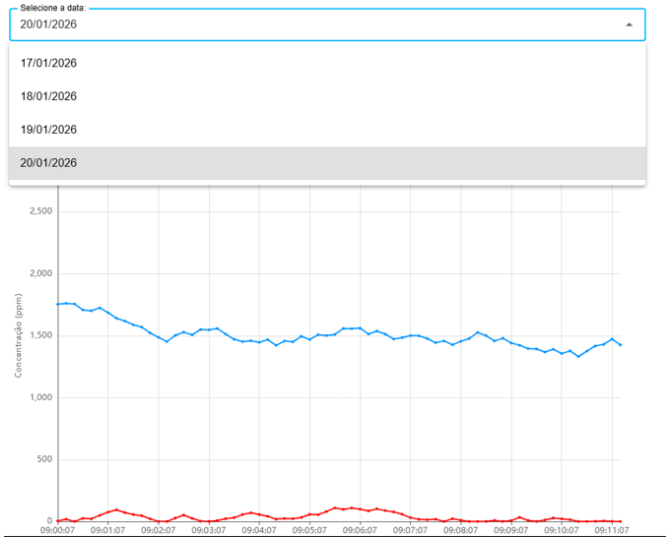
Figura 27 - Exemplo de dashboard dos dados em tempo real



Fonte: Autoria própria

E de uma maneira semelhante, os dados salvos localmente poderão ser vistos pela dashboard, porém com ambos os dados de concentração no mesmo gráfico.

Figura 28 - Exemplo da dashboard dos dados armazenados localmente



Fonte: Autoria própria

Para a elaboração do condigo fonte presente no ESP32, foi utilizado o editor Visual Studio Code com a extensão PlatformIO (addon voltado para microcontroladores). A escolha é devido a maior quantidade de material disponível.

5.1 ESP32: Instalando o PlatformIO e configurações iniciais

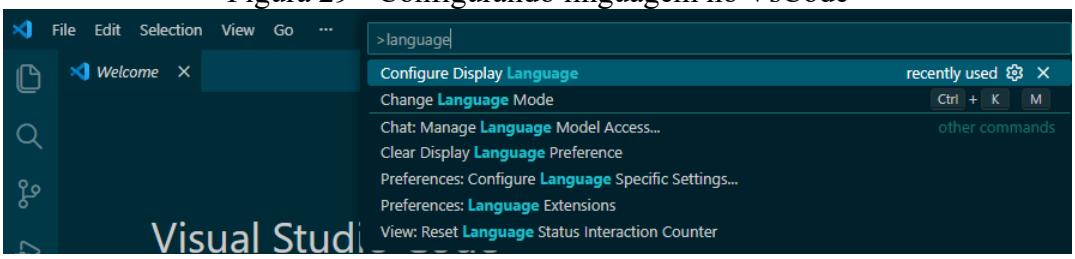
Inicialmente é necessário anexar a extensão “platformIO” ao Visual Studio Code, pois é este que reconhecerá diversos dispositivos programáveis, como o ESP32.

Embora o fabricante do ESP32 forneça um framework oficial, o ESP-IDF, o mesmo não foi utilizado por possuir pouco suporte didático disponível, além de possuir uma curva de aprendizado e complexidade muito acentuada. Dito isso, foi utilizado

um framework do Arduino, pois este, facilita ainda mais o processo de programação para iniciantes.

Após instalar o VsCode, é recomendado colocar a linguagem da interface em português. O mesmo pode ser feito abrindo o atalho de comando: Ctrl + Shift + P, e em seguida digitando “language”, em seguida selecione a opção: “configure Display language”(Figura 29), após isto, selecione português (Brasil).

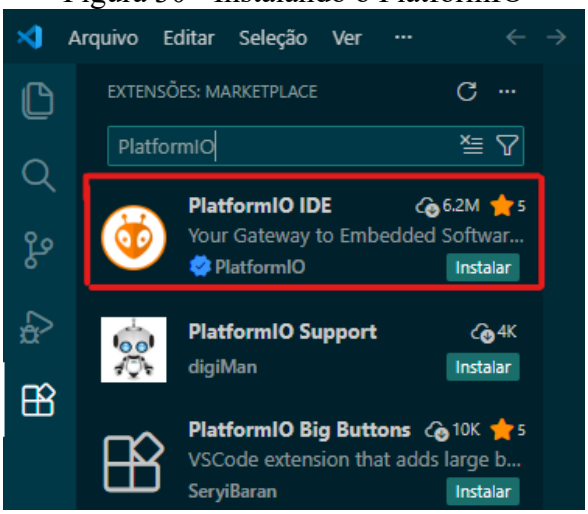
Figura 29 - Configurando linguagem no VsCode



Fonte: Autoria própria.

A extensão PlatformIO pode ser buscada através do ícone “extensões” presentes no canto superior esquerdo, abrirá um campo de busca onde poderá ser digitado a extensão demandada, como visto na Figura 30.

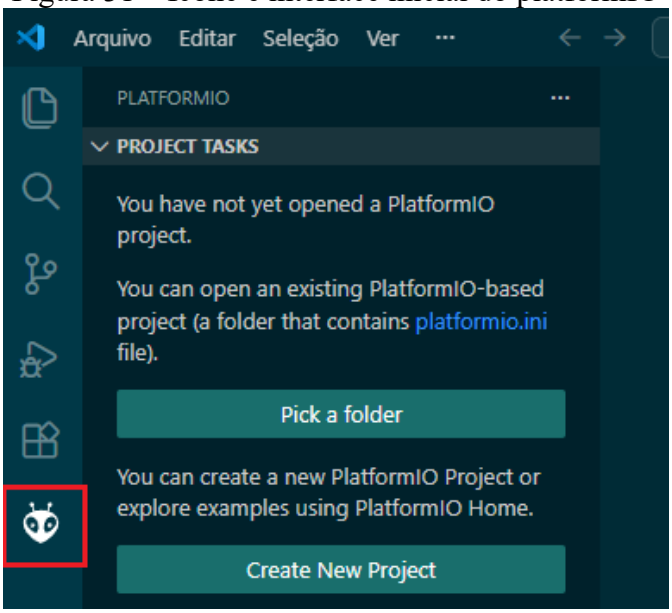
Figura 30 - Instalando o PlatformIO



Fonte: Autoria própria.

Após instalar o platformIO IDE, o ícone do mesmo aparecerá no final do canto inferior esquerdo, como visto na Figura 31. Desta maneira um novo projeto poderá ser feito a partir da extensão, clicando em: “Create New Project”.

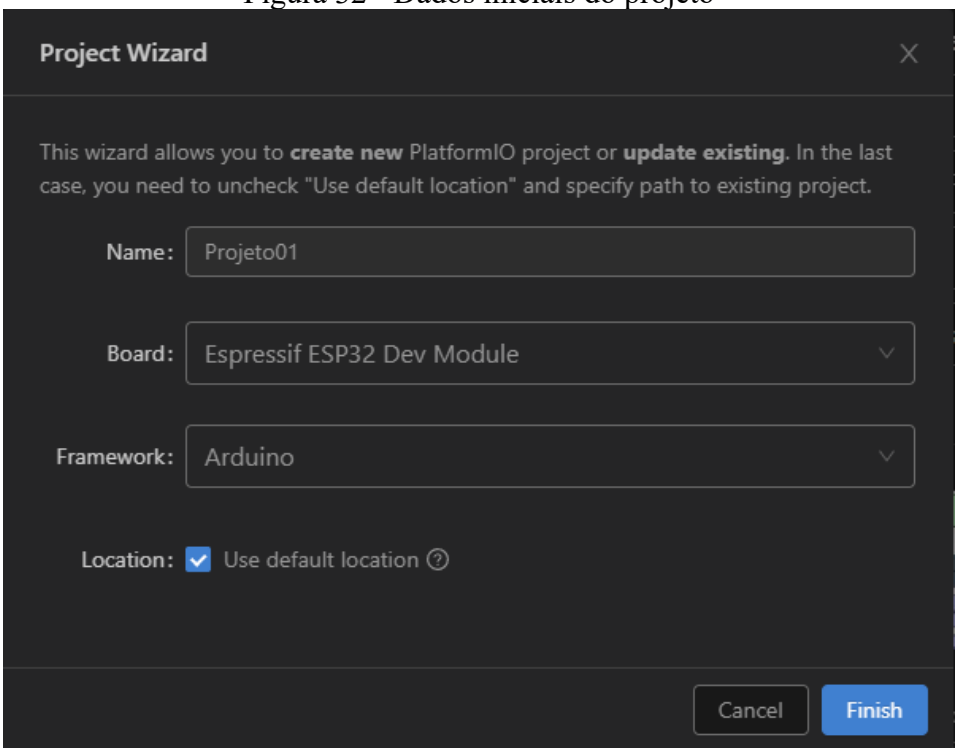
Figura 31 – Icone e interface inicial do platformIO



Fonte: Autoria própria.

Primeiramente, é necessário definir um nome para o projeto, bem como indicar a placa de desenvolvimento e o framework adotados. Neste projeto, a placa utilizada é a Espressif ESP32 Dev Module, juntamente com o framework do Arduino. As especificações podem ser visualizadas na Figura 32

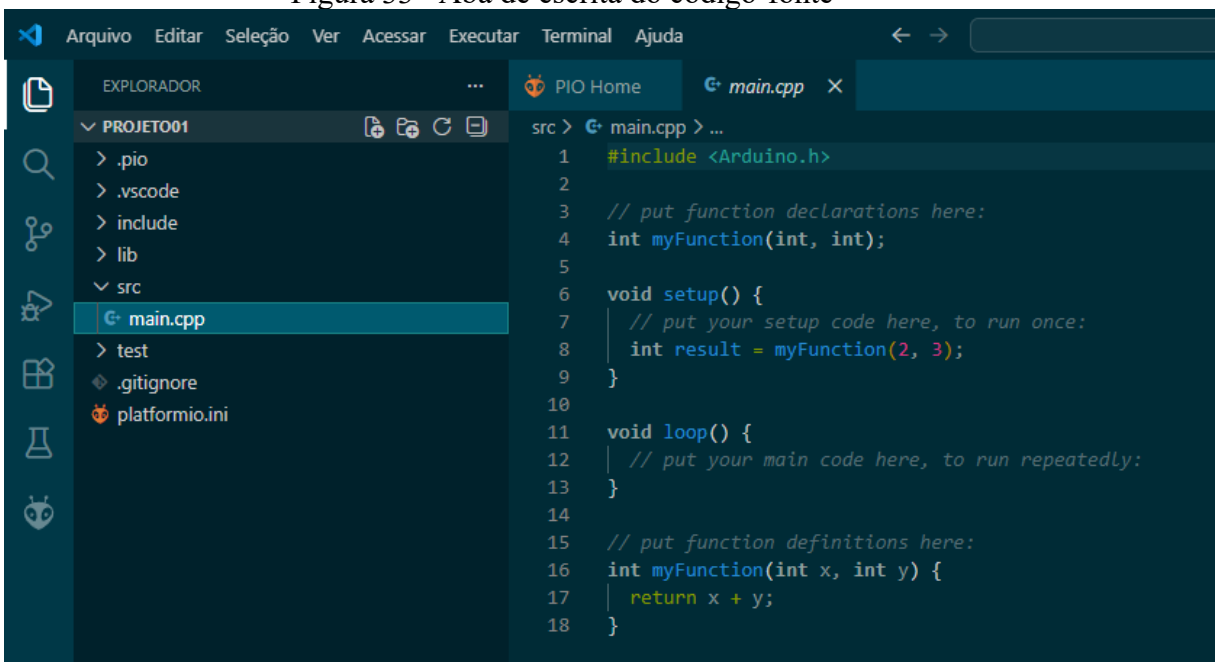
Figura 32 - Dados iniciais do projeto



Fonte: Autoria própria.

Após a criação do projeto ser concluída, o arquivo sairá da aba PlatformIO e seguirá para a aba explorador. A área para a escrita do código fonte estará presente em “src” com o nome “main.cpp”, na sub guia a direita da aba principal, como visto na Figura 33. O nome dado ao projeto poderá ser localizado na pasta documentos, na pasta raiz do VsCode do computador.

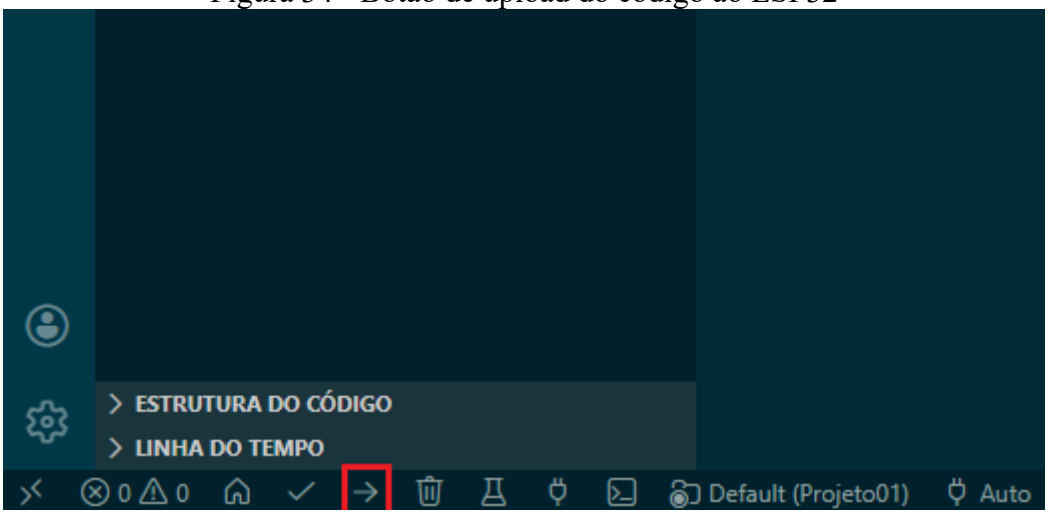
Figura 33 - Aba de escrita do código-fonte



Fonte: Autoria própria.

Após o término dos passos iniciais, a plataforma já estará pronta para receber o ESP32, que ao ser conectado via entrada USB, poderá receber toda a programação feita a partir do botão upload presente no canto inferior esquerdo, como visto na Figura 34.

Figura 34 - Botão de upload do código ao ESP32



Fonte: Autoria própria.

5.2 Configurando o WI-FI do ESP32

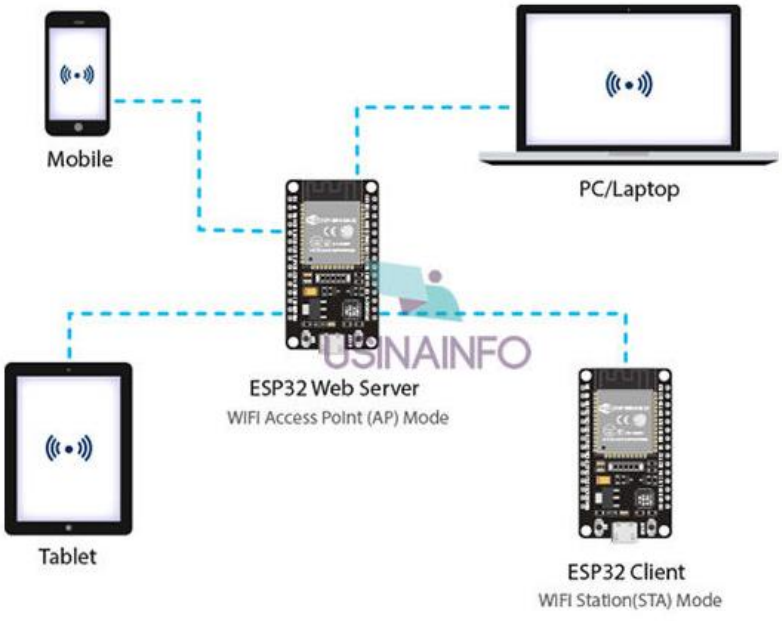
Um dos principais diferenciais do ESP32 é a sua capacidade de estabelecer conexão Wi-Fi, possibilitando a comunicação e a troca de dados com outros dispositivos que também possuam esse recurso.

Conforme suas especificações, o seu chip opera em uma frequência de 2,4 GHz, possuindo capacidade de alcançar taxas de transmissão de dados de até 150 Mbps. Além disso, esse microcontrolador oferece quatro modos de comunicação (interface) Wi-Fi, que podem ser configurados de acordo com a demanda do usuário: AP, STA, APSTA e NULL.

Para o modo null ou off, a estrutura de dados relativa ao funcionamento da rede não estará alocada no chip, desta forma, as rotinas do modo ap ou sta não serão inicializadas;

Para o modo AP, o Chip funcionará como um servidor/roteador, ou seja: o microcontrolador é configurado como fonte de wi-fi próprio, e qualquer dispositivo poderá conectar diretamente a ele, trocando dados entre sí[31]. Neste modo não é possível conectar a redes externas, como a internet. A Figura 35 ilustra o processo.

Figura 35 - Modo de acesso AP ESP32

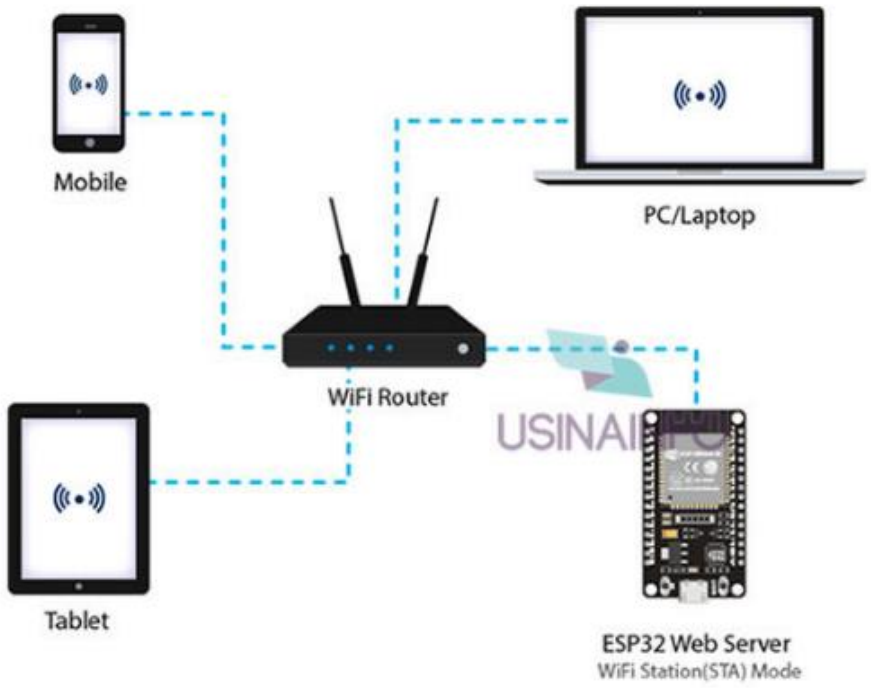


Fonte: [31]

Já no modo STA ou station mode, o dispositivo irá operar como um “cliente wireless”, ou seja: o ESP32 será um cliente de um dispositivo terceiro. Em outras palavras, o roteador ou outro dispositivo wi-fi fornecerá um ip único para o ESP32.

Desta maneira, os dispositivos presentem na mesma rede WI-FI poderão comunicar entre sí e ainda se conectarem a internet. A topologia em questão pode ser vista na Figura 36.

Figura 36 - Modo de acesso STA ESP32



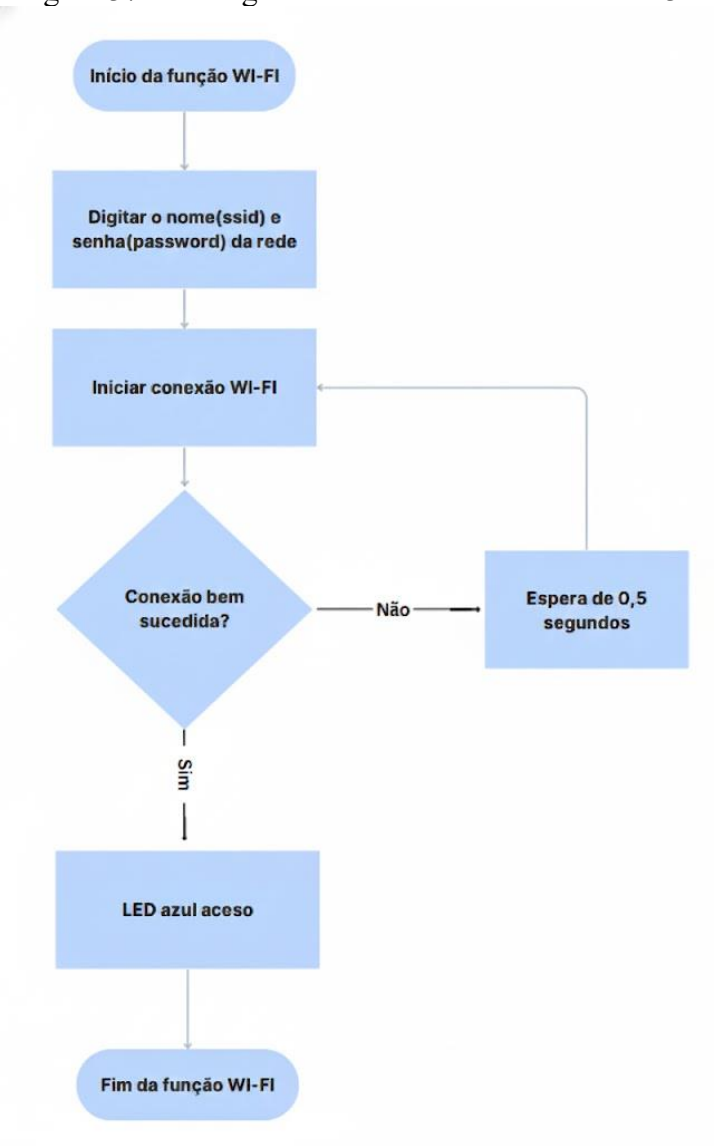
Fonte: [31]

Além disso, ainda existe o modo APSTA, que basicamente é um estado em que coexistem os modos STA E AP, no entanto, o ESP32 priorizará o moto STA ao invés do AP. Este modo não é recomendado devido a limitações de desempenho e confiabilidade – visto que o chip do ESP32 possui um único radio WI-FI que precisaria gerenciar duas tarefas simultaneamente [32].

Com isso, o modo utilizado é o STA (station), pois a conexão à internet por meio do RASPBERRY é importantíssima.

O código responsável pela conexão (comentada linha a linha) pode ser visto no apêndice A, além disso, o fluxograma que representa a lógica de conexão pode ser visto na Figura 37.

Figura 37 - Fluxograma da conexão WI-FI do ESP32



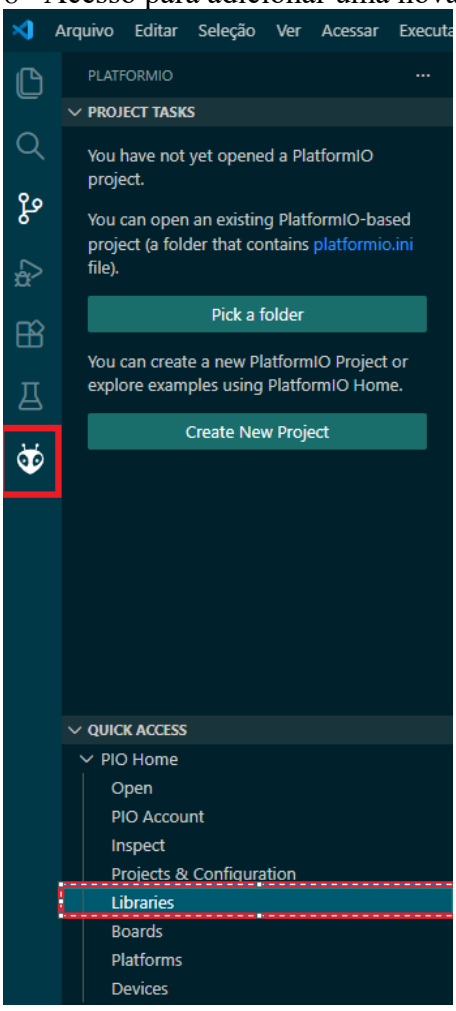
Fonte: Autoria própria

5.3 Configurando o broker no ESP32

Como já dito anteriormente, o ESP32, será o publicador de dois dados diferentes: a concentração de amônia e dióxido de carbono. Ambos em partícula por milhão (ppm), desta maneira, o broker Mosquitto (presente no Raspberry pi 3B) separará ambos em duas categorias diferentes, para que posteriormente seja montado o visualizador gráfico.

Para tal, primeiramente será necessário adicionar a biblioteca PubSubClient que realizará a conexão do ESP32 com o broker. Para que a biblioteca seja adicionada, primeiramente é necessário acessar o ícone PlatformIO (símbolo do alienígena), em seguida ir em “libraries”, que estará disponível na sub aba Quick Access, como visto na Figura 38:

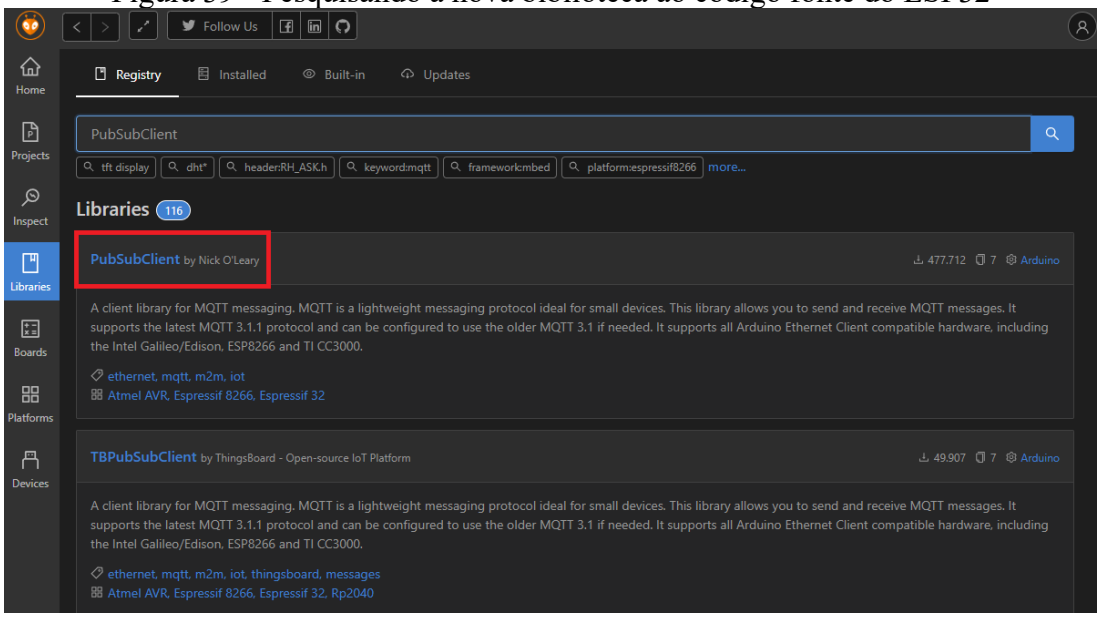
Figura 38 - Acesso para adicionar uma nova livraria



Fonte: Autoria própria

Após isso, uma nova aba se abrirá (este é a interface do PlatformIO), e a biblioteca “PubSubClient” deverá ser pesquisada na barra de pesquisa da aba “libraries”, como visto na Figura 39.

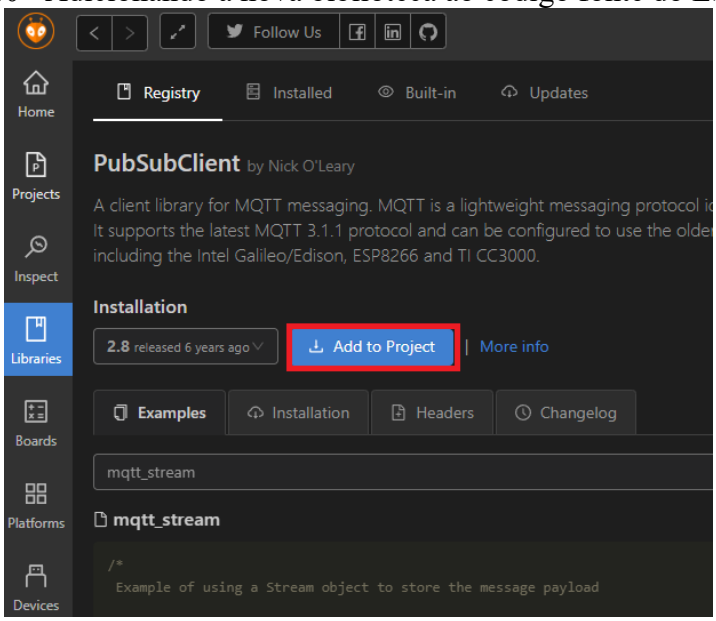
Figura 39 - Pesquisando a nova biblioteca ao código fonte do ESP32



Fonte: Autoria própria

Em seguida, a aba para adicionar a biblioteca poderá ser aberta clicando em “add to Project”, como mostra a Figura 40.

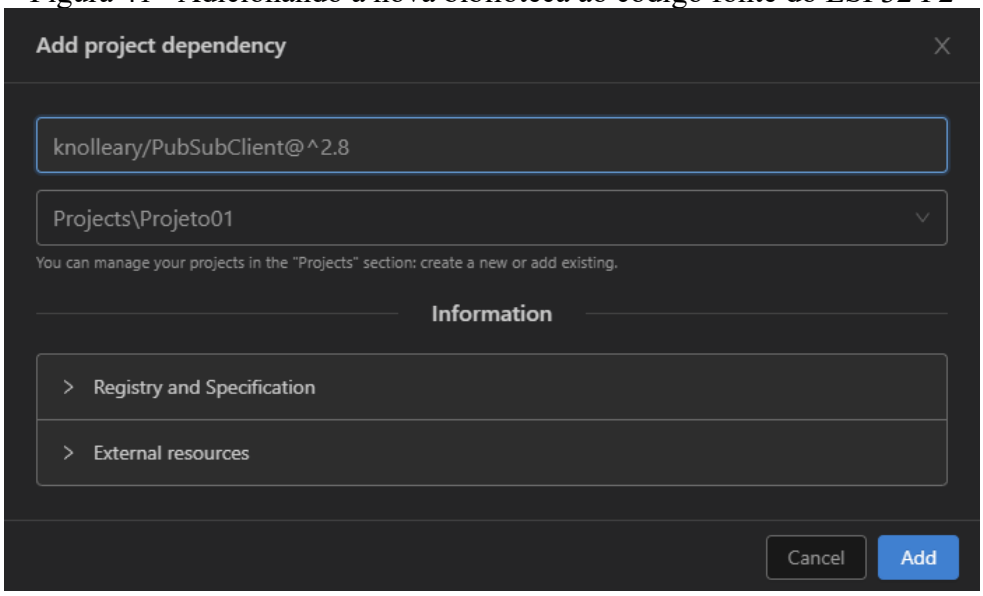
Figura 40 - Adicionando a nova biblioteca ao código fonte do ESP32 P1



Fonte: Autoria própria

Em sequência, o projeto anteriormente nomeado e editado deve ser selecionado. Neste caso, foi nomeado como “Projeto01”, e então, a biblioteca será adicionada ao código fonte do ESP32. Como visto na Figura 41:

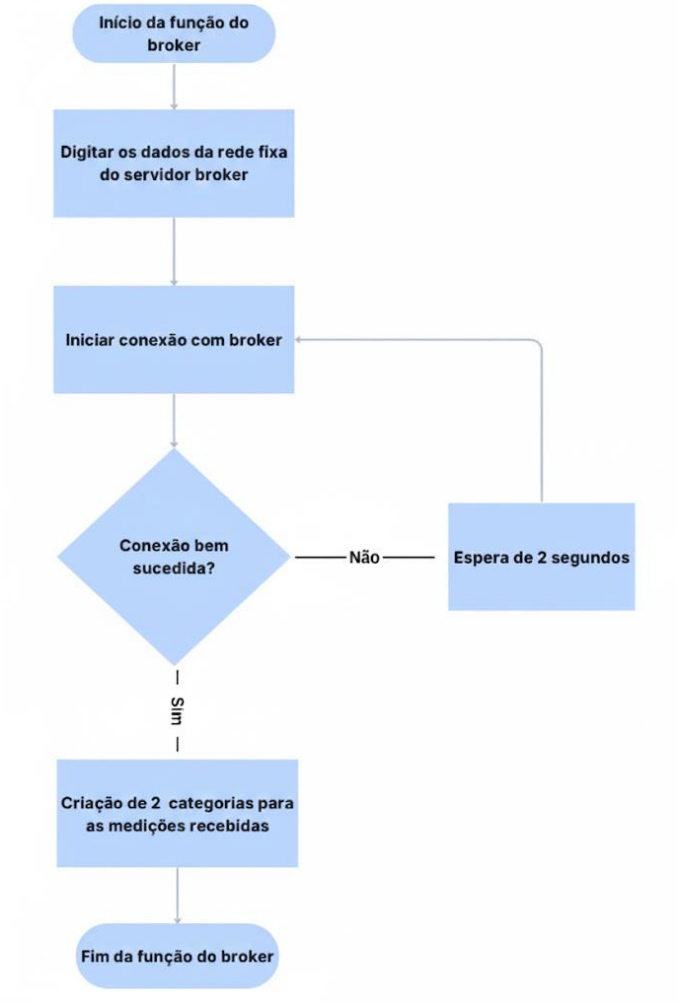
Figura 41 - Adicionando a nova biblioteca ao código fonte do ESP32 P2



Fonte: Autoria própria

Com isso, o código fonte para realizar a conexão com o broker poderá ser escrito e implementado ao ESP32. O algoritmo desenvolvido [33] foi bem parecido com a rotina da conexão WI-FI, a mesma pode ser visualizada na Figura 42 e estará presente comentada linha a linha no apêndice B. É importante observar as linhas relacionadas ao ao ip estático, que deve ser idêntico ao ip configurado (será explicado nos próximos tópicos) no Raspberry.

Figura 42 - Lógica da conexão entre o broker e ESP32



Fonte: Autoria própria

5.4 Aquisição dos dados no ESP32

Como o ESP32 faz a leitura das portas IO em tensão e posteriormente converte em valores digitais por meio de seu conversor interno de 12bits. É necessário desenvolver um algoritmo para converter as leituras dos sensores em valores uteis que serão enviados para o raspberry.

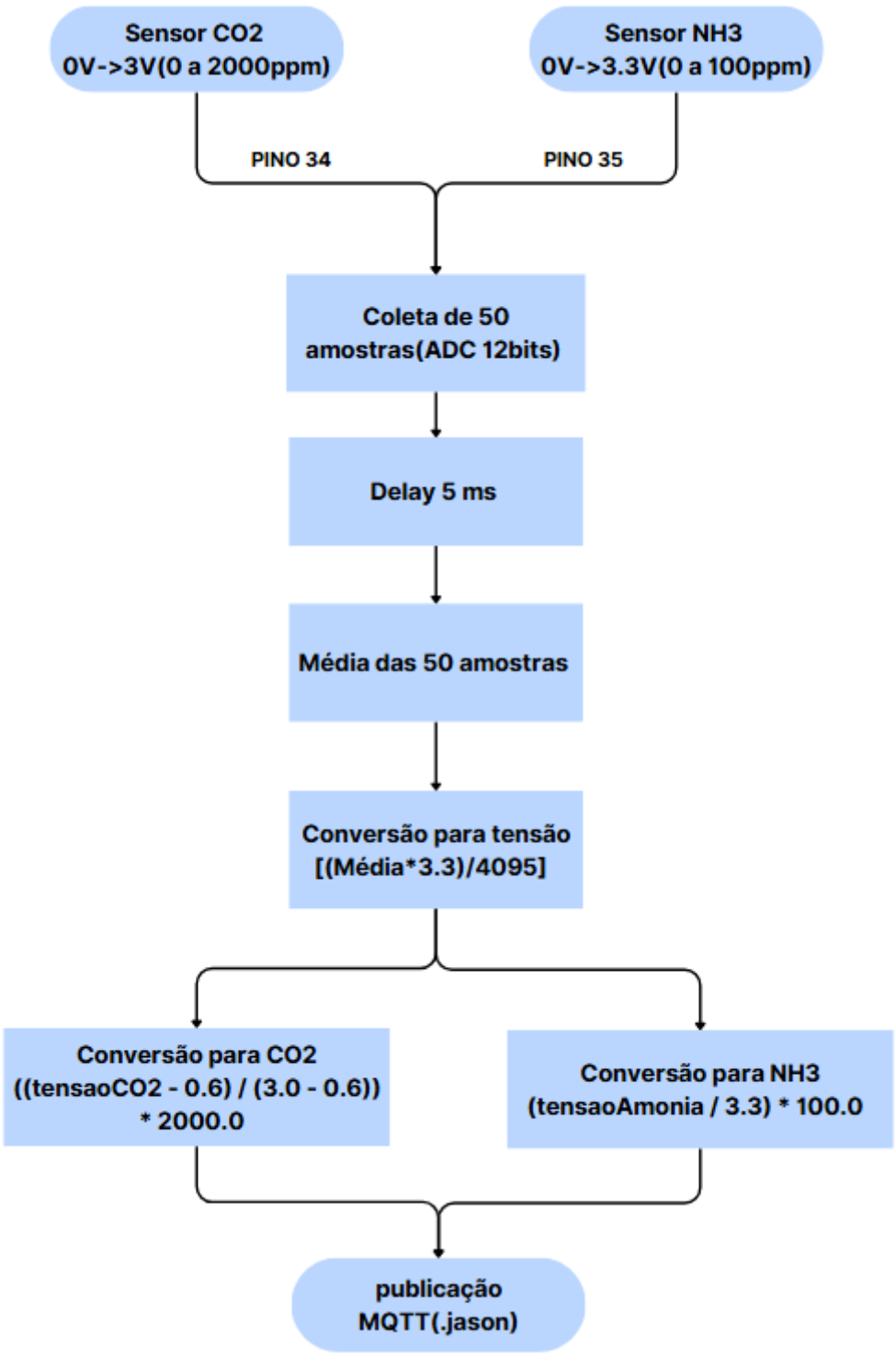
Inicialmente foi atribuído duas variáveis diferentes para coletar 50 amostras de cada sensor para posteriormente ser realizado a média aritmética entre elas. A ideia é diminuir o erro de leitura pela variação interna do ESP32. Em seguida, há a conversão para valores de tensão.

Para os valores de CO_2, a tensão correspondente é subtraída em 6 e posteriormente é dividida por (3-0.6) para então ser multiplicada por 2000(fundo de escala do sensor). Isto é feito pois como a saída do sensor de CO_2 é de 4 a 20 mA, um resistor shunt de 150 Ω é utilizado para convertê-la em uma tensão de 0,6 a 3,0 V. Como 0,6 V corresponde a 0 ppm e 3,0 V a 2000 ppm, a tensão medida pelo ESP32 é convertida para concentração por meio de uma interpolação linear. Inicialmente subtrai-se 0,6 V para deslocar a origem da escala, em seguida divide-se pelo intervalo útil de tensão (3,0 − 0,6 V) para normalizar a leitura e, por fim, multiplica-se pelo fundo de escala do sensor (2000 ppm), obtendo-se a concentração de dióxido de carbono.

Já para os valores de NH_3, a tensão correspondente é dividida por 3.3 e multiplicada por 100(fundo de escala do sensor).

Após isso, ambos os valores são armazenados em uma variável tipo .json (será explicada mais a frente) para serem enviados via MQTT para o raspberry. A Figura 43 mostra o fluxograma descrito. Além disso, o código completo estará disponível no apêndice C.

Figura 43 – Fluxograma da aquisição dos dados dos sensores



Fonte: Autoria própria

5.5 Broker Mosquitto no Raspberry Pi 3B

5.5.1 Instalação

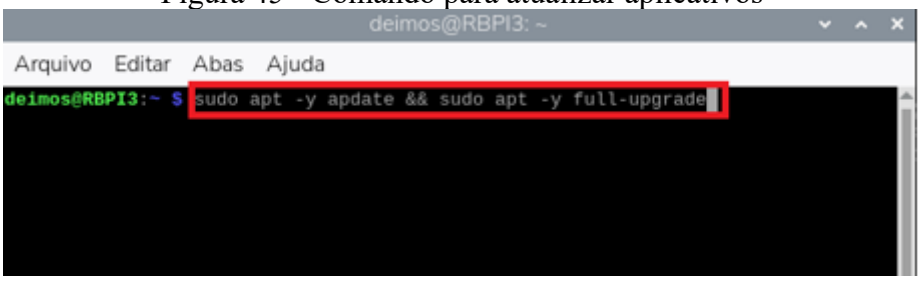
Antes de tudo, é importante verificar se há alguma atualização disponível para os aplicativos do Raspberry, para evitar qualquer conflito futuramente. Para tal, o terminal de comandos pode ser aberto a partir da barra de tarefas do S.O. do Raspberry, disponível no canto superior esquerdo. Como visto na Figura 44, também é possível atualizar por outros meios, porém este é mais prático.

Figura 44 - Localização do terminal de comandos do Raspberry

Fonte: Autoria própria

Após o terminal ser aberto, o comando “ sudo apt -y update && sudo apt -y full upgrade ” deve ser feito, como visto na Figura 45. O tempo de atualização dependerá da velocidade da internet e a quantidade de aplicativos desatualizados.

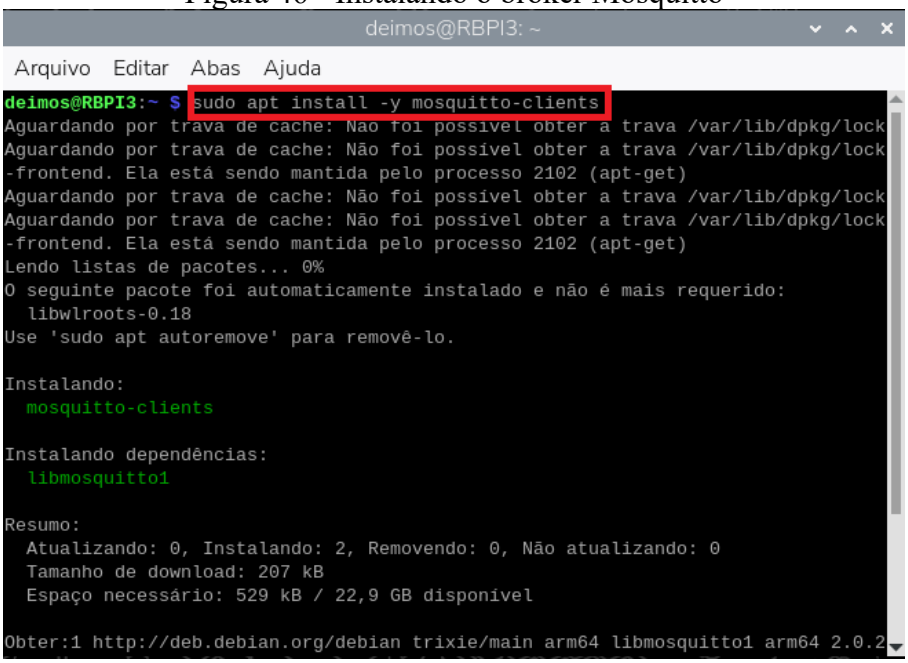
Figura 45 - Comando para atualizar aplicativos



Fonte: Autoria própria

Após a conclusão da atualização, o broker Mosquitto poderá ser instalado a partir do comando “ sudo apt install -y mosquitto mosquitto-clients “ como visto na Figura 46.

Figura 46 - Instalando o broker Mosquitto



Fonte: Autoria própria

Após a conclusão da atualização, o gerenciador de dados (envios e recebimentos de mensagens) estará pronto para ser configurado. No entanto, antes disso, é interessante inicializar o Mosquitto junto com o sistema operacional, pois caso aconteça algum problema de alimentação ou qualquer outro problema que force o

Raspberry Pi reiniciar, o broker também será iniciado junto. Evitando a necessidade de executar o mesmo manualmente.

Para tal, o comando “ sudo systemctl enable mosquitto.service ”, poderá ser digitado no terminal de comandos, como visto na figura a seguir:

Figura 47 - Inicializando o broker junto com o S.O.

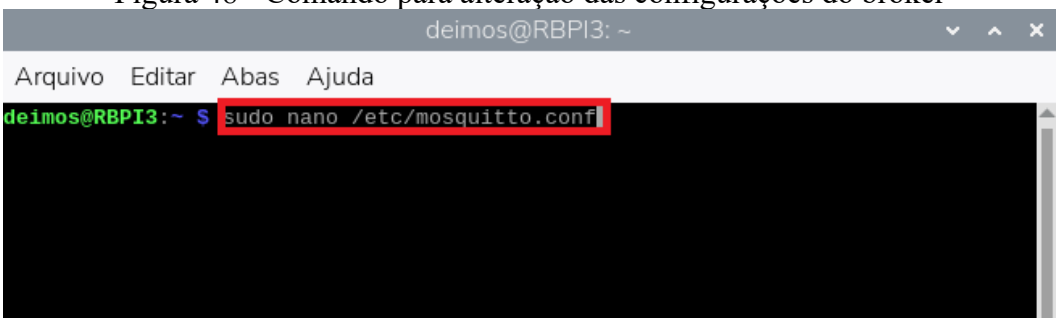
Fonte: Autoria própria

5.5.2 Configuração da segurança

Para que a comunicação da rede seja limitada apenas entre o raspberry e o ESP32, é necessário realizar algumas configurações.

Ao terminar de instalar o broker, ele já vem com algumas configurações pré-estabelecidas. Uma delas é a permissão de comunicação com dispositivos presentes no mesmo local de execução do broker. Para realizar a alteração desse e de qualquer parâmetro de configuração, o comando “ sudo nano /etc/mosquitto/mosquitto.conf “ deve ser digitado, como visto na Figura 48:

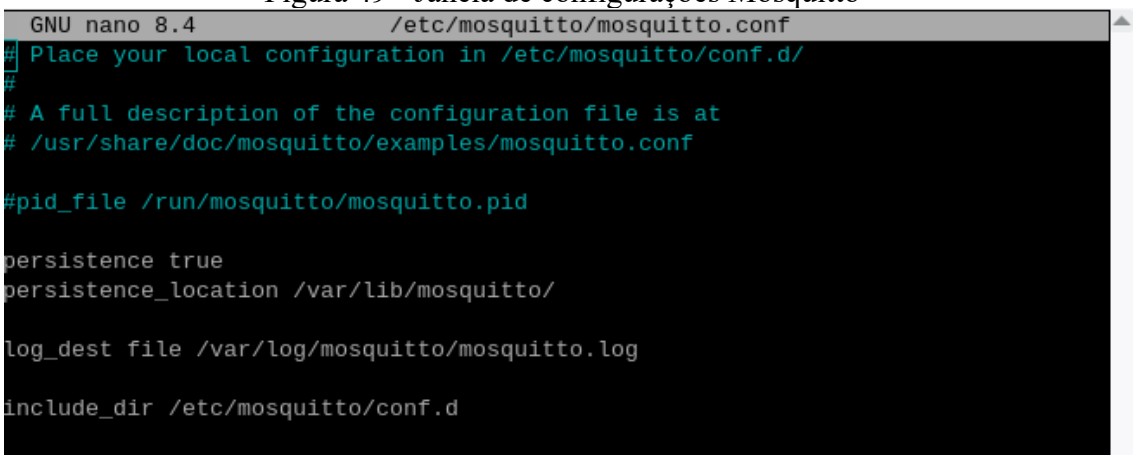
Figura 48 - Comando para alteração das configurações do broker



Fonte: Autoria própria

Em seguida, alguns comandos e opções aparecerão na janela:

Figura 49 - Janela de configurações Mosquitto

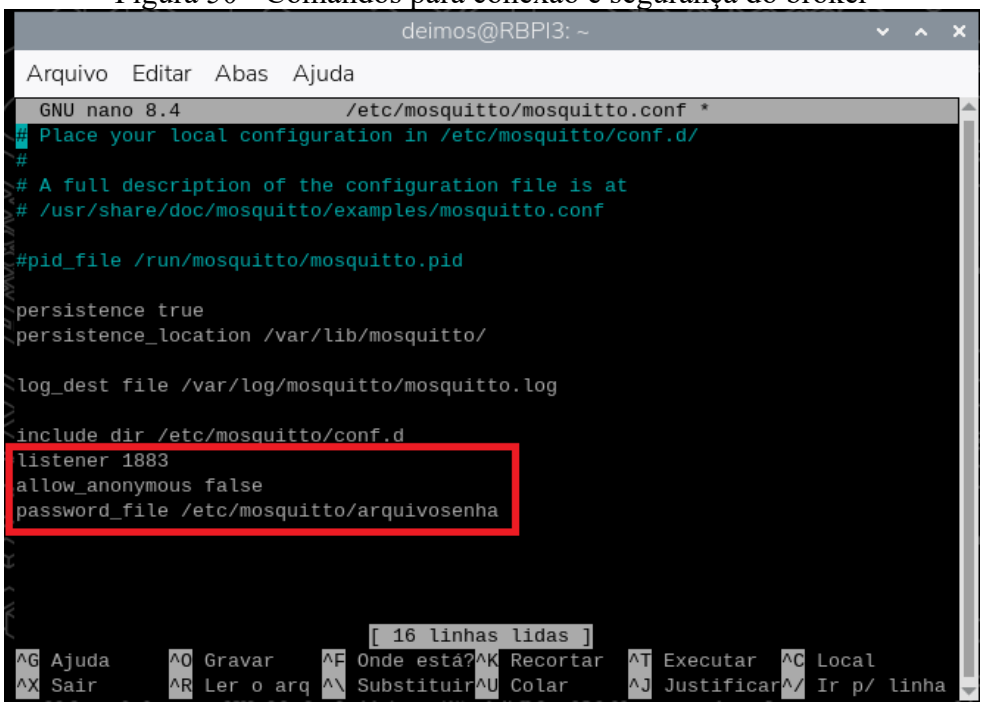


Fonte: Autoria própria

Com isto, o cursor de escrita deverá ser definido para a última linha, para que os três comando a seguir sejam escritos, um em cada linha a partir da última: 1° linha: “ listener 1883 ”, que definirá o canal de comunicação para a porta 83. 2º linha: “ allow_anonymous false “ , configuração que negará conexão com dispositivos não conhecidos/ autorizados. 3º linha: “ password_file /etc/mosquitto/arquivosenha “, este comando indicará onde o arquivo que contenha os dados de usuário e senha estará disponível para autorização de conexão com outros dispositivos que serão conectados ao broker. O nome do arquivo “arquivosenha” poderá ser definido a gosto. As respectivas linhas de comandos podem ser vistas na Figura 50.

Para salvar as configurações, o botão CTRL deve ser mantido pressionado, em seguida, a tecla X. Com isso a confirmação se dará com S ou Y (dependendo da língua escolhida), caso o usuário queira cancelar, basta apertar a tecla N. Para que as novas configurações sejam aplicadas, o sistema deve ser reiniciado.

Figura 50 - Comandos para conexão e segurança do broker



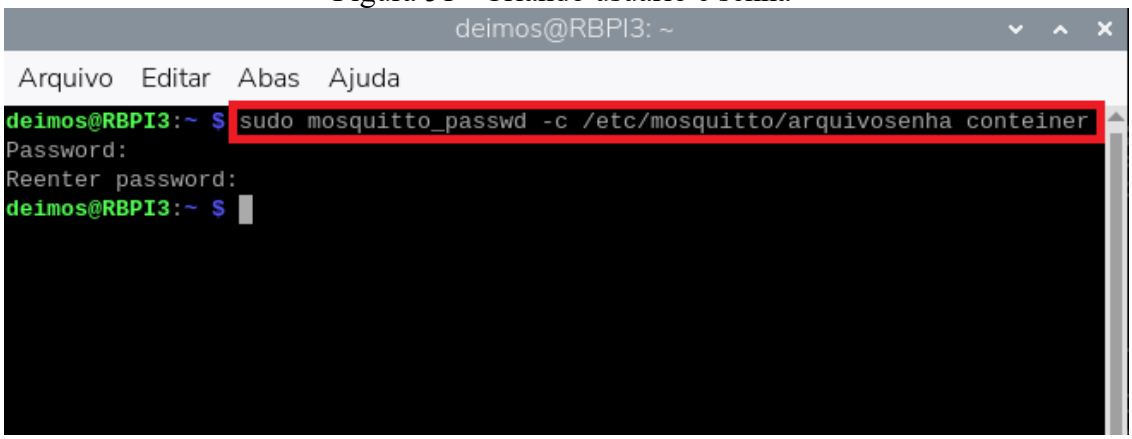
Fonte: Autoria própria

Para que nenhum dispositivo conectado a mesma rede do broker possa publicar dados sem permissão, a criação de um usuário e senha deve ser definido (como dito na configuração do ESP32).

Para isso, o comando “ sudo mosquitto_passwd -c /etc/mosquitto/arquivosenha conteiner “ deve ser digitado. No lugar de “conteiner”, o nome desejado deve ser digitado. Após ser pressionado o enter, a senha a ser definida será pedida. Digite-a, confirme e digite-a novamente para confirmar. Caso a senha digitada para definição esteja coerente (ambas iguais), o usuário e senha será criado.

É importante notar que neste passo, a senha digitada não aparecerá no terminal de comando. A Figura 51 mostra todos os comandos:

Figura 51 - Criando usuário e senha



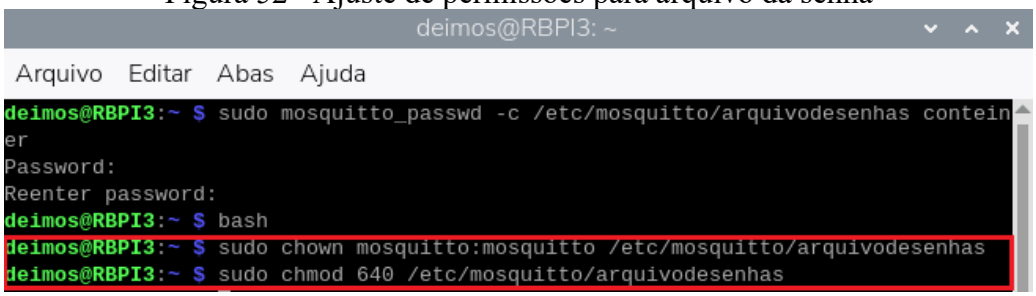
Fonte: Autoria própria

Pode-se criar a quantidade de usuários que for necessário, incluindo a definição de permissão de publicações ou assinaturas de cada um. Para este caso, a credencial padrão será tanto para publicação, quanto para assinatura.

Para definir credenciais especificas, o comando “ sudo nano /etc/mosquitto/aclfile “ pode ser inserido, de forma a abrir uma nova aba de configuração. Então, a credencial do usuário poderá ser definida a partir de duas opções: “write” para publicar e “read” para assinar. Caso o usuário “contêiner” fosse apenas para assinatura, neste momento o comando “ user conteiner” e “ topic read sensores “ deveria ser digitado e confirmado. Como este não é o caso, a credencial para assinatura e publicação continuará.

Após este passo, os seguintes comandos devem ser inseridos no terminal: “ sudo chown mosquitto:mosquitto /etc/mosquitto/ arquivodesenha” e “ sudo chmod 640 /etc/mosquitto/arquivodesenha”. Eles garantem o ajunte de permissão do arquivo de senha para evitar qualquer problema futuro.

Figura 52 - Ajuste de permissões para arquivo da senha

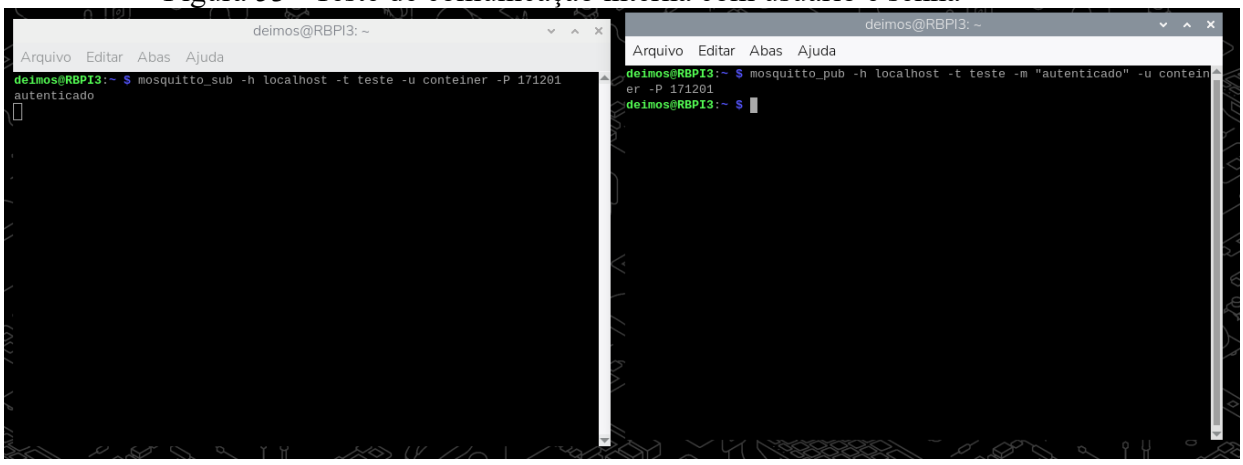


Fonte: Autoria própria

Já há a possibilidade de testar o envio e o recebimento de mensagens para confirmar se o Mosquitto está respeitando os dados de usuário e senha. Para isso, abra dois terminais de comando. No primeiro (este será o assinante) digite o código: “ mosquitto_sub -h localhost -t teste -u usuario -P senha “, sendo que o usuário e senha devem ser substituídos. No segundo terminal(publicador), o código deve ser inserido: “ mosquitto_pub -h localhost -t teste -m "autenticado" -u usuario -P senha “. Caso o primeiro terminal receba a mensagem “autenticado” então está ok.

O mesmo teste pode ser repetido, só que dessa vez, colocando o usuário ou senha incorreta, onde a mensagem não pode deve ser transmitida. A Figura 53 mostra isso.

Figura 53 - Teste de comunicação interna com usuário e senha



Fonte: Autoria própria

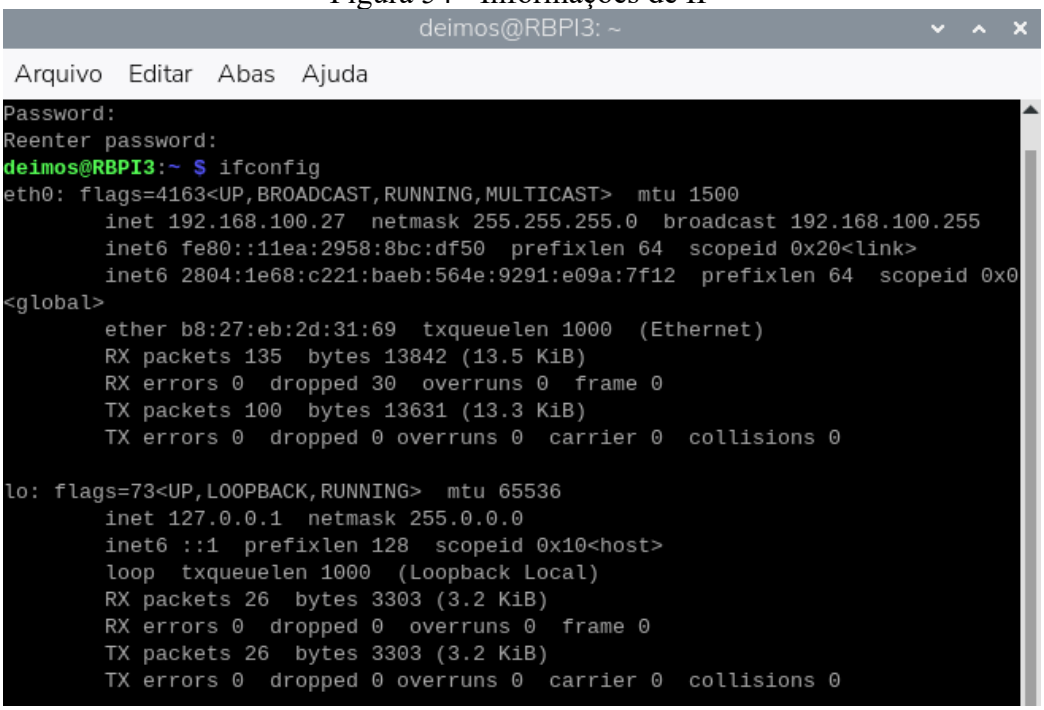
5.5.3 Configuração de rede

Para que outros dispositivos possam acessar o Raspberry Pi repetidamente, sem problemas de conexão via Wi-Fi, é necessário configurar um endereço IP fixo para ele. Isso porque, sempre que um dispositivo se conecta a um roteador Wi-Fi, é atribuído a ele um endereço IP com parte final variável. Esse endereço funciona como uma identificação única do dispositivo na rede.

No contexto desta aplicação, entretanto, essa característica não é desejável. Caso o Raspberry Pi permaneça com um endereço IP dinâmico, sempre que houver a conexão de um novo dispositivo à rede ou ocorrer uma interrupção no fornecimento de energia, por exemplo, os endereços IP dos dispositivos poderão ser alterados. Como consequência, será necessário reconfigurar no ESP32 o endereço IP utilizado para localizar o Raspberry Pi, essa situação torna o sistema pouco prático e reduz sua confiabilidade.

Para evitar tal ocorrência, é preciso digitar “ifconfig” no menu de comandos, para que as interfaces de rede do raspberry sejam exibidas. Em seguida aparecerão três grupos de dados: o eth0 diz respeito as configurações de rede quando o rapsberry está conectado a rede via cabo, lo para loopback (ip virtual para a máquina comunicar consigo mesmo, dando retorno a sua própria trilha TCP/IP) e wlan0, para quando o dispositivo estiver conectada via WI-FI. Tais informações podem ser vistas na Figura 54.

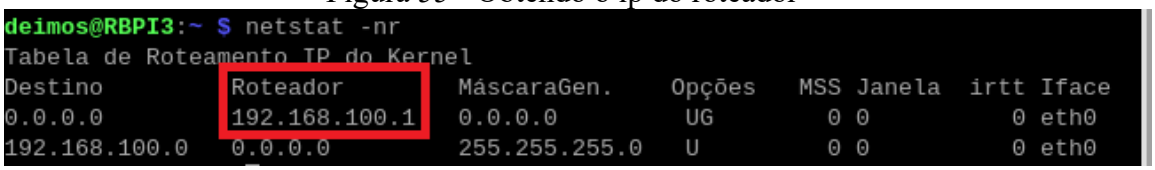
Figura 54 - Informações de IP



Fonte: Autoria própria

No caso acima, o raspberry está conectado a rede via cabo, logo o ip a ser fixo(inet) será o 192.168.100.27; o final .27 é o numeral que identifica o dispositivo na rede. Além disso, também será necessário saber o ip que identifica o roteador. Para isso, basta digitar o comando “netstat -nr”, uma nova aba de informações será aberta e o ip do roteador estará disponível, como visto na Figura 55:

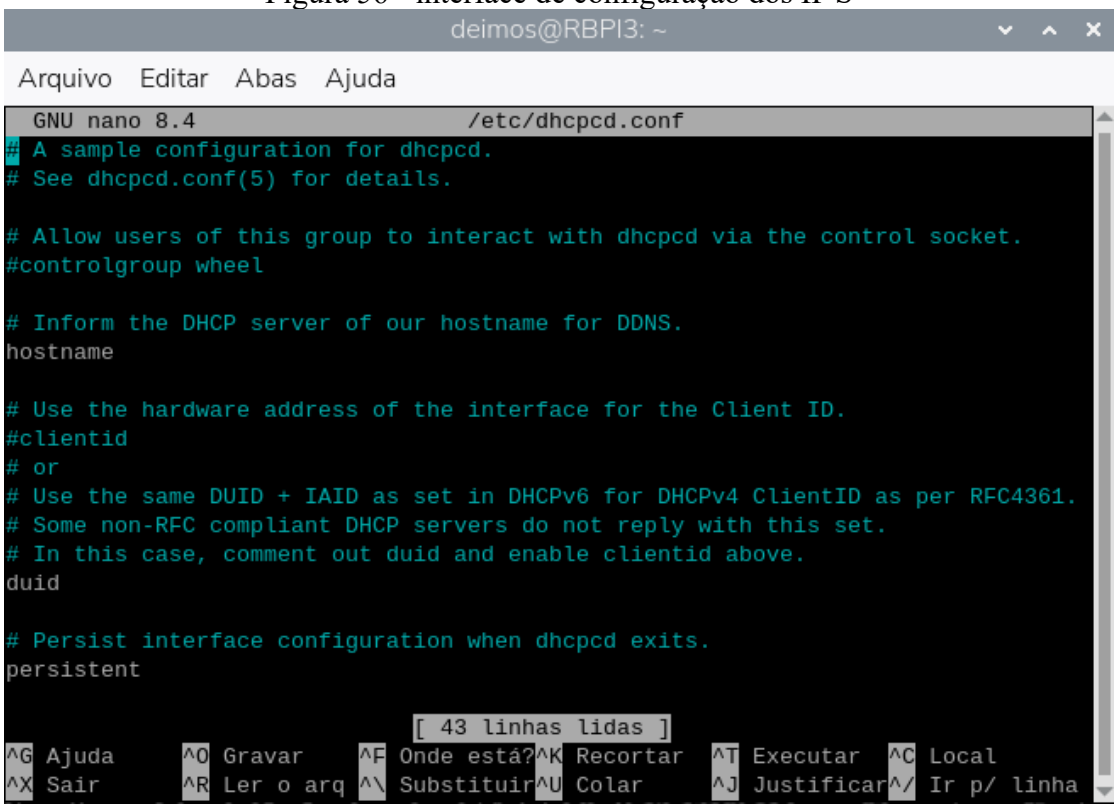
Figura 55 - Obtendo o ip do roteador



Fonte: Autoria própria

Para fixar ambos os ip’s no raspberry, é necessário digitar o comando “ sudo nano /etc/dhcpcd.conf ” para que a aba de configurações da interface de rede seja aberta. Como visto na Figura 56:

Figura 56 - interface de configuração dos IP'S



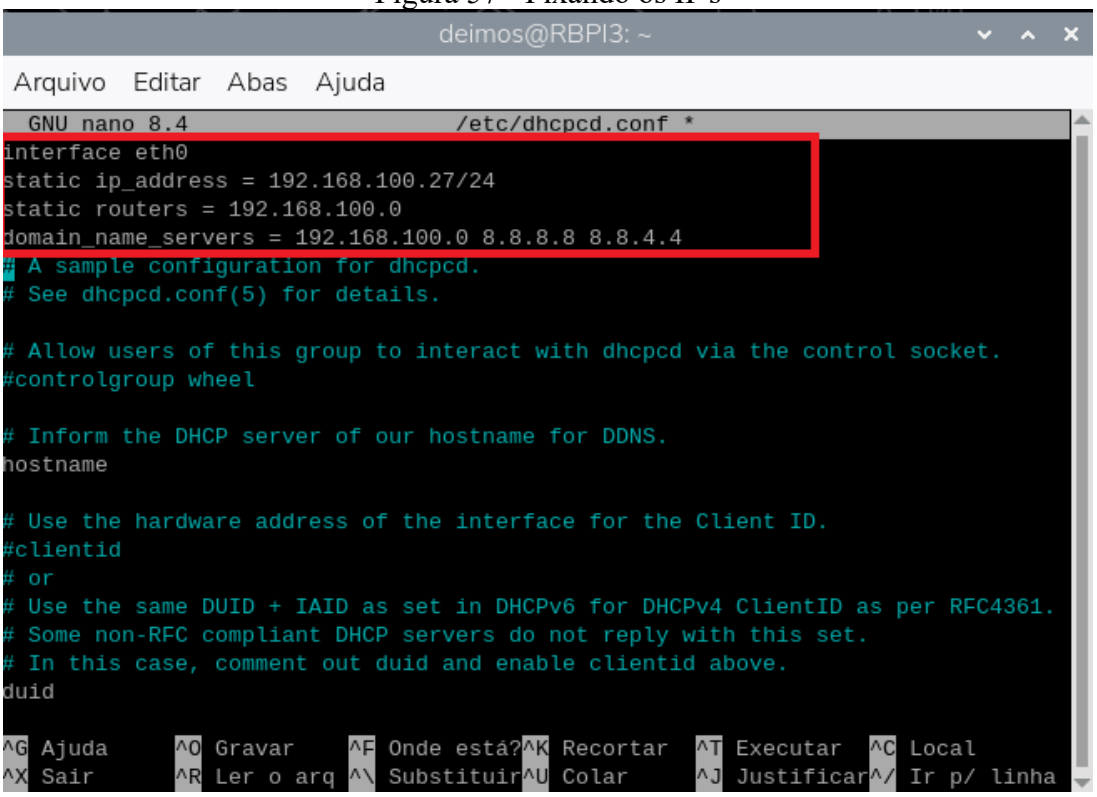
Fonte: Autoria própria

Em seguida, o cursor de digitação deve ser definido para a primeira linha, para que as quatro linhas de comando possam ser digitadas: 1º linha diz respeito ao tipo de conexão (cabo ou wi-fi), “ interface eth0 ” para cabo de rede ou “ interface wlan0 ” para conexão sem fio.

Para a 2º linha, o comando a ser digitado diz respeito ao ip que identifica o raspberry, ou seja: “ static ip_address = 192.168.100.27/24 ” o /24 é um notação CIDR própria do raspberry. Já para a 3º linha, o comando “static routers = 192.168.100.1” diz respeito ao ip a ser fixo do roteador.

Por fim, na quarta linha será fixado o endereço DNS do roteador e outros dois externos relacionados ao google (não sendo necessário [23]), com isso o comando a ser digitado será o “ domain_name_servers = 192.168.100.1 8.8.8.8 8.8.4.4 “ . Todo o passo a passo pode ser visualizado na Figura 57.

Figura 57 - Fixando os IP's



Fonte: Autoria própria

Para salvar as configurações, o botão CTRL deve ser mantido pressionado, em seguida, a tecla X. Com isso a confirmação se dará com S ou Y (dependendo da língua escolhida), caso o usuário queira cancelar, basta apertar a tecla N. Para que as novas configurações sejam aplicadas, o sistema deve ser reiniciado.

A partir de agora, o IP que identifica o rapsberry na rede sempre serão fixos, independente de reinicialização, quedas de energia, aumento ou diminuição de equipamentos no roteador. Outra maneira de fixar um IP de um dispositivo é pela interface do próprio roteador, no entanto é um processo especifico, visto que cada fabricante possui sua interface e modo de alteração.

Outro aspecto que deve ser levado em consideração é a alteração de rede, por exemplo: caso o sistema seja mudado de rede, ou que a própria rede altere de fornecedor, todo o processo de fixação de ip deve ser repetido.

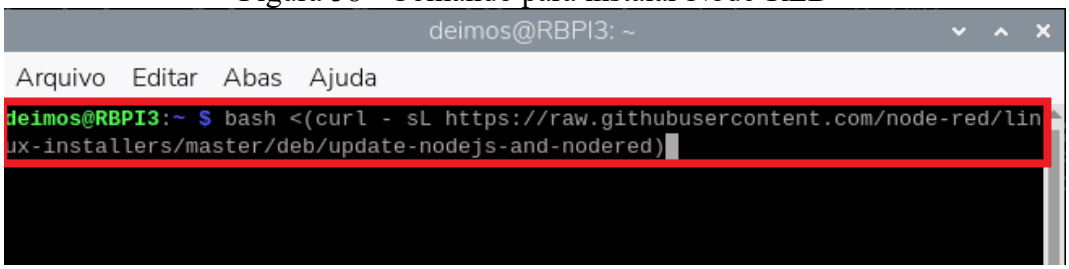
5.6 Node-RED no Raspberry Pi 3B

Para o gerenciamento e armazenamento dos dados além da criação de uma interface gráfica, o Node-Red servirá muito bem. Com isto, o mesmo atuará como um assinante, de forma a apenas receber os dados provenientes do ESP32 e o disponibilizando em uma interface em tempo real... Além de armazena-los localmente.

5.6.1 Instalação do Node-RED

Para instalar o Node-RED, o código " bash <(curl -sL https://raw.githubusercontent.com/node-red/linux-installers/master/deb/update-nodejs-and-nodered) " deve ser inserido no central de comandos (Figura 58) . Após dar o enter, a interface perguntará se a extensão node.js poderá ser instalada junto, deve-se confirmar apertando a tecla y ou s (dependendo da língua utilizada).

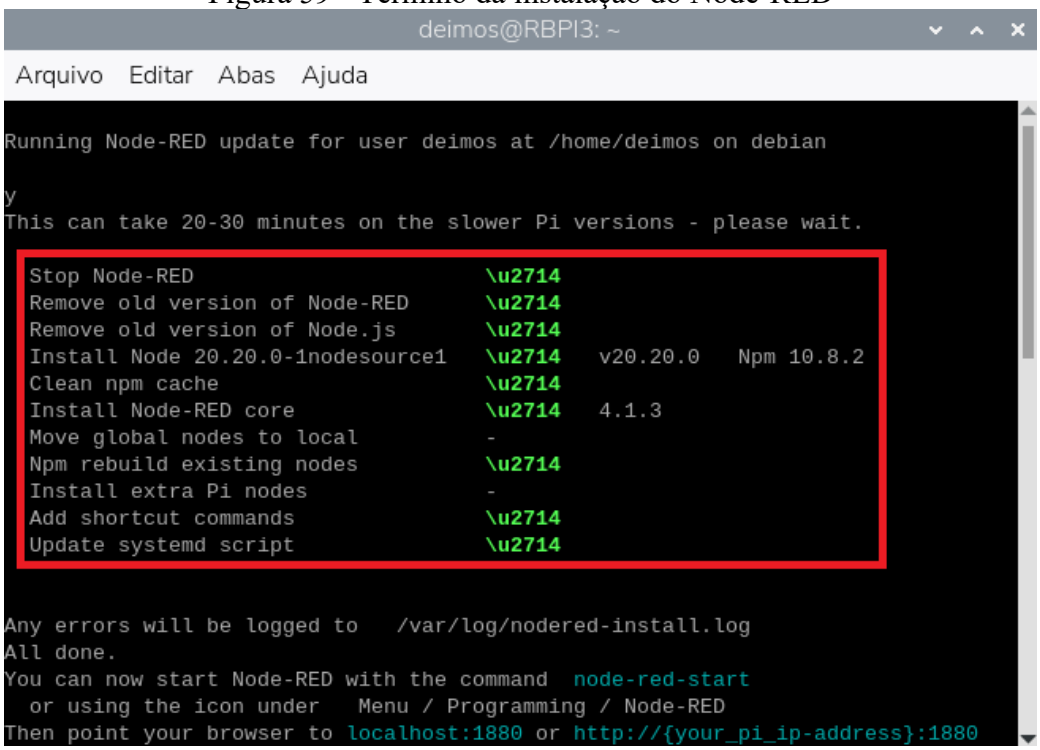
Figura 58 - Comando para instalar Node-RED



Fonte: Autoria própria

O tempo de instalação vai depender tanto da velocidade do microSD utilizado pelo raspberry, quanto pela velocidade da conexão. Após o término da instalação, as seguintes confirmações devem estar de acordo:

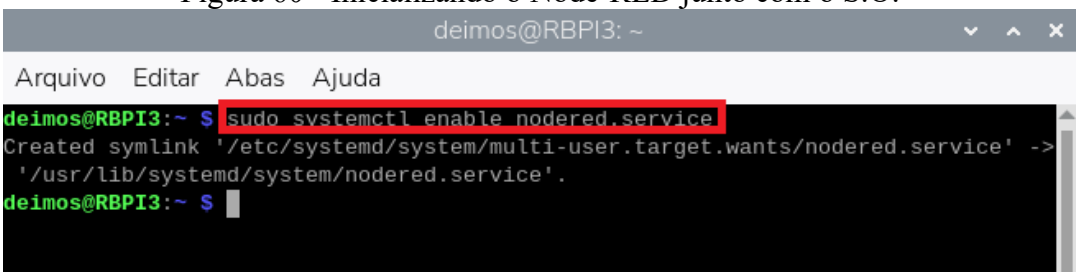
Figura 59 - Término da instalação do Node-RED



Fonte: Autoria própria

Após a instalação, é conveniente que o Node-RED inicialize junto com o S.O., seguindo a mesma lógica da inicialização automática do Broker Mosquitto. Para tal, o comando “ sudo systemctl enable nodered.service “ deve ser inserido na central de comandos, como visto na Figura 60. Após isso, é recomendado reiniciar o sistema.

Figura 60 - Inicializando o Node-RED junto com o S.O.

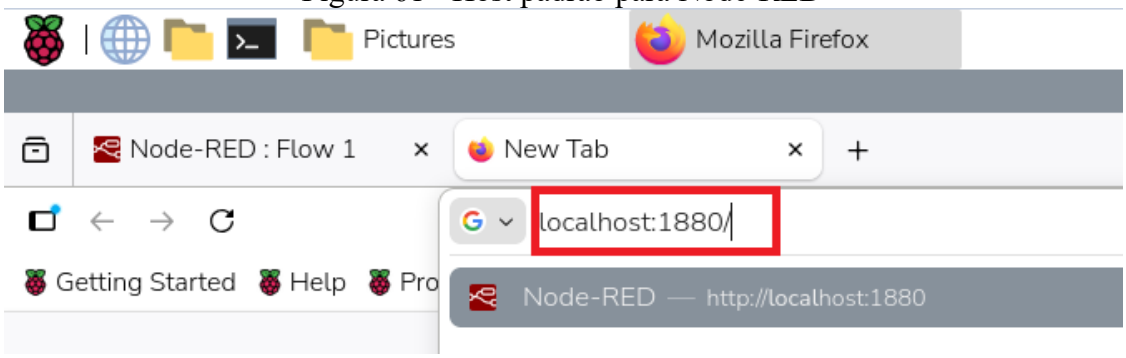


Fonte: Autoria própria

Como dito anteriormente, o Node-RED é baseado em navegador, com isso, para acessar a interface de edição e criação dos nodes, é necessário abrir um

navegador e digitar “ localhost:1880 “,como visto na Figura 61. Este endereço localiza a interface do Node-RED. É recomendado utilizar o Firefox no Raspberry PI 3B por ser mais leve para o dispositivo, evitando travamentos durante o uso.

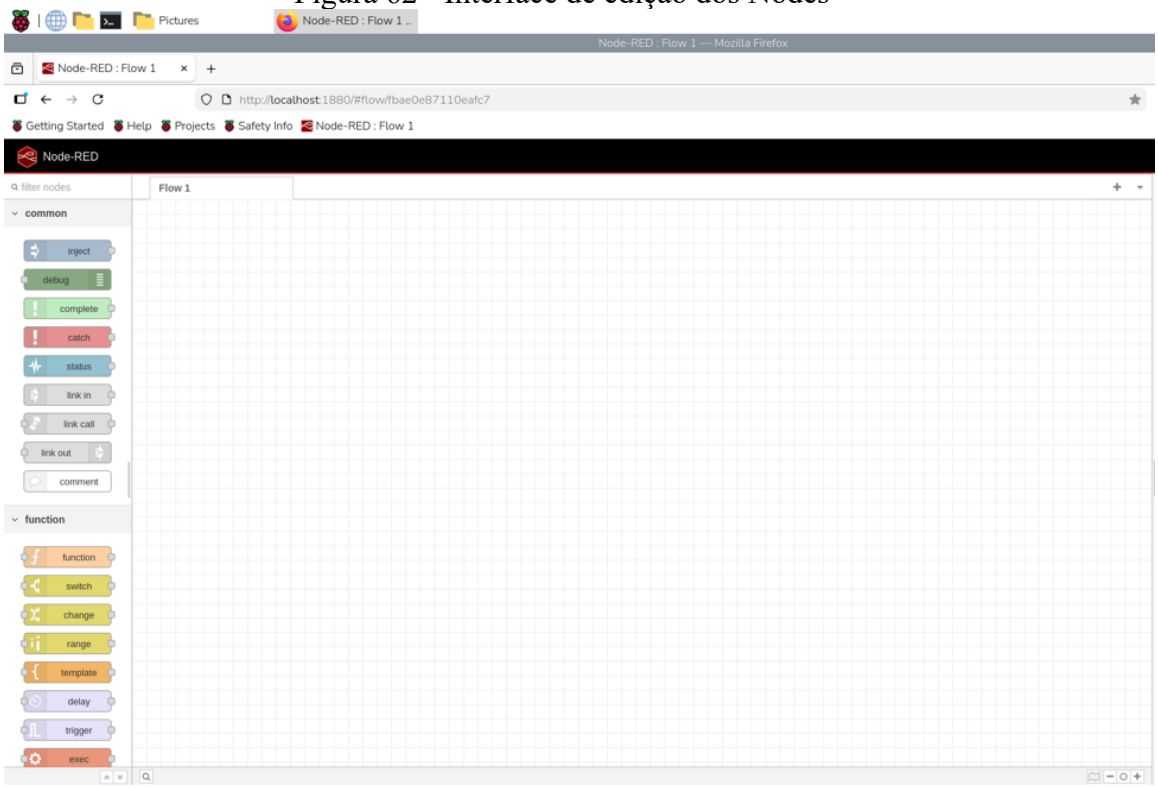
Figura 61 - Host padrão para Node-RED



Fonte: Autoria própria

Após o carregamento, a pagina onde será criado e editado os nodes, irá aparecer (Figura 62). Nota-se que qualquer pessoa presente na rede poderá criar e alterar os projetos, visto que não possui segurança nenhuma. Problema este, que será resolvido a seguir.

Figura 62 - Interface de edição dos Nodes



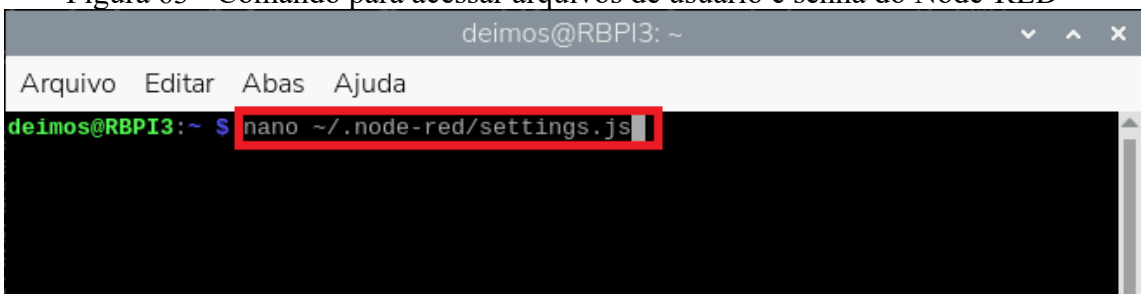
Fonte: Autoria própria

5.6.2 Criação de usuário e senha para acessar os nodes

Com a criação de credenciais para usuários, o nível de acesso nos nodes poderão ser modificados. Sendo que toda vez que o host padrão do Node-RED for aberto por qualquer dispositivo presente na rede, será pedido um usuário e senha para ganhar acesso ao mesmo.

Para acessar o arquivo onde o usuário e senha será criado. O comando “ nano ~/.node-red/settings.js “ deve ser digitado no central de comandos, como visto na Figura 63.

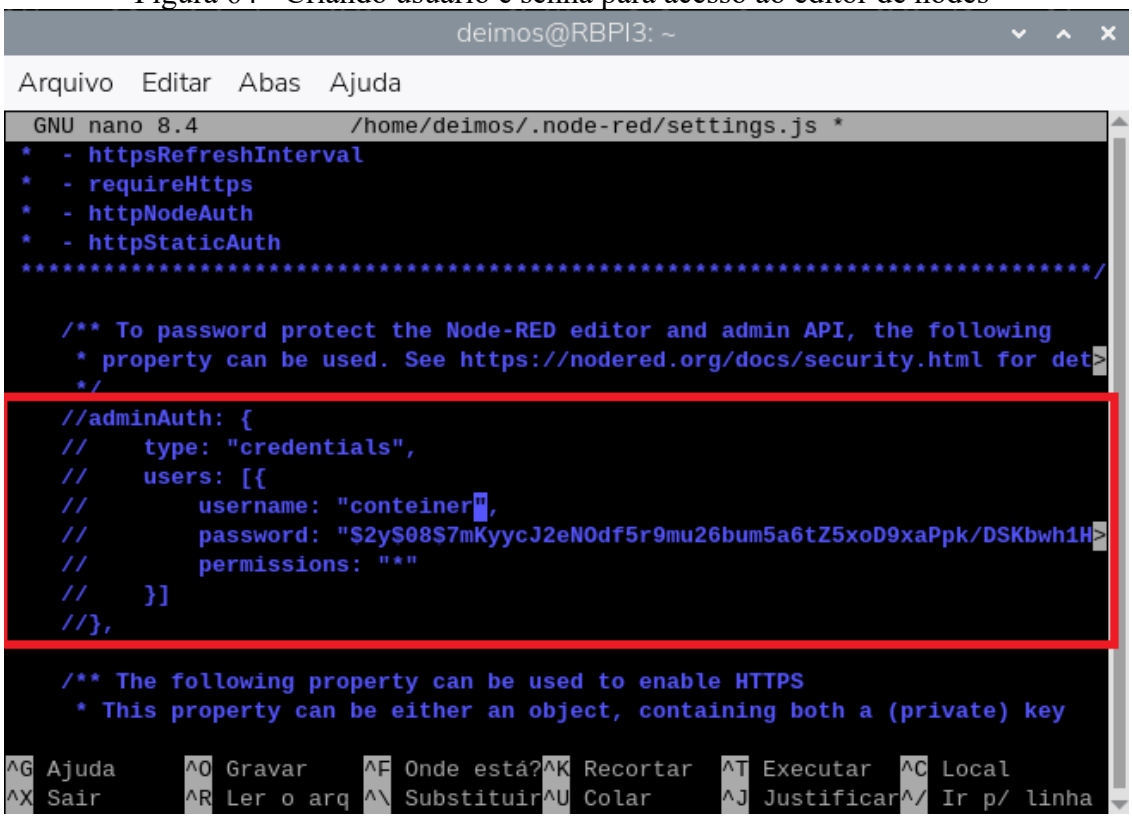
Figura 63 - Comando para acessar arquivos de usuário e senha do Node-RED



Fonte: Autoria própria

Em seguida, uma série de informações presente nesta pasta aparecerá na aba. Deve-se rolar o cursor para baixo até que o conjunto referente ao usuário e senha, denominado como “adminAuth”, apareça. Como visto na Figura 64:

Figura 64 - Criando usuário e senha para acesso ao editor de nodes



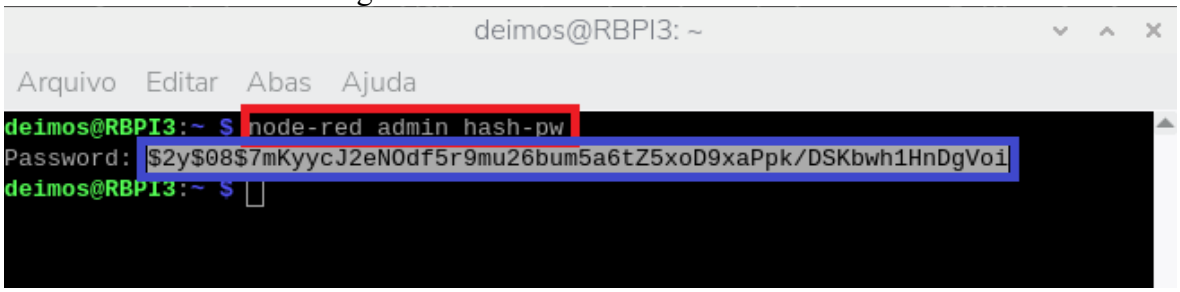
Fonte: Autoria própria

O nome do usuário deve ser substituído logo a frente de “username”, na Figura 64 o usuário foi denominado como “contêiner “. Nota-se que no espaço referente a

senha, há uma série de dígitos nada convenientes. Esta sequência se chama hash, muito utilizada para geração de senhas... É como se fosse uma “criptografia” para aumentar a segurança do dispositivo. O Node-RED utiliza esse protocolo (também conhecido como bcrypt), que comparará ao pré-definido na hora de acessar. A ideia é ser lento e confiável para acessar.

Para geração do hash de senha escolhido, o usuário deverá digitar o comando “ node-red-admin hash pw “, e após dar enter, será pedido a senha para conversão no padrão hash. É importante salientar que a senha digitada não aparecerá na tela por padrão. A Figura 65 mostra o comando e a senha gerada em hash após a conclusão:

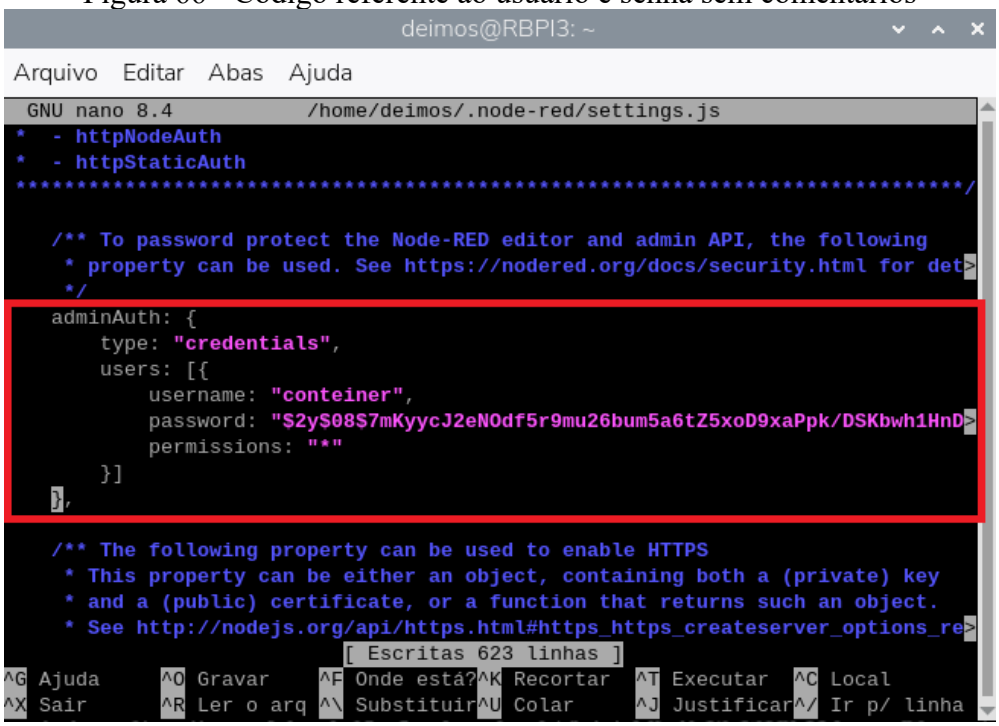
Figura 65 - Criando uma senha hash



Fonte: Autoria própria

Após a senha ser gerada em hash, a mesma deverá ser copiada e substituída no local da antiga padrão (logo a frente em password, Figura 64). Após isso, todos os comentários (//) relacionados a aba “adminAuth” deverá ser retirada para que o código seja compilado pelo Node-RED, para então poder ser pressionado a tecla Ctrl + O para gravar e em seguida, enter para confirmar. A Figura 66 mostra como o código ficará após a retirada dos comentários.

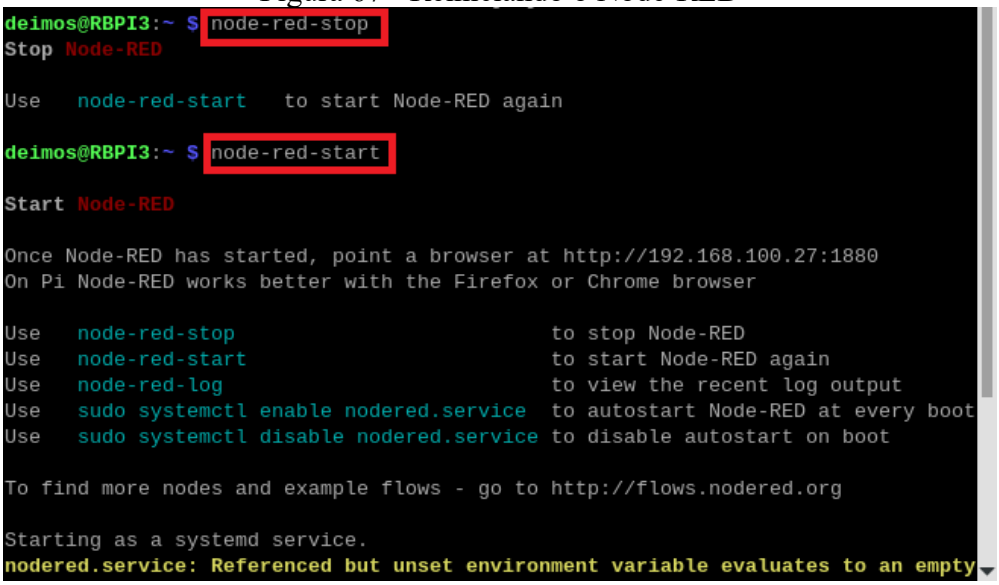
Figura 66 - Código referente ao usuário e senha sem comentários



Fonte: Autoria própria

Com estes passos feitos, é necessário reiniciar a aplicação do Node-RED. Para tal, o comando “ node-red-stop “ deverá ser colocado para pausar a aplicação. Após alguns segundos, coloque o comando “ node-red-start “ para inicia-lo novamente. Como visto na Figura 67.

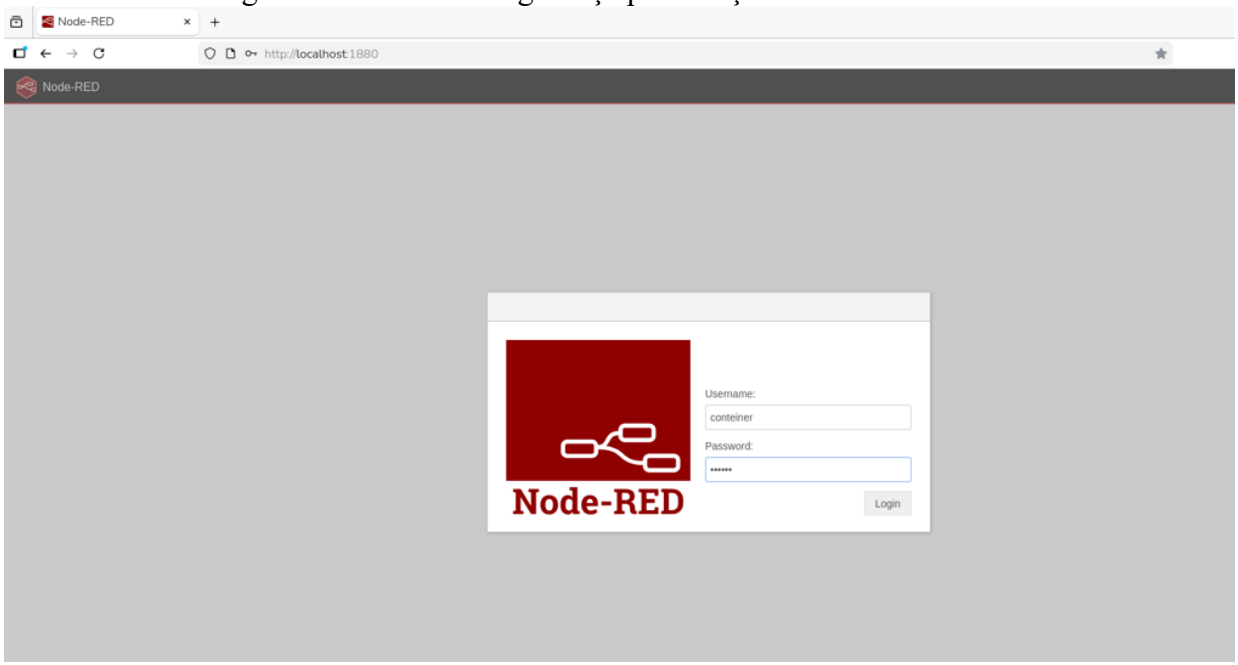
Figura 67 - Reiniciando o Node-RED



Fonte: Autoria própria

Após o término deste passo a passo, toda que vez que algum dispositivo tentar acessar o host de edição do Node-RED, aparecerá uma janela pedindo usuário e senha para acessa-lo. Como mostra a Figura 68.

Figura 68 - Janela de segurança para edição dos nodes

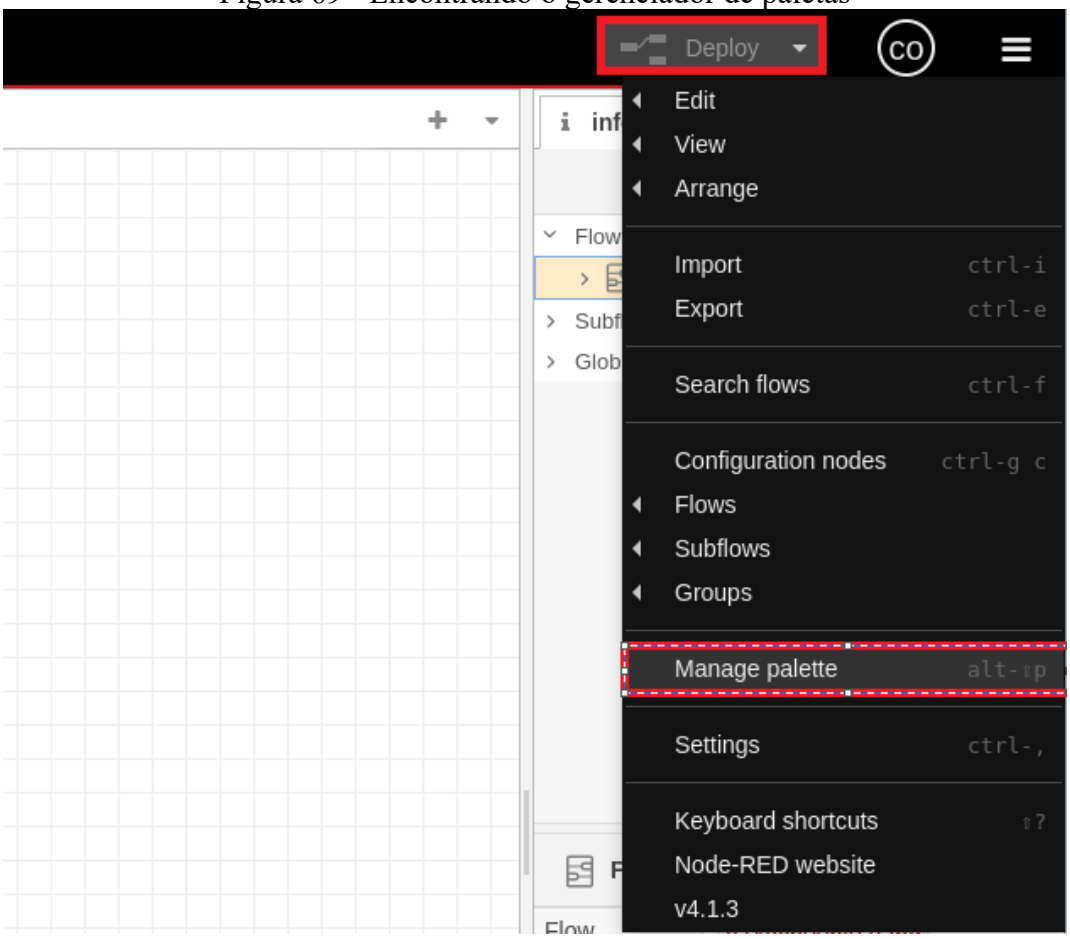


Fonte: Autoria própria

5.6.3 Instalação de Nodes extras

Ao acessar a barra lateral direita da interface de desenvolvimento do Node-RED, será possível encontrar o gerenciador de paletas, que estará disponível logo no canto superior. Em seguida, uma aba de várias opções aparecerá, clicando em “manage palette” será possível abrir o gerenciador. Outro modo de abertura, é pelo atalho alt+p.

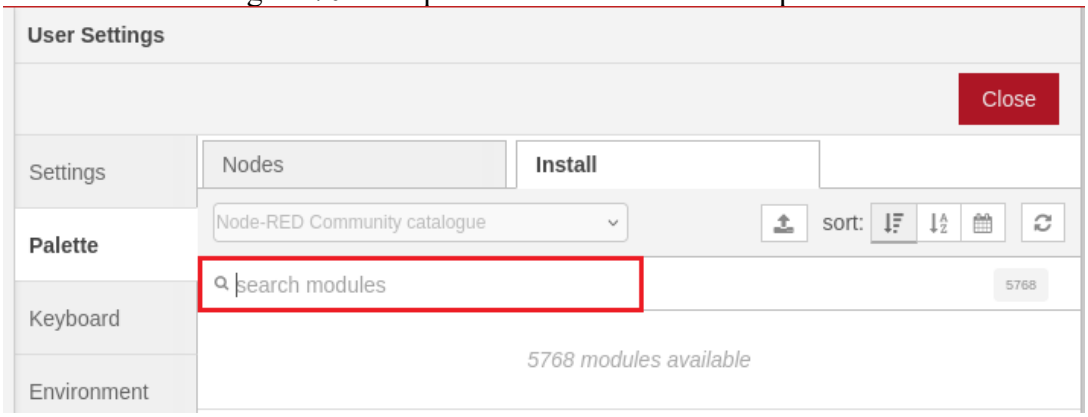
Figura 69 - Encontrando o gerenciador de paletas



Fonte: Autoria própria

Ao abrir, o motor de buscas poderá ser acessado em “install”. Como visto na Figura 70:

Figura 70 - Pesquisando novos modulos de paletas



Fonte: Autoria própria

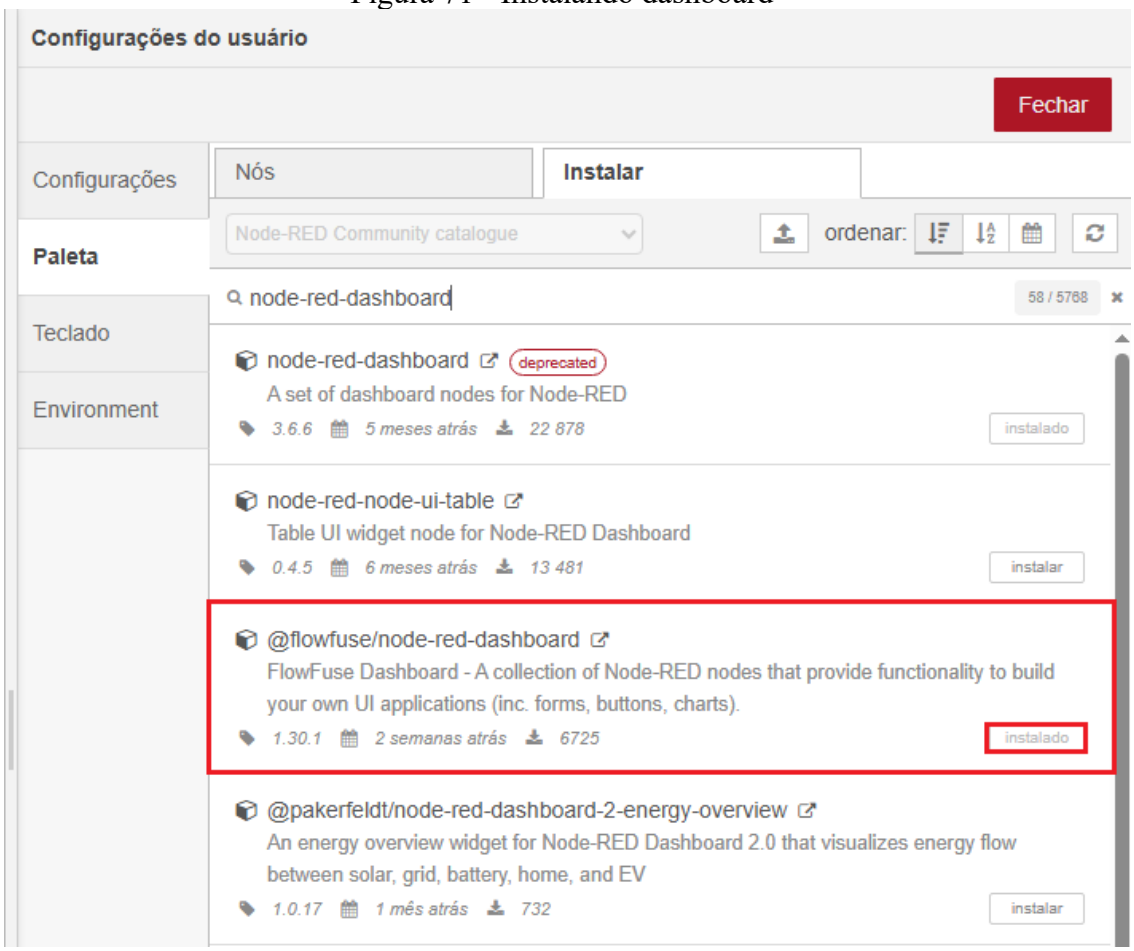
E é desta maneira que os nodes necessários poderão ser acrescentados a paletas de nodes já inclusas.

5.6.3.1 Módulo Node dashboard 2.0

O primeiro modulo a ser instalado será o dashboard 2.0. Será um conjunto de nodes para a criação da interface e apresentação dos dados recebidos via Mosquitto (provenientes do ESP32).

Para instalá-lo, basta pesquisar por “node-red-dashboard” no motor de buscas e clicar em install. (Figura 71).

Figura 71 - Instalando dashboard



Fonte: Autoria própria

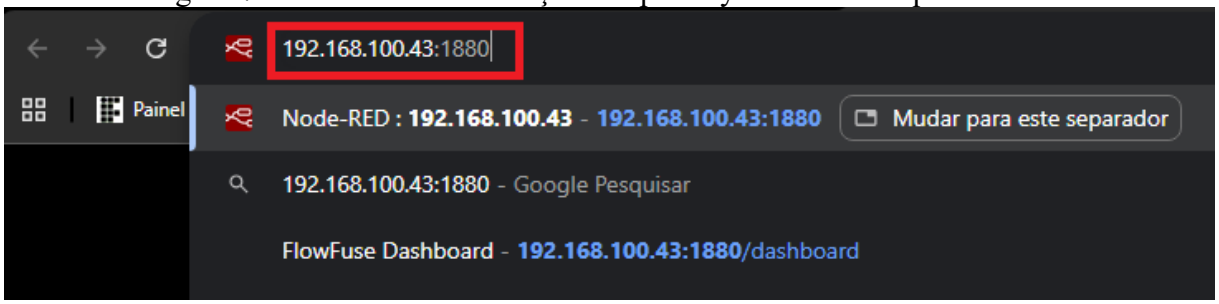
5.6.3.2 Módulo Node para armazenamento

Outro node que foi utilizado, será o Node “ node-red contrib-fs” , que deve ser instalado seguindo as mesmas instruções dos módulos anteriores. Este node será responsável para armazenar os dados finais, além de entregar a possiblidade de realizar o download dos mesmo em outros dispositivos. Após instalar, o mesmo estará presente na aba “storage”, sendo denominado como “ Node fs-file-lister”.

5.7 Desenvolvendo a interface gráfica no Node-RED

Antes de iniciar, é recomendado que os próximos passos realizados no Node-RED seja feita em outra máquina. Para tal, é necessário manter o Raspberry ligado e conectado a internet. Em seguida, no navegador de outro computador ligado na mesma rede do broker, digite “IP DO RAPSBERRY”:1880. Como visto na Figura 72. Pois Desta maneira, o desenvolvimento renderá mais.

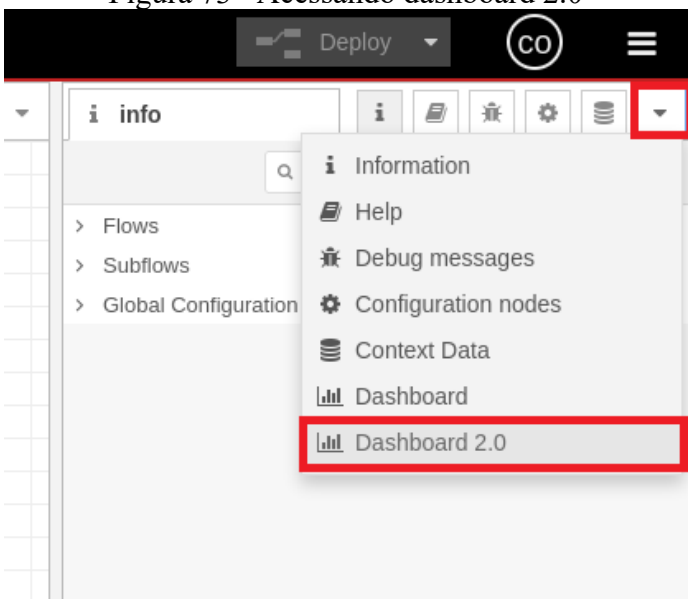
Figura 72 - Inserindo o endereço do rapsberry em outra máquina



Fonte: Autoria própria

Com isso, é necessário abrir a dashboard 2.0 para criação de um grupo de nodes, visto que cada node necessitará de uma orientação de grupo para existir uma correlação com os outros. Para isto, basta clicar na seta apontada para baixo (localizada no canto superior direito), e em seguida clicar em dashboard 2.0. Como visto na Figura 73:

Figura 73 - Acessando dashboard 2.0

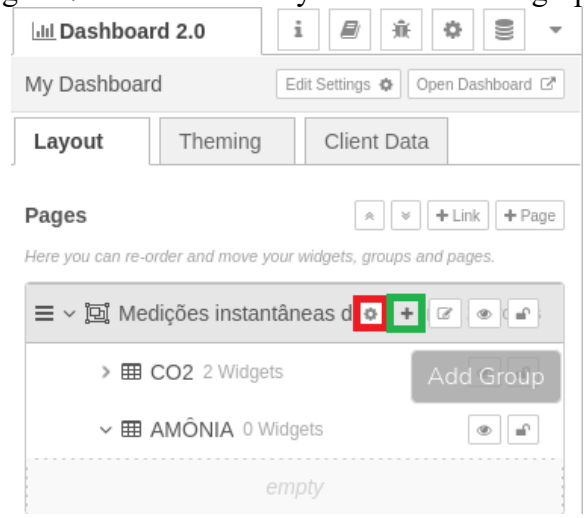


Fonte: Autoria própria

Em seguida, abrirá uma guia, onde será possível determinar a quantidade de layouts, e a quantidade de grupos desse layout. Todo layout deve ter grupos de nodes. É nesta forma que o Node-RED associa os nós nos layouts.

Para acrescentar grupos ao layout, basta clicar no ícone de “+”, (em verde na Figura 74) . Já para editar o nome e outras características do layout, basta clicar no símbolo de engrenagem (em vermelho na Figura 74).

Figura 74 - Editando o layout e adicionado grupos

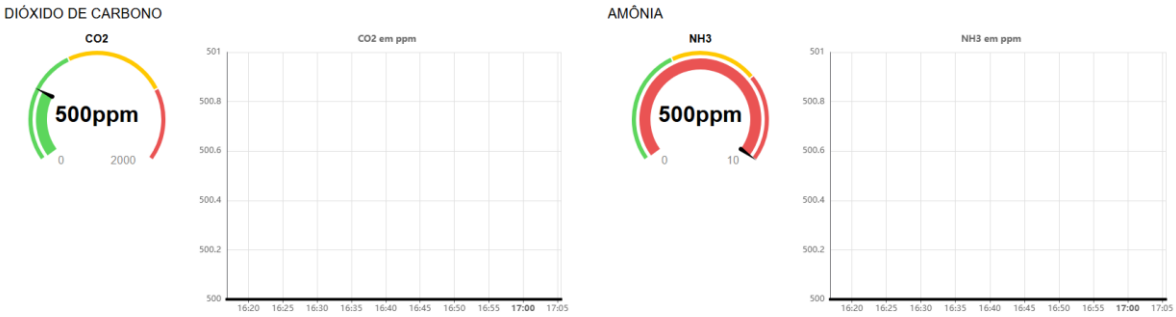


Fonte: Autoria própria

A plataforma oferece uma ampla variedade de possibilidades para personalização do layout, especialmente na versão 2.0 das dashboards, que disponibiliza recursos avançados de configuração e edição. Considerando a abrangência dessas funcionalidades e por não constituírem o foco deste trabalho, essa etapa não será abordada.

Após colocar dois grupos de nodes em um layout, ambos aparecerão no mesmo dashboard (Figura 75).

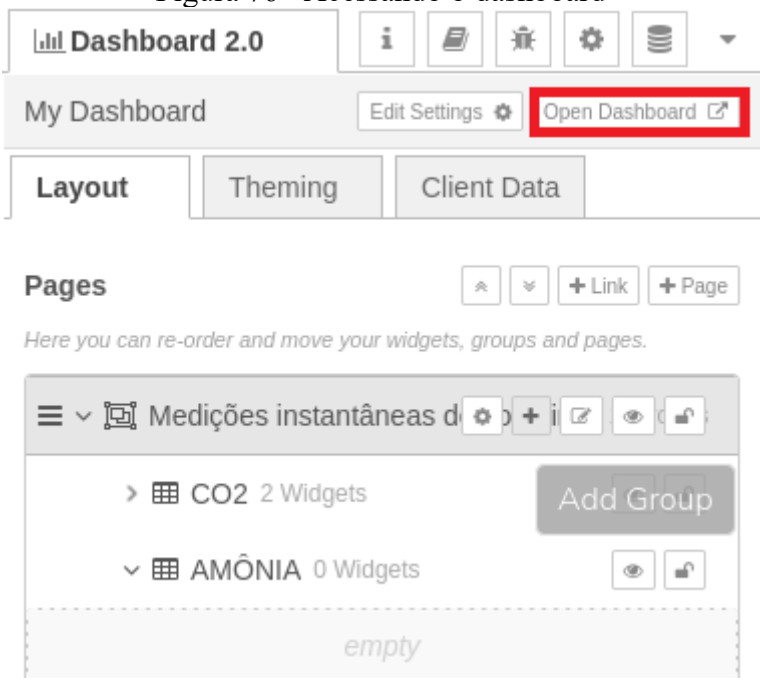
Figura 75 - Dois grupos no mesmo dashboard



Fonte: Autoria própria

Para acessar o dashboard, a tecla “acessar dashboard” pode ser acessada no canto superior direito, como visto na Figura 76.

Figura 76 - Acessando o dashboard

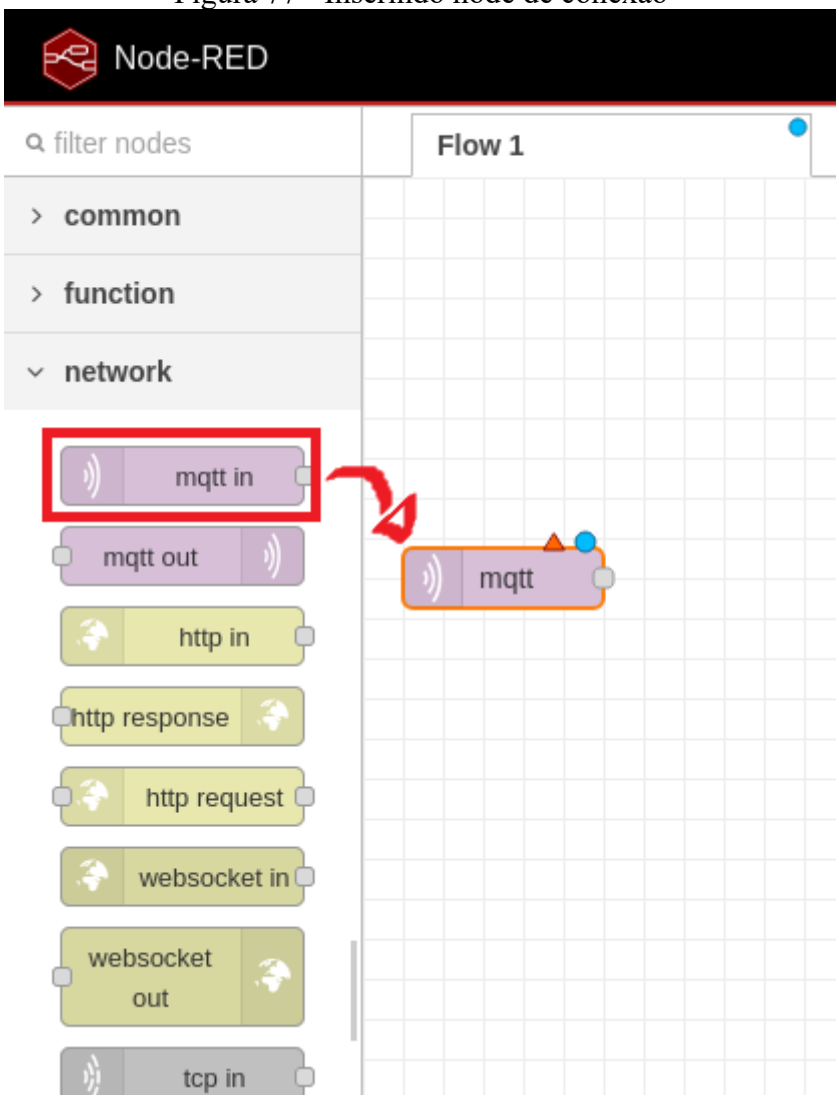


Fonte: Autoria própria

5.7.1 Node de conexão MQTT

Após realizar a criação do layout e do primeiro grupo de nodes, é o momento de criar o nó de conexão com o Mosquitto. Para isto, basta entrar no grupo de funções “network” disponível na lateral esquerda e arrastar a função “mqttt in” para a área de trabalho do Node-RED. Como visto na Figura 77.

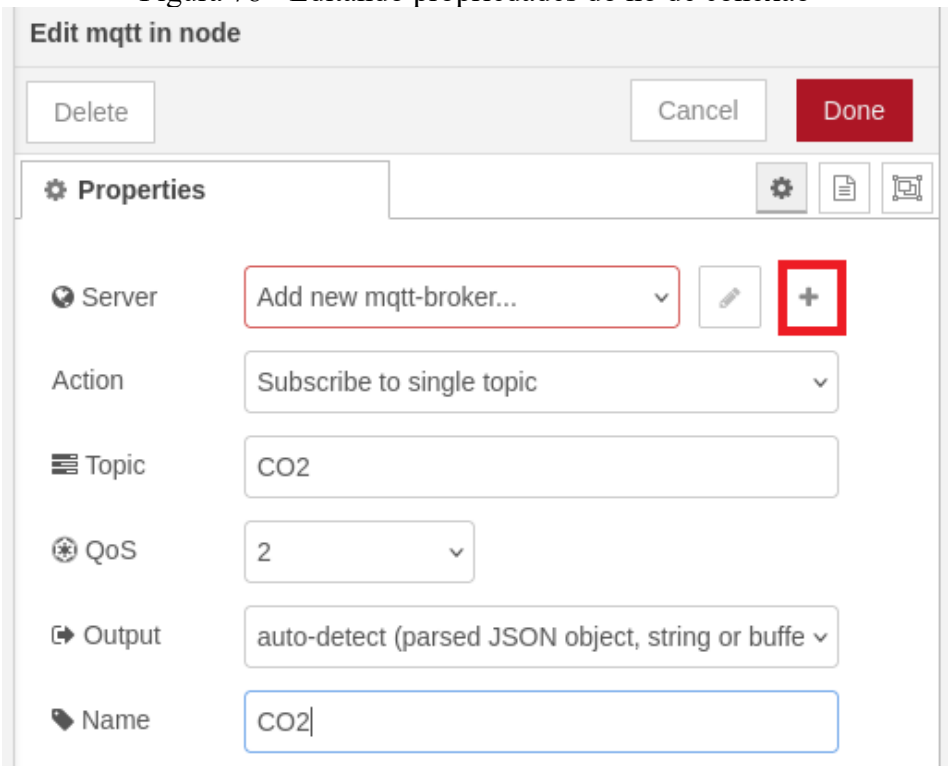
Figura 77 - Inserindo node de conexão



Fonte: Autoria própria

Após colocar este node na área de trabalho, a aba de configuração do mesmo se abrirá clicando duas vezes em seu ícone na área de trabalho. Como visto a seguir:

Figura 78 - Editando propriedades do nó de conexão



Fonte: Autoria própria

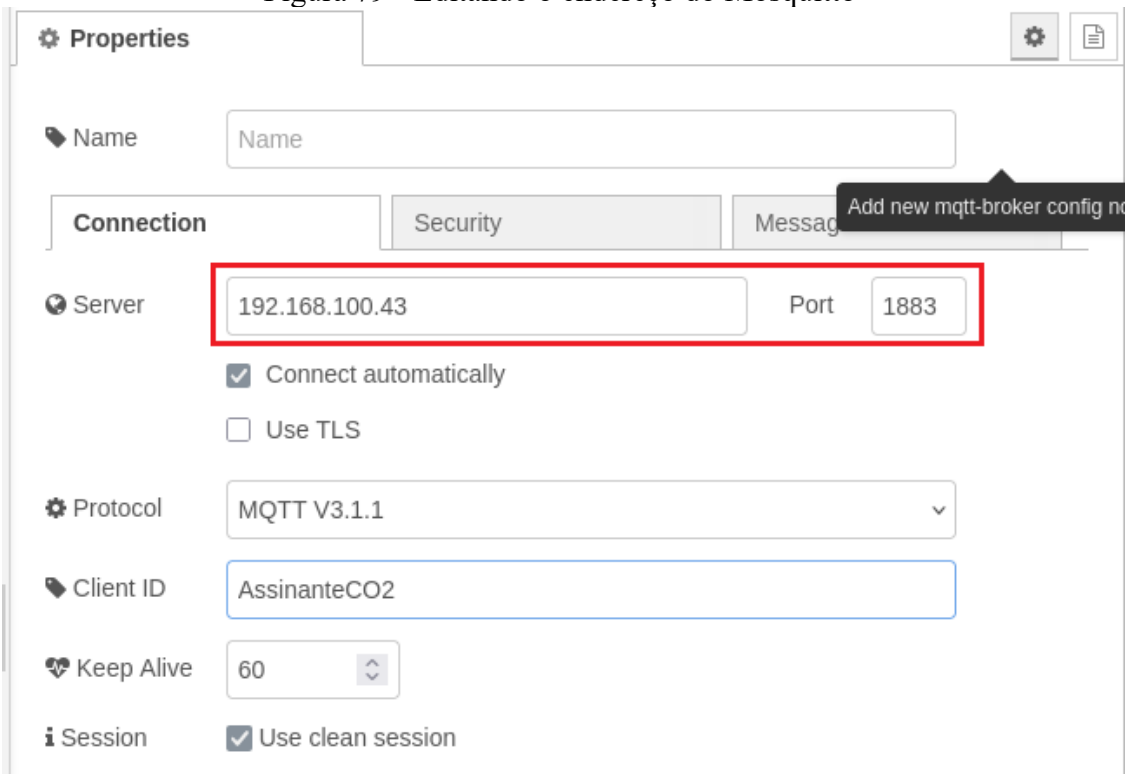
Em “Topic”, a variável exata que será entregue pelo ESP32 deve ser colocada, no exemplo acima era CO_2.

Em “QoS”, diz respeito a qualidade de transmissão de mensagens. Como já explicado anteriormente, o nível dois entrega pelo menos uma mensagem para cada valor sem duplicação. Este foi o escolhido.

Em “name”, com símbolo de tag, diz respeito ao nome do node que será apresentado na área de trabalho.

Para configurar a conexão, é necessário criar um tópico em server (símbolo de +, mostrando em vermelho na Figura 79). Após clicar neste símbolo, a sub-aba “connection” se abrirá, como visto a seguir:

Figura 79 - Editando o endereço do Mosquitto



Fonte: Autoria própria

Em “server”, o ip fixado no raspberry deve ser inserido. Neste caso era 192.168.100.43; Já em “Port”, a porta TCP/UDP também deve ser a mesma do raspberry, ou seja: 1883.

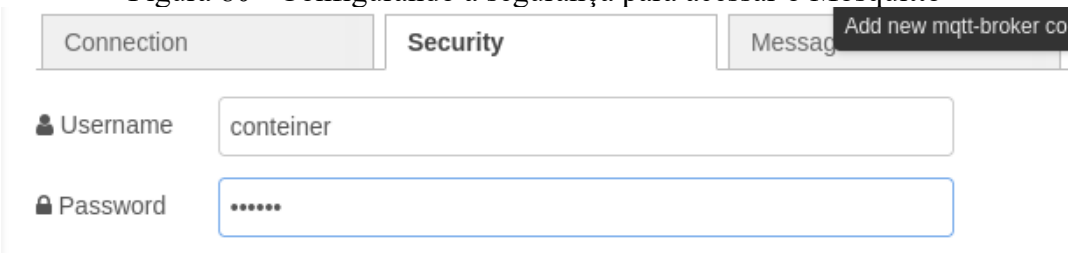
Em “protocol”, deve ser mantido o tipo MQTT v3.1.1, pois trata-se do mais moderno.

“Client ID” diz respeito em como este node será identificado na central de conexões do broker Mosquitto. Aqui pode ser inserido qualquer nome, contando que não tenha outro assinante com mesmo nome.

O “keep alive”, diz respeito ao tempo em que o assinante do Node-RED e o servidor broker MQTT se manterá ativa. Neste caso, o número 60 deve ser mantido como padrão.

Ao lado da sub-aba “connection” se encontra a “security’, que deve ser acessada para definição dos usuários e senha definidos no Mosquitto.

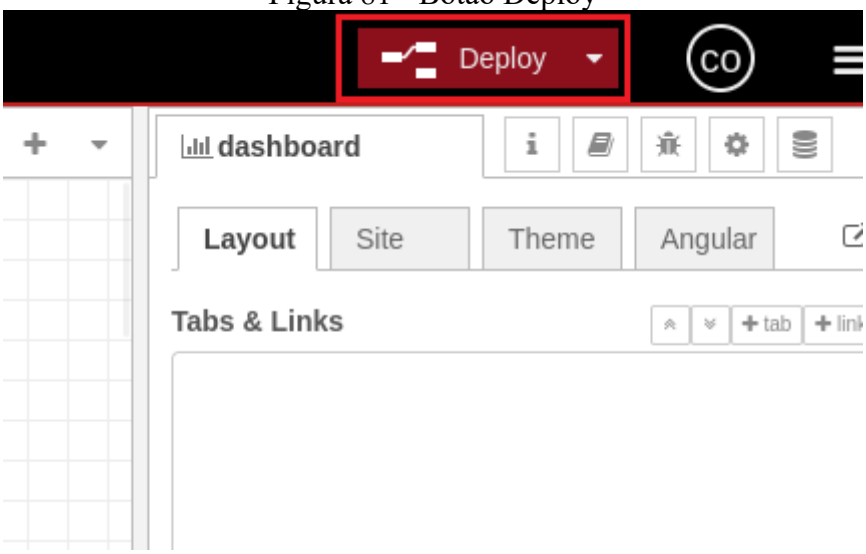
Figura 80 - Configurando a segurança para acessar o Mosquitto



Fonte: Autoria própria

No campo “username” e “password” devem ser inseridos os mesmos usuários e senhas criados para acesso do Mosquitto, caso contrário o mesmo não se conectará. Ao termino das edições, o botão “Done”, presente acima das abas de configurações (Figura 78) poderá ser clicada, e após isso, o botão “deploy”, presente no canto superior direito (Figura 81) deve ser clicado para salvar os nodes na área de trabalho.

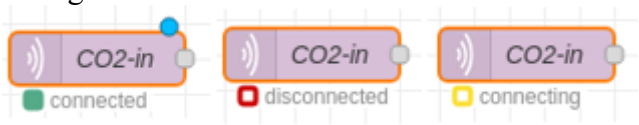
Figura 81 - Botão Deploy



Fonte: Autoria própria

Caso todos as informações inseridas na configuração estejam corretas, a conexão com o broker Mosquitto já estará sendo feita, sendo que seu status de conexão é visível abaixo do node, como visto na Figura 82. Uma vez que a variável da mensagem esteja igual do ESP32, o node de conexão já está pronto para repassar esses valores para outras funções.

Figura 82 - Status de conexão com o broker

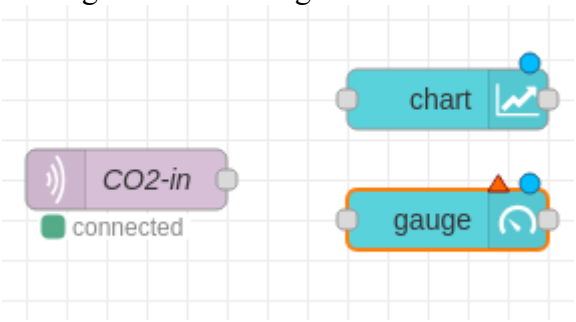


Fonte: Autoria própria

5.7.2 Node de gráfico e mostrador tipo gauge

Ao acessar a barra de nodes na lateral esqueda, em dashboard 2, haverá duas denominadas como “gauge” e “chart”. Ambas poderão ser arrastadas para a área de trabalho para serem configuradas.

Figura 83 - Nodes gráfico e medidor



Fonte: Autoria própria

Estas serão utilizadas tanto para representar os valores instantâneos (mostrador gauge), quanto para medidas ao longo do tempo (armazenamento e valores com faixa de 2 horas).

5.7.2.1 Configurando o Node “chart” e “gauge’

Para que o nó de gráfico possa ser representado no dashboard, o grupo pertencente a ele deve ser selecionado, neste exemplo o layout será “Medições instantâneas do contêiner” e o grupo que ele pertencerá dentro do dashboard será “DIOXIDO DE CARBONO”. No entanto, ele só aparecerá ao clicar no ícone em vermelho, se o usuário já tiver criado ele anteriormente (Figura 74). Caso contrário, um grupo poderá ser criado no momento clicando no icone de “ + “, destacado em verde na Figura 84.

Além disso, também é interessante nomear o node em “name” para que seja fácil identificação para os passos adiante.

Figura 84 - Selecionando grupo do node de gráfico

Fonte: Autoria própria

Dentro desta aba de opções também há várias maneiras de modificar o gráfico de valores que será mostrado no dashboard, ficando a critério do usuário.

Seguindo a mesma linha de raciocínio da configuração anterior, no gauge também deve ser definido o grupo para que o mesmo seja mostrado na dashboard. Neste caso, foi adicionado no mesmo grupo, para que haja tanto informação gráfica, quanto de medição do mesmo valor recebido (neste caso, concentração de CO_2).

5.7.2.2 Ligação e exemplo no dashboard

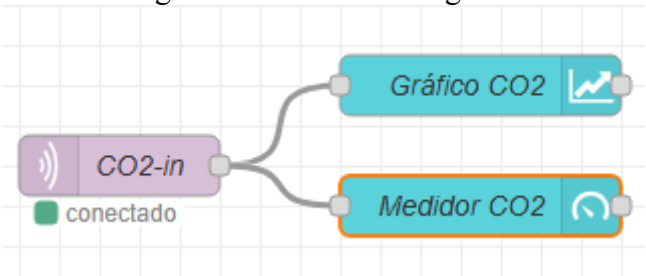
Após definir os grupos corretamente aos nodes, os mesmos já aparecerão identificados no grupo do layout, localizado a direita da tela, como visto na Figura 85:

Figura 85 - Identificação no grupo do layout

Fonte: Autoria própria

Com isto, já é possível realizar a ligação dos mesmos ao node de conexão, bastando clicar em um dos pontos de corda de um nó e liga-lo ao outro, como visto na figura a seguir:

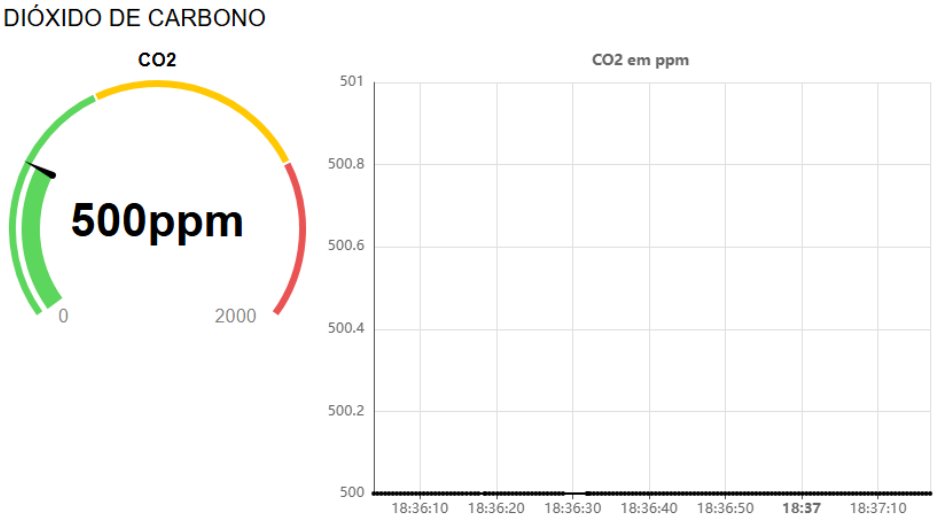
Figura 86 - Nodes interligados



Fonte: Autoria própria

Ao abrir o dashboard, já será possível observar os mesmos presentes e marcando, como visto na figura a seguir:

Figura 87 - Exemplo de gráfico e gauge implementado



Fonte: Autoria própria

5.7.3 Função de Armazenamento e download dos dados

Caso o usuário queira acrescentar esta função no Node-RED, será necessário alterar poucas linhas no código do ESP32 e alguns nodes no Node-RED. Pois para que os dados sejam armazenados e acessados para download posteriormente, é necessário que o Node-RED retorne estes dados organizados em data e hora para o armazenamento local no raspberry.

Ou seja, é necessário que os dados recebidos estejam em formato json, onde todos os dados são recebidos por uma única mensagem.

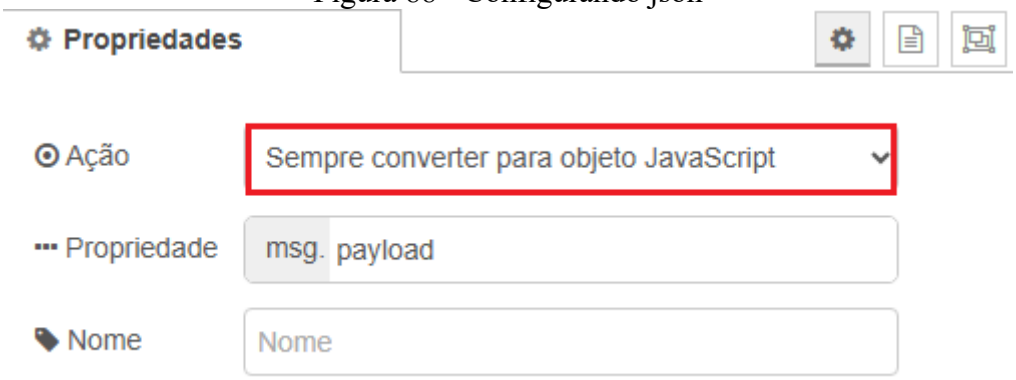
5.7.3.1 Alterando nodes do dashboard

Como as mensagens referentes às concentrações de CO_2 e NH_3 passaram a ser recebidas em uma única estrutura de dados, torna-se necessária a separação dessas informações em mensagens distintas para o processamento adequado.

Para isso, adicione um nó do tipo JSON, disponível na barra lateral esquerda do Node-RED, e dê um duplo clique sobre ele para acessar suas configurações. Em seguida, altere a opção *Action para Always convert to JavaScript Object*, garantindo

que a mensagem seja convertida para um objeto JavaScript. Essa configuração é apresentada na Figura 88.

Figura 88 - Configurando json

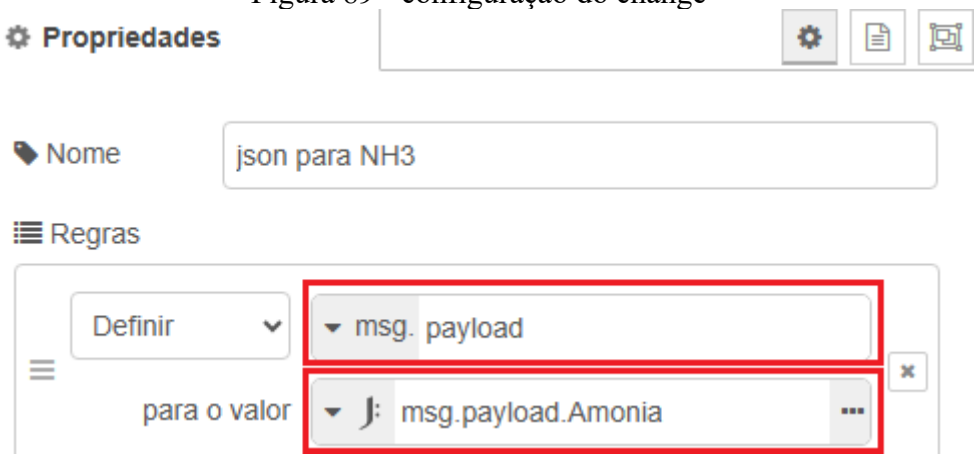


Fonte: Autoria própria

Em seguida, deve ser adicionado um nó do tipo Change após o nó JSON, com a finalidade de extrair apenas o dado de interesse da estrutura da mensagem. Neste exemplo, serão utilizados dois nós Change conectados ao nó JSON: um destinado à concentração de CO_2 e outro à concentração de NH_3.

Ao abrir as configurações do nó Change, deve-se definir o campo msg.payload para receber uma expressão que aponte para o atributo desejado do objeto JavaScript. No caso da amônia, a expressão utilizada é msg.payload.Amonia, de modo que apenas esse valor seja atribuído à variável msg.payload. O mesmo procedimento deve ser realizado para o CO_2, utilizando a expressão correspondente ao respectivo atributo da mensagem.

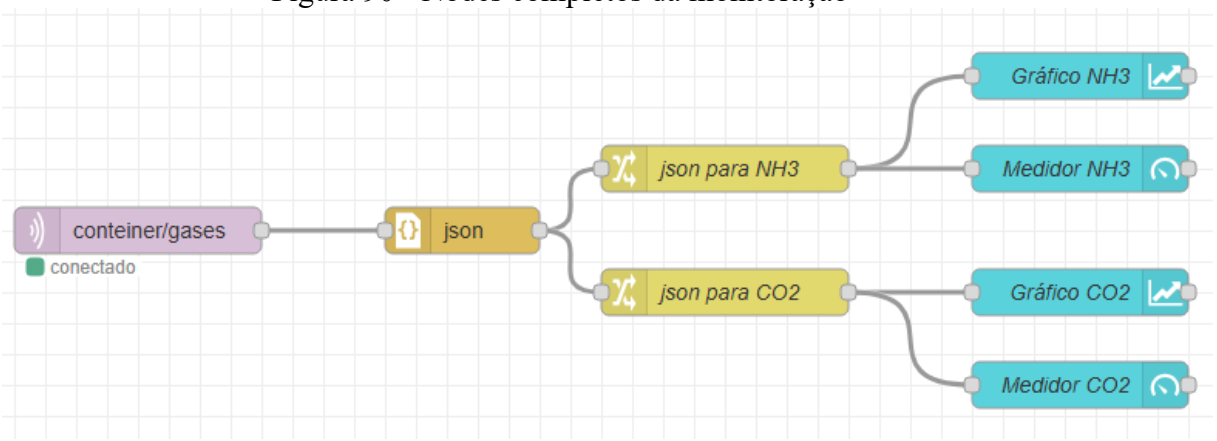
Figura 89 - configuração do change



Fonte: Autoria própria

Em seguida, o node change tanto da NH_3 quanto para o CO_2, poderá ser ligado aos seus respectivos chart's e gauge's, como visto na Figura 90

Figura 90 - Nodes completos da monitoração



Fonte: Autoria própria

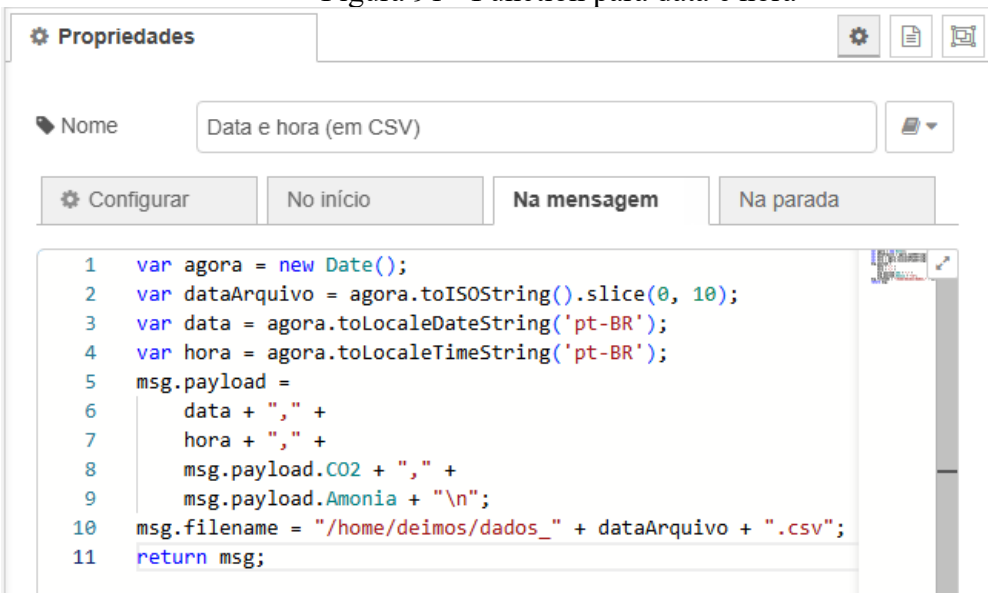
5.7.3.2 Nodes para organizar e armazenar

Primeiramente, deve ser colocado um node tipo "function", que fará a adição de data e hora de cada valor recebido, além de converter o arquivo para csv. No campo em que consta "deimos", deve ser informado o nome atribuído ao Raspberry Pi, uma

vez que esse identificador será utilizado pelo Node-RED para encaminhar os dados correspondentes a cada data.

Além disso, caso o usuário deseje armazenar os arquivos em um diretório diferente, basta alterar o caminho especificado na propriedade msg.filename, indicando o novo local de destino para o salvamento dos arquivos.

Figura 91 - Function para data e hora



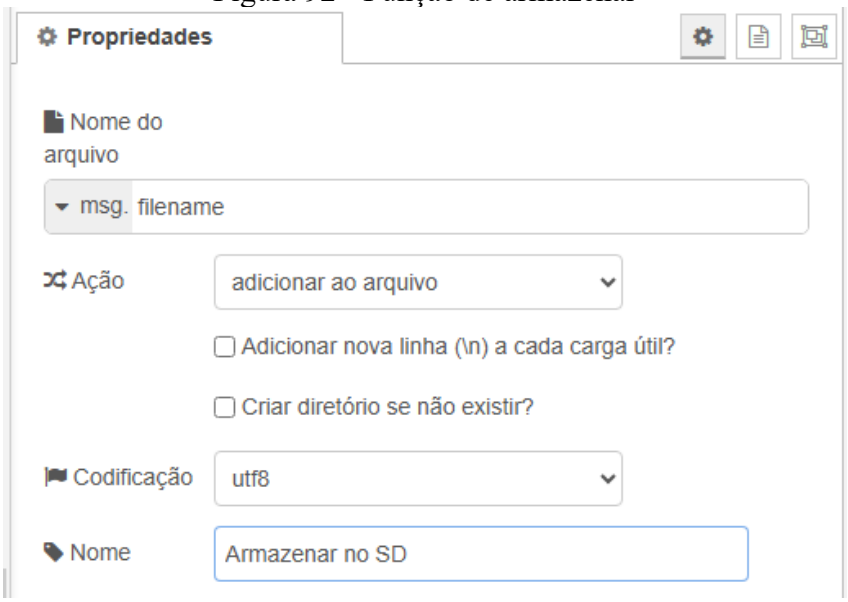
Fonte: Autoria própria

Agora, acrescente um node de “escrever arquivo” ou “file out”. Em seguida, coloque o tipo do arquivo para msg e denomine o nome do arquivo para fillename, pois assim.o armazenamento de dados é feito dinamicamente por meio da propriedade do próprio Node-RED que já separará os registros para a coleta futuramente.

Em seguida, a ação deve ser colocada para “acrescentar ao arquivo”, desta forma, o Node-RED não irá apagar e nem subscrever um dado por cima do outro.

A linha \n pode ser desmarcada, pois este passo já foi feito anteriormente na função de data e hora. Por último, a codificação deve ser em utf8. Todos os passos podem ser vistos na Figura 92.

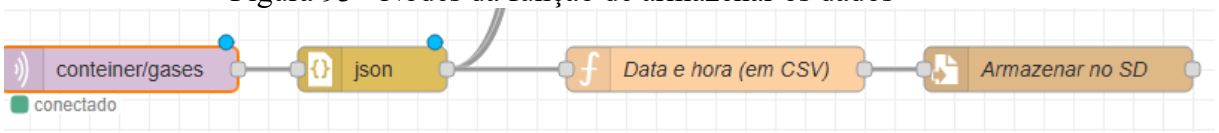
Figura 92 - Função de armazenar



Fonte: Autoria própria

Ao final, os nodes responsáveis pelo armazenamento ficarão da seguinte forma:

Figura 93 - Nodes da função de armazenar os dados

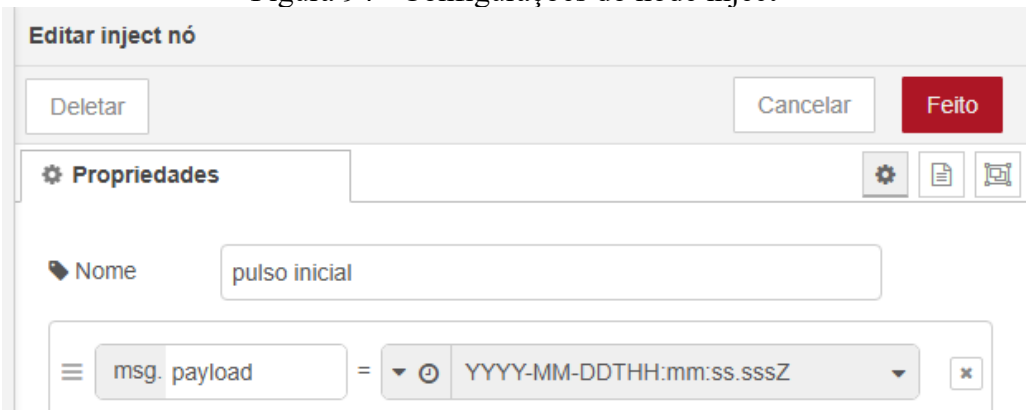


Fonte: Autoria própria

5.7.3.3 Nodes para exibir histórico armazenado no dashboard

Inicialmente, é preciso acrescentar um node tipo inject, onde o mesmo será responsável em dar o pulso inicial para que o node a frente funcione (será explicado adiante). Para tal, suas configurações devem ser de acordo com a Figura 94.

Figura 94 - Configurações do node inject

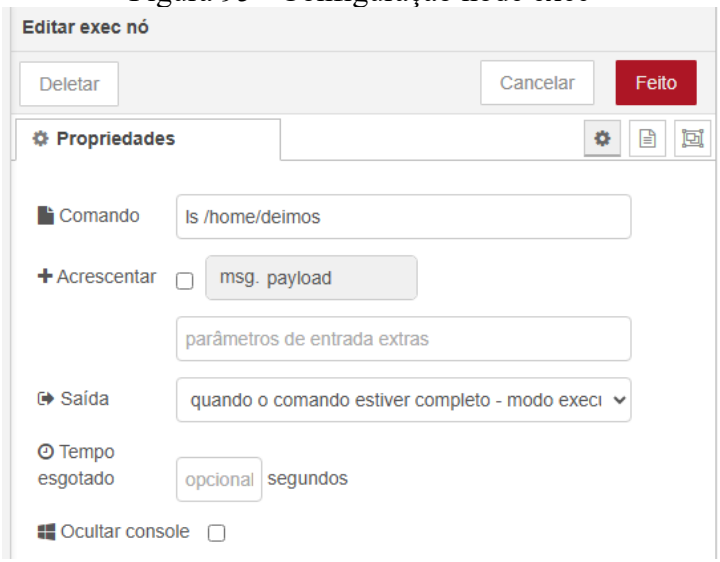


Fonte: Autoria própria

Em seguida, deve-se acrescentar um node tipo exec (disponível na aba função). Este node será responsável em buscar todos os arquivos armazenados anteriormente pelo Node-RED na memória interna do Raspberry. Ele basicamente irá executar um comando no console do raspberry afim de identificar todos os arquivos .csv presentes (no caso, datas de leituras).

Esse nó somente é executado quando está conectado a um nó do tipo Inject, responsável por iniciar sua execução. Em termos de funcionamento, esse processo equivale à inserção manual de um comando no terminal do próprio Raspberry Pi, permitindo que a instrução configurada seja executada pelo sistema operacional. O local de armazenamento dos dados dependerá do diretório especificado previamente pelo usuário durante a configuração do sistema. Dessa forma, os arquivos serão salvos no caminho definido na propriedade correspondente. A configuração utilizada neste exemplo é apresentada na Figura 95.

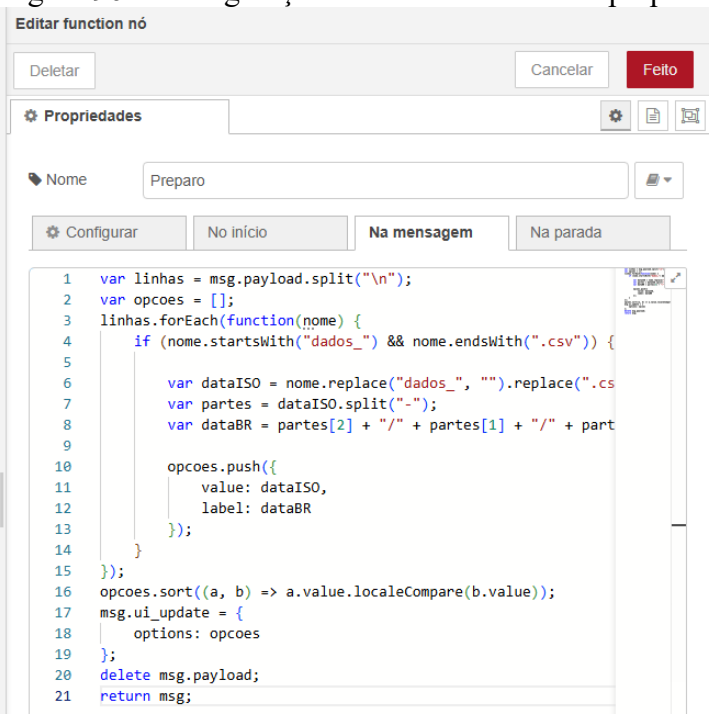
Figura 95 - Configuração node exec



Fonte: Autoria própria

Com isto, será o momento de preparar estes dados de forma que o Node-RED consiga “ler” para enviar para o node de seleção de dados (node adiante). Para isto, foi necessário o acréscimo de um node tipo function. Ele vai basicamente renomear a página, substituir pontos por virgula e retirar o .csv do nome. Como visto a seguir:

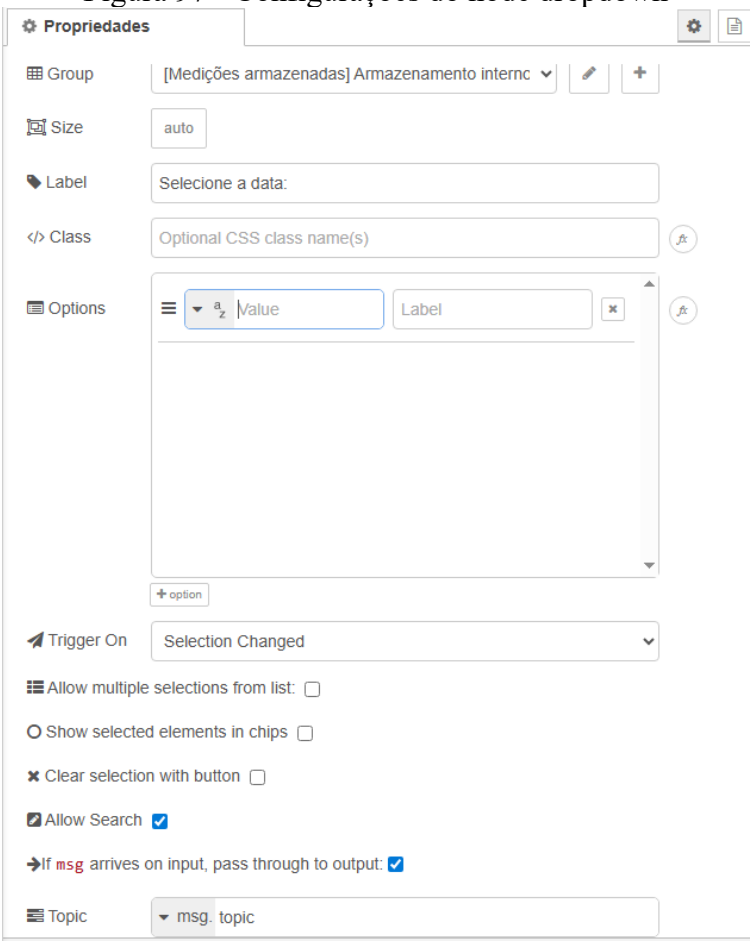
Figura 96 - Configurações do node function de preparo



Fonte: Autoria própria

Após o preparo dos arquivos contendo os dados. Será o momento de inserir um node de seleção, onde o usuário escolherá a data para aparecer em gráfico. Para tal, foi necessário um node tipo dropdown (disponível na aba dashboard 2.0). Vale salientar, que o mesmo já deve estar em outro grupo no dashboard, pois irá aparecer em outra aba no layout. As configurações desse node devem ser seguidas iguais a Figura 97.

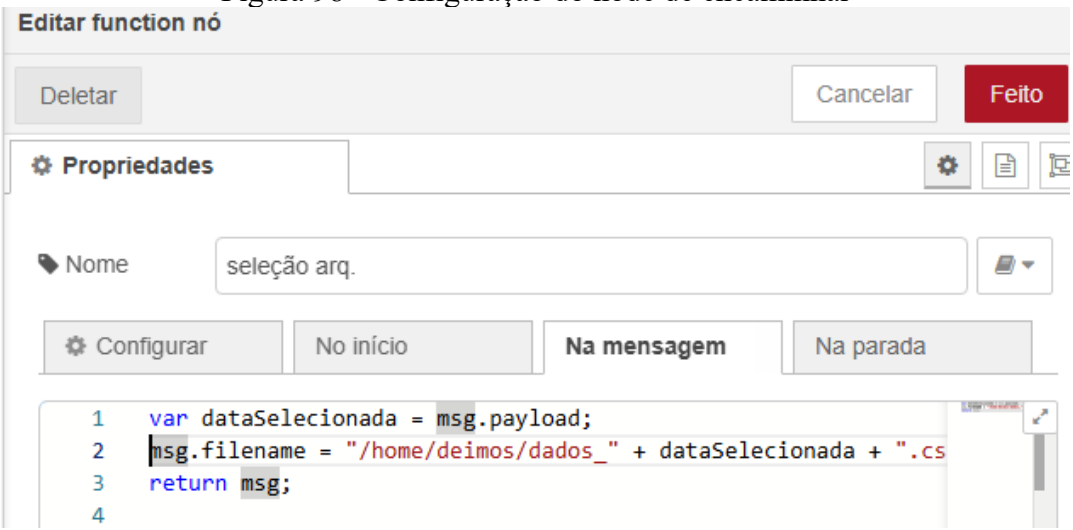
Figura 97 - Configurações do node dropdown



Fonte: Autoria própria

Após isto, outro node tipo function deve ser inserido para montar o caminho do dia selecionado. A configuração deste é evidenciado na Figura 98.

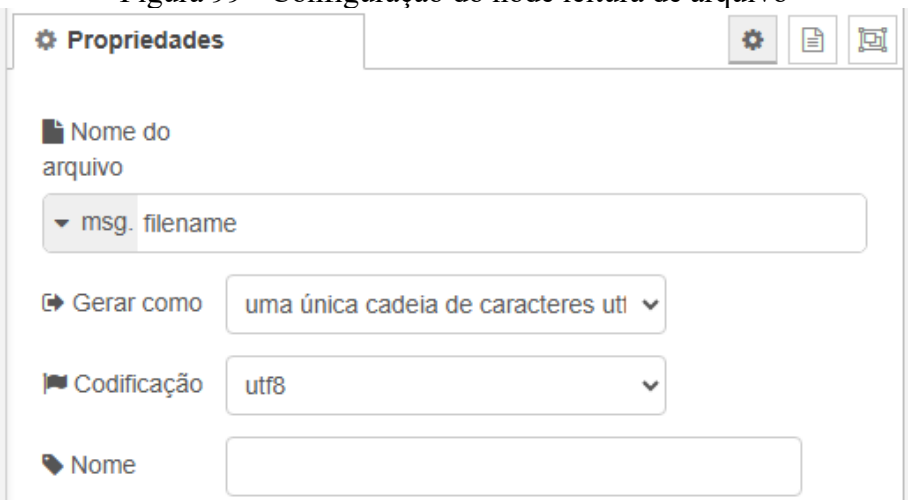
Figura 98 - Configuração do node de encaminhar



Fonte: Autoria própria

O próximo node a ser utilizado, será o responsável em ler o arquivo selecionado e converte-lo para a codificação utf8, para que no próximo node, possa ser inserido as identificações devidas para cada variável contida no arquivo. Suas configurações são apresentadas na Figura 99.

Figura 99 - Configuração do node leitura de arquivo



Fonte: Autoria própria

O penúltimo node do fluxo, será responsável em converter o formato anteriormente dito (csv decodificado) para um formato especifico que o node chart

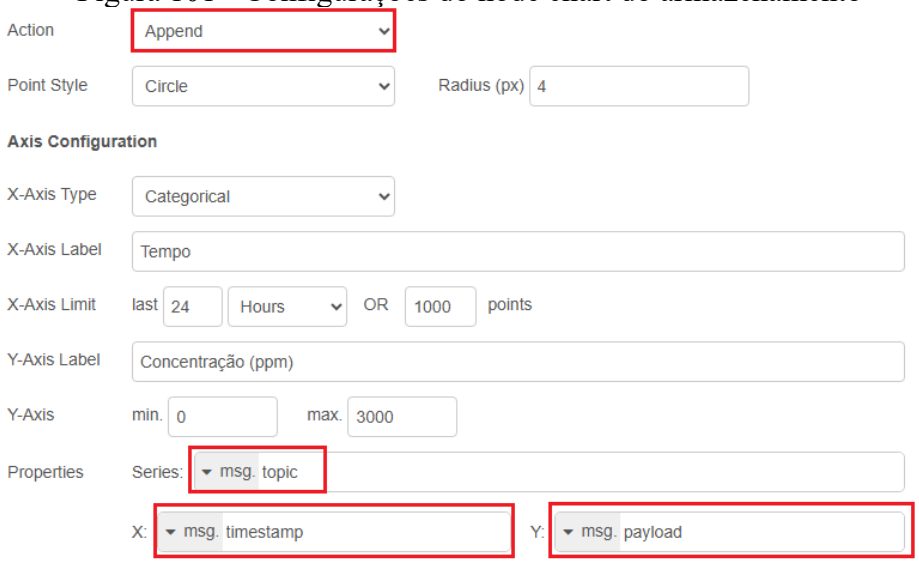
consiga ler. Para isto, adicione um node tipo function, com as seguintes configurações feitas:

Figura 100 - Código node de conversão para chart

Fonte: Autoria própria

O último node será o chart (mostrador de gráficos). Para isto, o mesmo poderá ser adicionado, e suas configurações especificas devem ser seguidas de acordo com a Figura 101. Tendo atenção as opções demarcadas em vermelho.

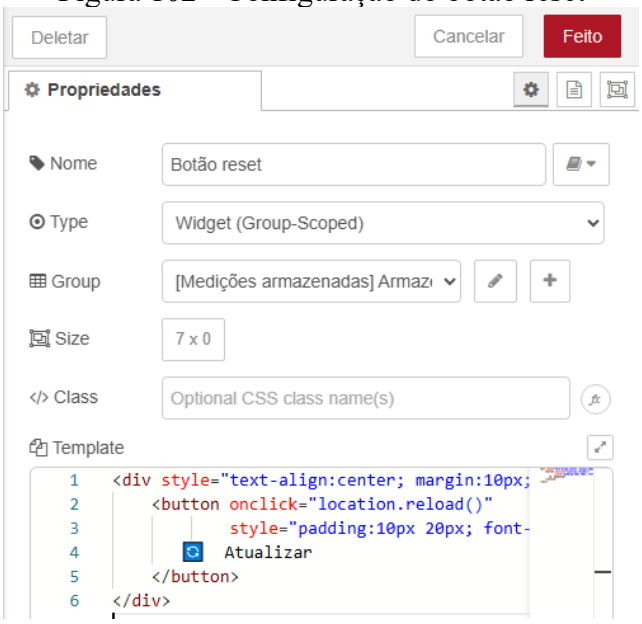
Figura 101 - Configurações do node chart do armazenamento



Fonte: Autoria própria

Para que os dados de uma data não continuem à frente de outro previamente selecionado, é necessário dar um reset na página para que o gráfico também se reinicie. Para tal, foi implementado um botão no layout por meio de um node tipo *template*. Seu código pode ser visto na Figura 102 (o símbolo 🔄 foi colocado no código).

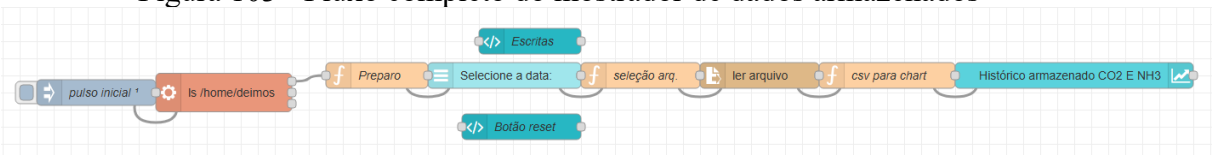
Figura 102 - Configuração do botão reset



Fonte: Autoria própria

O fluxo completo da função de mostrar os dados armazenados, pode ser vista na Figura 103.

Figura 103 - Fluxo completo do mostrador de dados armazenados



Fonte: Autoria própria

6 CAPÍTULO 5 – METODOLOGIA EXPERIMENTAL PARA VALIDAÇÃO FUNCIONAL DA PLATAFORMA IOT

A validação funcional da plataforma IoT é importante para verificar se os dispositivos desenvolvidos são capazes de monitorar, registrar e transmitir dados de concentração de gases em condições controladas, visando sua futura aplicação em sistemas de cultivo em contêineres. Para essa etapa, foi utilizado como aparato experimental uma caixa contendo os sensores de CO_2 e NH_3, permitindo avaliar a resposta da plataforma em um ambiente fechado e com volume conhecido. Nesse contexto, foi realizado um estudo associado à transferência de massa, de modo a simular variações nas concentrações de CO_2 e NH_3 que podem ocorrer em aplicações reais. Em sistemas de cultivo em contêineres, a decomposição do bicarbonato de amônio pode liberar dióxido de carbono, gás de interesse para o processo fotossintético, e amônia, composto indesejável que precisa ser monitorado e removido antes de entrar no contêiner. Assim, a metodologia proposta permitiu avaliar a capacidade da plataforma em acompanhar a dinâmica desses gases ao longo do tempo e fornecendo subsídios para avaliação da sensibilidade dos sensores para o controle e a segurança do processo.

6.1 Aparato Experimental

A caixa utilizada no aparato experimental (Figura 104) possuía dimensões aproximadas de 28 × 30 × 42 cm, com volume útil de 21,5 L, sendo empregada para a realização dos ensaios de validação da plataforma IoT desenvolvida. Em seu interior, foram posicionados os sensores de CO_2 (1) e NH_3 (2), integrados ao sistema de aquisição de dados. Esse aparato foi utilizado para criar um ambiente fechado, com volume conhecido, permitindo avaliar a resposta dos sensores diante das variações de concentração dos gases ao longo do tempo.

Figura 104 - Aparato Experimental

Fonte: Autoria Própria

Para a realização dos ensaios, foram adicionados ao interior da caixa 100 mL de solução aquosa de hidróxido de amônio (NH_4OH) em uma capsula de evaporação (3), em concentração conhecida, possibilitando a liberação de NH_3 para a fase gasosa. A dinâmica do sistema foi avaliada principalmente a partir das curvas de variação da concentração de amônia em função do tempo, permitindo analisar a resposta do sensor e a sensibilidade da plataforma frente às alterações de concentração no ambiente experimental. Os níveis de CO_2 também foram monitorados durante os ensaios, entretanto, esses dados foram utilizados apenas como acompanhamento complementar, não sendo realizada uma análise dinâmica específica para associar à diminuição da concentração no meio gasoso ou à possível solubilização e reação com a solução de hidróxido de amônio presente na caixa. Dessa forma, a análise de transferência de massa foi direcionada à interpretação do

comportamento temporal da amônia no interior do aparato experimental, servindo como base para a validação funcional da plataforma IoT.

Com o objetivo de reduzir perdas por vazamento dos gases, as bordas da caixa foram preenchidas com cola de poliuretano (PU) (4), utilizando a própria tampa como molde, e mantidas em repouso por 24 horas para secagem. Após esse período, formou-se uma camada flexível nas extremidades, semelhante a uma junta de vedação. Para aumentar a estanqueidade do sistema, foi utilizado papel-filme (5) na região superior da caixa, devido à sua aderência à cola PU seca. Em seguida, a tampa era posicionada sobre o conjunto, pressionando o papel-filme contra a vedação formada. A cada novo experimento, o papel-filme era substituído, reduzindo a possibilidade de interferência entre os ensaios.

Os testes foram realizados na capela do laboratório T301 do Câmpus Itumbiara (Figura 105), com temperatura ambiente controlada em aproximadamente 20 °C por meio de sistema de ar-condicionado.

Figura 105 - Capela utilizada

Fonte: Autoria própria

6.2 Fundamentos dos ensaios: transferência de massa e equilíbrio liquido-gasoso

Neste trabalho, a análise não teve como objetivo determinar propriedades termodinâmicas rigorosas do sistema NH_3- H_2O, mas utilizar a dinâmica de liberação da amônia como referência experimental para avaliar a resposta do sensor e da plataforma IoT. Em sistemas fechados, a concentração do gás no *headspace* varia com o tempo e pode alterar o próprio gradiente responsável pelo fluxo, tornando a análise temporal essencial (KUTZBACH et al., 2007).

Em solução aquosa, a amônia encontra-se em equilíbrio com o íon amônio, conforme a reação (Eq. 1).

$$NH_3(aq) + H_2O(l) \rightleftharpoons NH_4^+(aq) + OH^-(aq) \quad (1)$$

Esse equilíbrio indica que nem toda a amônia presente na solução está disponível diretamente para volatilização. A fração molecular NH_3 (aq) é a parcela mais diretamente associada à transferência para a fase gasosa, que pode ser representada de forma simplificada pela Eq (2).

$$NH_3(aq) \rightleftharpoons NH_3(g) \quad (2)$$

A volatilização da amônia é descrita na literatura como um processo dependente da concentração de NH_3 em equilíbrio com a solução, da concentração de $NH3$ no ar, do equilíbrio NH_3/NH_4+, da constante de Henry, do pH, da temperatura, da força iônica e das condições de transporte na fase gasosa. Esse processo pode ser representado por modelos de transferência de massa nos quais o fluxo de NH_3 depende do gradiente entre a concentração gasosa em equilíbrio com a solução e a concentração no ar ambiente (Montes; Rotz; Chaoui, 2009).

No entanto, o modelo de volatilização de amônia apresentado por Montes, Rotz e Chaoui (2009) não deve ser transferido diretamente para o presente experimento,

pois os autores tratam principalmente de superfícies emissoras expostas ao escoamento de ar, como esterco e soluções amoniacais em câmara de fluxo.

No aparato deste trabalho, a caixa permaneceu fechada durante o intervalo de medição. Assim, a variação da concentração de amônia na fase gasosa pode ser descrita por meio de um balanço macroscópico de massa. Admitindo que a amônia liberada pela solução se acumula no volume gasoso útil da caixa, o balanço de massa pode ser representado de acordo com a Eq. (3).

$$V_g \frac{dC}{dt} = AJ_{ap} \qquad (3)$$

Neste caso, Vg é o volume gasoso útil da caixa, C_g é a concentração de NH_3 medida na fase gasosa, A é a área superficial da solução exposta ao gás e J_{ap} é o fluxo aparente de transferência de amônia para o volume gasoso.

A Eq. (3) representa o princípio de acúmulo de massa no interior da caixa. Assim, a amônia transferida para a fase gasosa provoca o aumento da concentração medida ao longo do tempo. A partir desse balanço, o fluxo aparente pode ser estimado de acordo com a Eq. (4).

$$J_{ap} = \frac{V_g}{A} \frac{dC}{dt} \qquad (4)$$

O termo J_{ap} deve ser interpretado como um fluxo aparente baseado na taxa de acúmulo de amônia detectada no volume gasoso da caixa, e não necessariamente como o fluxo absoluto de volatilização na interface líquido-gás. Essa distinção é necessária porque o valor calculado depende diretamente das condições experimentais adotadas, como o volume interno disponível, a área superficial da solução exposta, o intervalo de medição e a concentração registrada pelo sensor.

Como a concentração de NH_3 no interior da caixa aumenta durante o ensaio, a força motriz para a liberação da amônia tende a diminuir progressivamente. Por isso, não se espera que a concentração cresça de forma linear durante todo o experimento. Em estudos com câmaras fechadas, Kutzbach et al. (2007) demonstram que a evolução da concentração de um soluto pode ser não linear e que modelos

exponenciais podem representar melhor essa variação do que regressões lineares simples.

Dessa forma, adotou-se um modelo de primeira ordem para descrever a dinâmica do sistema, na qual a taxa de variação da concentração é proporcional à diferença entre a concentração final (C∞) e a concentração em um determinado instante (C), conforme apresentado na Eq. (5).

$$\frac{d\mathrm{C}_0}{dt} = \mathrm{k}(C_\infty - \mathrm{C}_0) \qquad (5)$$

Em que C_0 é a concentração de amônia ao longo do tempo na fase gasosa, C_∞ é a concentração assintótica aparente observada experimentalmente, associada ao estado final do sistema nas condições do ensaio, e k é a constante cinética aparente relacionada à transferência e ao acúmulo de amônia no volume gasoso da caixa.

A solução da equação diferencial apresentada na Eq. (3) é descrita pela Eq. (6).

$$\mathrm{C}(t) = C_\infty - (C_\infty - \mathrm{C}_0)e^{-\mathrm{k}t} \qquad (6)$$

Neste caso, C_0 é a concentração inicial de amônia na fase gasosa que foi considerada zero.

Durante os instantes iniciais do ensaio, pode ocorrer a formação de uma pluma de amônia próxima à região de liberação, a qual pode atingir o sensor antes que o gás esteja mais bem distribuído no volume interno da caixa. Nessa condição, a leitura inicial do sensor pode representar uma concentração local, influenciada pela posição relativa entre a solução e o sensor, e não necessariamente a concentração média do volume gasoso.

Por esse motivo, ao ajustar a Eq. (6) aos dados experimentais de variação temporal da concentração de amônia, fez-se necessária a exclusão dos pontos correspondentes à formação da pluma, visto que tais valores comprometiam a média do sistema. Desse modo, o ajuste foi realizado desconsiderando-se esses pontos

discrepantes, visando a uma representação mais fiel do comportamento global do processo.

O parâmetro k, determinado por regressão não linear, foi utilizado para avaliar a velocidade aparente com que a concentração de NH_3 se aproxima do patamar de estabilização no interior da caixa experimental. Valores mais elevados de k indicam uma resposta mais rápida do sistema, enquanto valores menores indicam uma aproximação mais lenta ao valor final aparente.

Entretanto, k não deve ser interpretado como um coeficiente físico isolado de difusão ou convecção para transferência de massa. Em estudos com câmaras fechadas, Kutzbach et al. (2007) destacam que os parâmetros ajustados em modelos exponenciais não podem ser interpretados diretamente como parâmetros físicos individuais.

Neste caso o parâmetro k ajuda a entender matematicamente a influência sobre a dinâmica global do sistema, como avaliado na modelagem de sistemas de controle de processos. Portanto, esse termo incorpora, de maneira conjunta, efeitos relacionados à difusão molecular da amônia no ar, possíveis movimentos convectivos naturais do sistema em função do gradiente de concentração, à distância entre a solução e o sensor, à área de contato líquido-gás e à umidade.

Essa abordagem é necessária porque o aparato experimental não possuía ventilação forçada, não foram medidos perfis espaciais de concentração e não foram determinadas separadamente as resistências de transferência de massa nas fases líquida e gasosa.

6.2.1 Determinação do fluxo de transferência de massa (NH_3)

A determinação do fluxo inicial de transferência de NH_3 ($J_{ap,0}$) permite avaliar a resposta inicial do aparato experimental à liberação de amônia para a fase gasosa, pois nesse instante a concentração acumulada na caixa ainda é mínima e a variação medida pelo sensor é mais representativa na condição inicial do processo. Assim, $J_{ap,0}$ foi estimado a partir do balanço de acúmulo de massa na fase gasosa,

utilizando a taxa inicial de aumento da concentração obtida pela derivada da curva ajustada em t=0, conforme a Eq. (7).

$$J_{ap,0} = \frac{V_g}{A}\left(\frac{dC}{dt}\right)_{t=0} \qquad (7)$$

O $J_{ap,0}$ foi utilizado para comparar a resposta do sistema sob diferentes concentrações da solução de NH_4OH inserida no interior da caixa, permitindo avaliar como a maior disponibilidade de amônia na fase líquida influencia a taxa inicial de acúmulo de NH_3 na fase gasosa. Além disso, a análise do fluxo inicial aparente possibilitou verificar a sensibilidade dinâmica dos sensores utilizados na plataforma IoT desenvolvida, especialmente quanto à sua capacidade de detectar variações iniciais na liberação de amônia e diferenciar as condições experimentais avaliadas.

Como os dados experimentais de concentração foram ajustados por um modelo de primeira ordem, conforme apresentado na Eq. (6), a taxa inicial de variação da concentração, $(dC/dt)_{t=0}$, foi obtida a partir da derivada desse modelo em relação ao tempo. Em seguida, substituindo-se t=0, determina-se a taxa inicial de aumento da concentração de NH_3 na fase gasosa, conforme apresentado na Eq. (8).

$$\left(\frac{dC_g}{dt}\right)_{t=0} = k(C_\infty - C_0) \qquad (8)$$

Considerando que, no início do experimento, a concentração inicial de NH_3 na fase gasosa é nula ($C_0 = 0$), a Eq. (8) pode ser substituída na Eq. (7), permitindo calcular o fluxo inicial aparente de transferência de NH_3 conforme apresentado na Eq. (9).

$$J_{ap,0} = \frac{V_g}{A} k C_\infty \qquad (9)$$

A Eq. (9) foi utilizada como parâmetro comparativo entre os ensaios, pois expressa o fluxo inicial aparente de NH_3 no instante em que a força motriz para transferência da amônia é máxima. Como o aparato corresponde a um sistema fechado, esse valor deve ser interpretado como fluxo inicial de acúmulo na fase gasosa da caixa.

Assim, a análise foi baseada em dois parâmetros principais: k, relacionado à velocidade aparente de aproximação da concentração de NH_3 ao patamar de estabilização, e $J_{ap,0}$, associado ao acúmulo inicial de amônia na fase gasosa. Essa abordagem permitiu comparar as diferentes condições experimentais no mesmo aparato e avaliar a resposta dinâmica da plataforma IoT desenvolvida, sem extrapolar os resultados para propriedades termodinâmicas rigorosas ou para coeficientes universais de transferência de massa do sistema (NH_3- H_2O)

6.3 Procedimento experimental e aquisição dos dados

Os experimentos foram conduzidos utilizando a caixa experimental descrita na seção 6,1, mantendo-se o arranjo físico de medição em todos os ensaios. As soluções foram preparadas pela diluição de volumes definidos de hidróxido de amônio 28-30% m/m P.A em água, até o volume final de 100 mL, com o objetivo de obter diferentes condições iniciais de liberação de amônia para a fase gasosa.

Considerando o hidróxido de amônio com teor aproximado de 29% m/m de NH_3, densidade estimada de 0,90 g/mL e massa molar de NH_4OH de 35,05 g/mol, foram calculadas as concentrações das soluções preparadas. Esses valores foram utilizados apenas para caracterizar as condições iniciais dos ensaios, uma vez que a concentração medida na fase gasosa depende do processo de liberação da amônia da solução, do acúmulo no volume gasoso, da geometria do sistema, da temperatura e da resposta do sensor. As condições experimentais avaliadas são apresentadas na Tabela 1.

Tabela 1 — Condições experimentais utilizadas nos ensaios de liberação de amônia.

Experimento	Volume de NH_4OH comercial (ml)	Volume final da solução (ml)	Concentração da solução de NH_4OH (mol/L)	Temperatura (°C)
E1	0,5	100	0,077	20 °C
E2	1,0	100	0,153	20 °C
E3	2,0	100	0,306	30 °C

A aquisição dos dados foi mantida até que a concentração de $NH_3(g)$ medida apresentasse tendência de estabilização, indicando a aproximação a um patamar aparente no interior da caixa. A partir das curvas experimentais de concentração de NH_3 em função do tempo, foram realizados os ajustes não lineares do modelo (Eq. 6) descrito anteriormente (seção 6.2), permitindo estimar os parâmetros associados à dinâmica de acúmulo da amônia na fase gasosa (k e $J_{app,0}$).

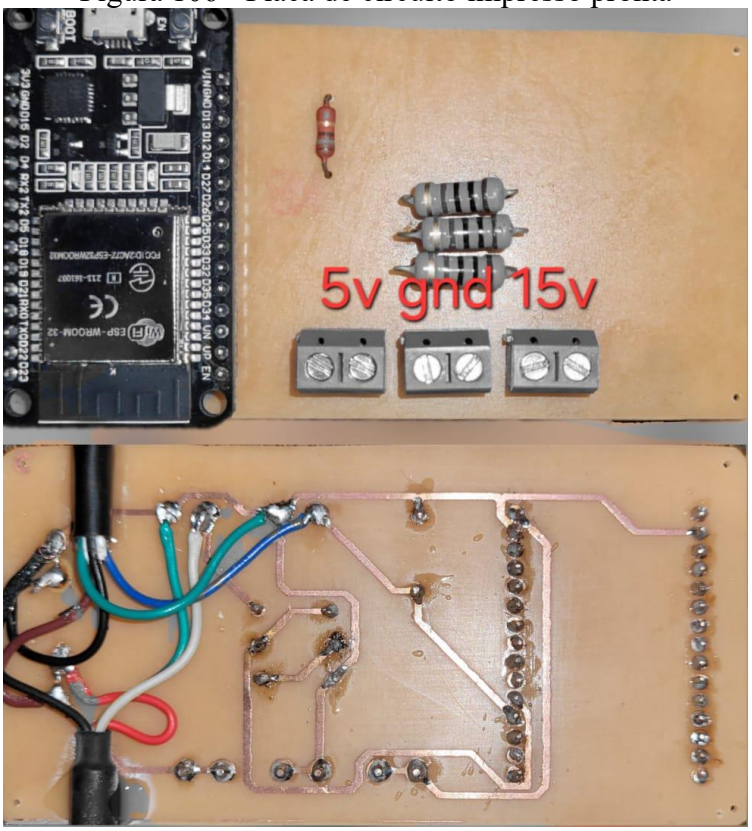
7 RESULTADOS E VALIDAÇÃO

7.1 Placa de circuito impresso

Para que o projeto tivesse uma maior robustez, uma placa de circuito impresso foi confeccionada para acomodar os sensores e o microcontrolador ESP32.Optou-se por soldar os cabos de alimentação e de dados diretamente na placa, pois a bitola dos mesmos era muito pequena. Podendo causar desconexões frequentes durante o transporte.

Já para alimentação geral por parte da fonte foram utilizados bornes duas vias, como visto na Figura 106.

Figura 106 - Placa de circuito impresso pronta



Fonte: Autoria própria

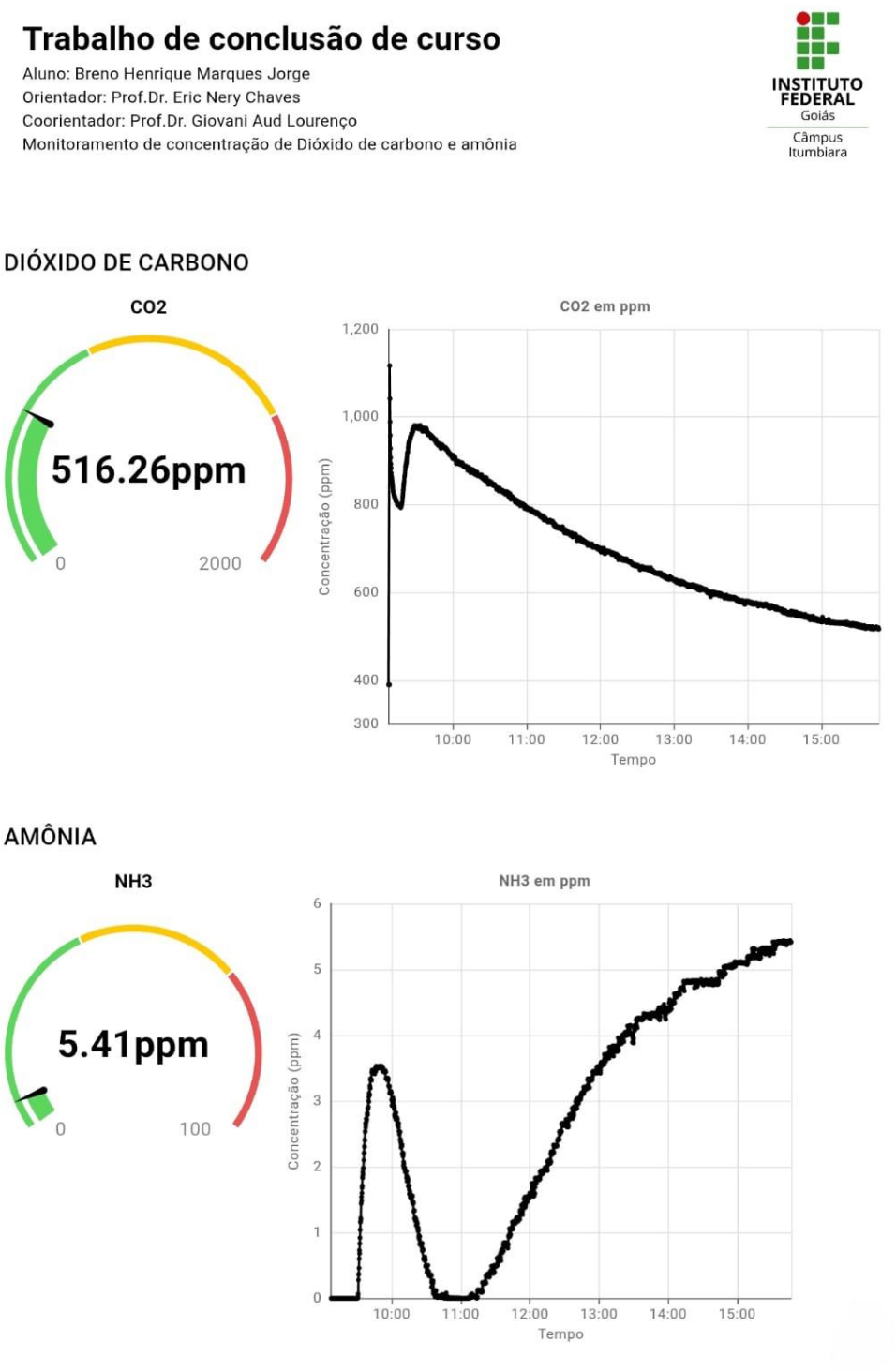
7.2 Dashboards

Trata-se basicamente da interface gráfica para exibição dos dados provenientes dos sensores. Divididas para leituras instantâneas e outra para leituras armazenadas localmente. As mesmas podem ser vistas na Figura 107,Figura 108 e Figura 109.

7.2.1 Leituras instantâneas

É composto por um medidor tipo gauge (analógico) e um chart auxiliar (gráfico) para acompanhamento das últimas 8 horas de leitura(Figura 107). Este é feito de forma dinâmica, não salvando nenhum dado na página caso aconteça qualquer interrupção na alimentação do sistema.

Figura 107 - Resultado final da dashboard dos dados instantâneos

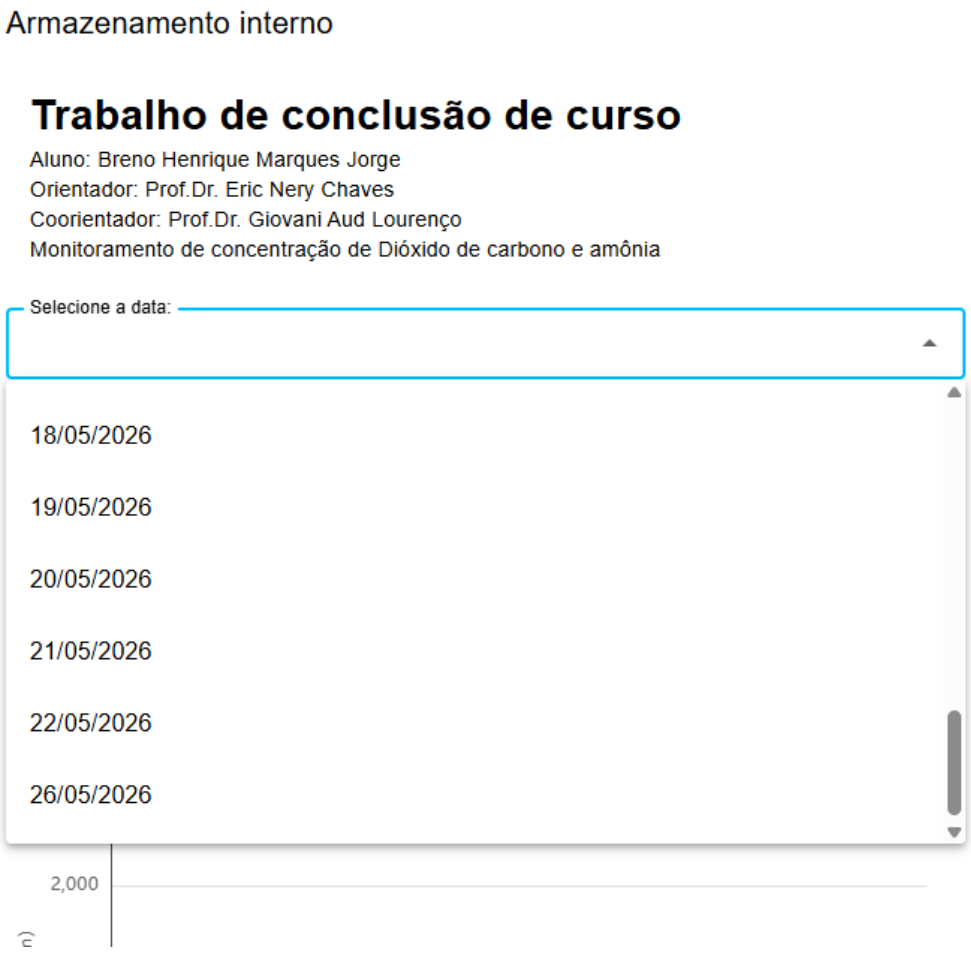


Fonte: Autoria própria

7.2.2 Leituras armazenadas localmente

Caso o usuário deseje imprimir os dados salvos no raspberry no Node-RED, aparecerá as opções de datas disponíveis no dispositivo, como visto na Figura 108.

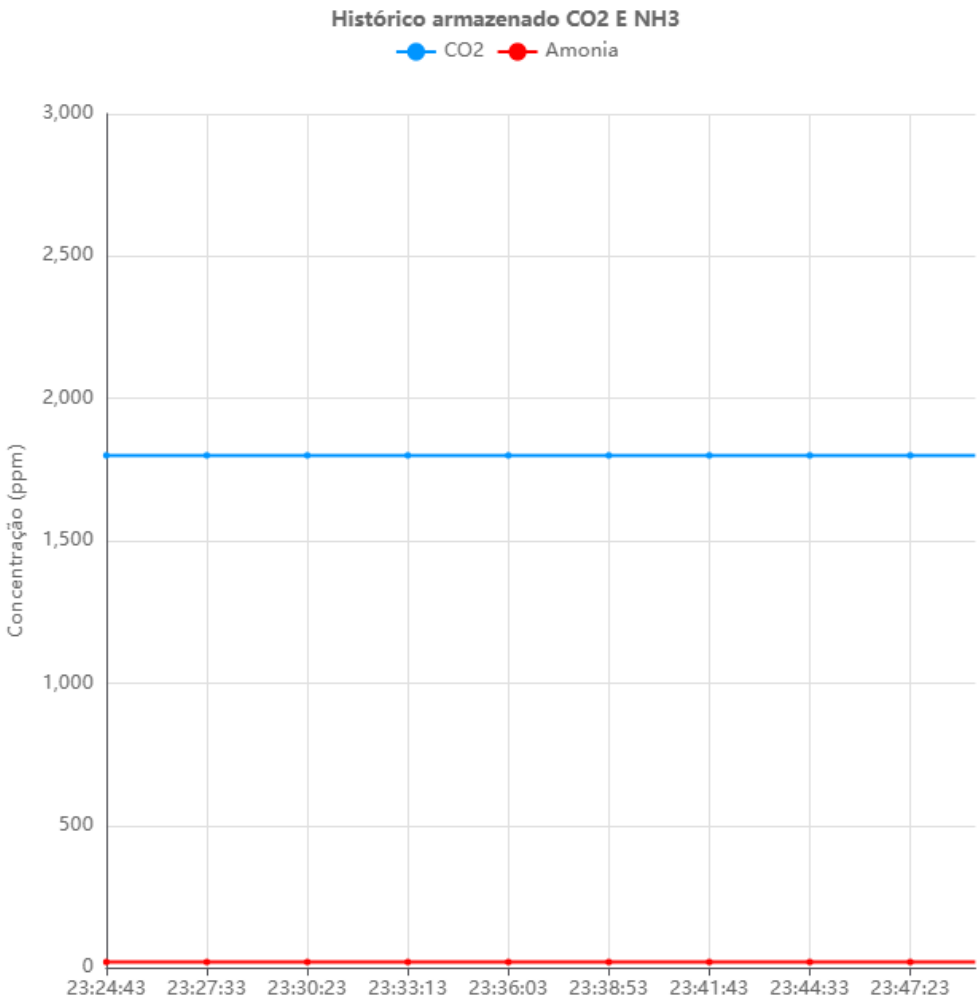
Figura 108 - Escolha dos dados disponíveis no armazenamento interno



Fonte: Autoria própria

Ao selecionar a data desejada, os gráficos de ambos os dados (CO_2 e NH_3) serão impressos em um mesmo gráfico (concentração x tempo), como visto Figura 109:

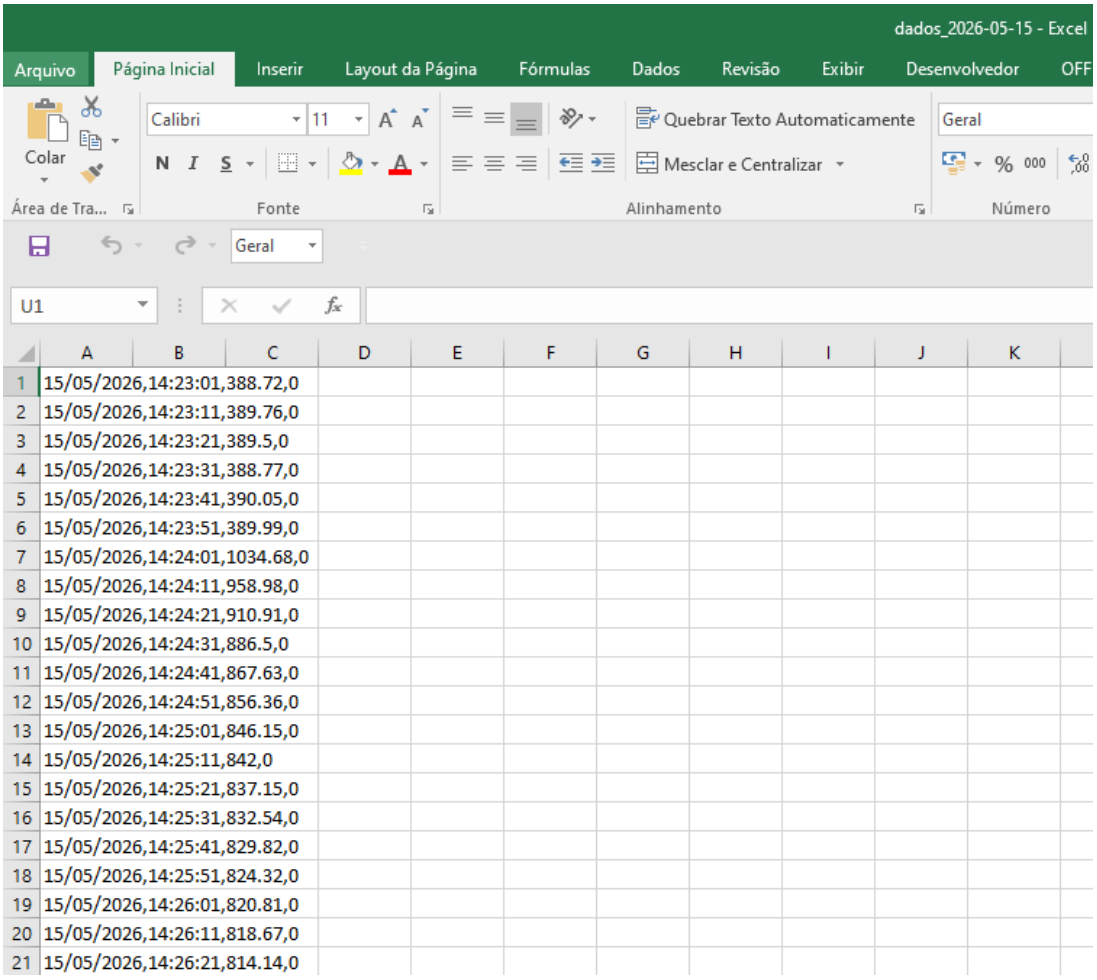
Figura 109 - Amostra de dados armazenados localmente



Fonte: Autoria própria

Além disso, também há a opção de o usuário extrair um arquivo .csv nomeado por data de coleta dos dados. Caso seja essa a escolha, internamente o arquivo conterá ambos valores de concentração (CO_2 e NH_3), tempo (hora, minutos, segundos), e data. Todos separados por vírgula, de forma que a formatação em aplicativos externos (como excel) torne-se mais fácil, como visto na Figura 110:

Figura 110 - Forma no qual os dados são salvos para extração



Fonte: Autoria própria

7.3 Validação experimental do sistema IoT para monitoramento de NH_3 e CO_2

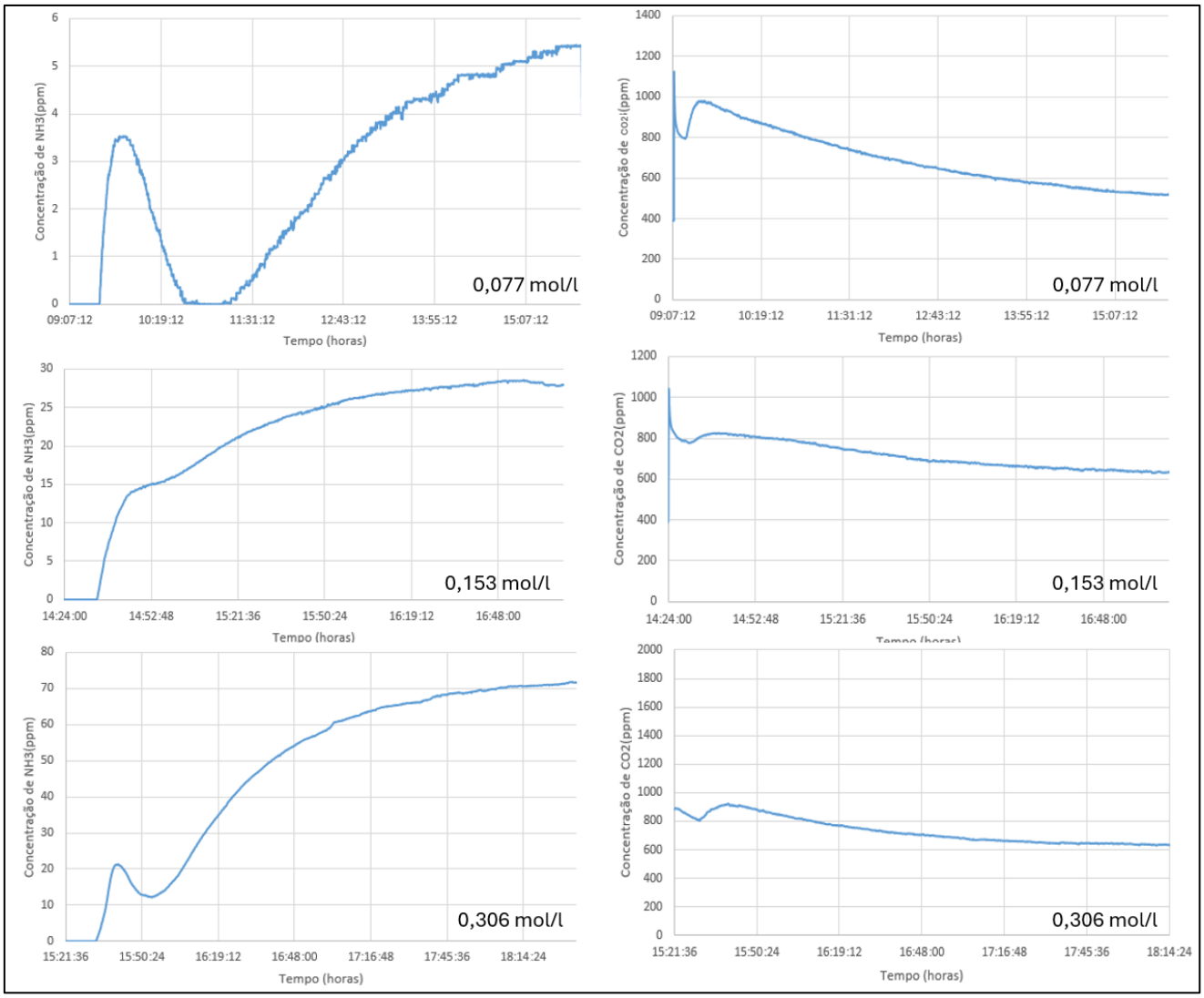
Com base na metodologia apresentada no Capítulo 5, esta seção apresenta os resultados obtidos na validação experimental do sistema IoT desenvolvido para o monitoramento das concentrações de NH_3 e CO_2. A análise foi realizada a partir do comportamento dinâmico das concentrações medidas ao longo do tempo, considerando os ensaios conduzidos em sistema fechado e as diferentes condições experimentais avaliadas. Dessa forma, busca-se verificar a resposta da plataforma de monitoramento em condições controladas, relacionadas a ambientes confinados, como o caso de um contêiner destinado ao cultivo de plantas.

7.3.1 Análise dinâmica do aparato experimental desenvolvido

A Figura 111 apresenta os gráficos do comportamento dinâmico das concentrações de NH_3 e CO_2 medidas no interior do aparato experimental, em função do tempo. No eixo y estão representadas as concentrações dos gases monitorados, em ppmv, enquanto o eixo x apresenta o tempo de realização dos ensaios. Foram avaliadas três condições experimentais, preparadas pela diluição de diferentes volumes de hidróxido de amônio em água, sempre totalizando 100 mL de solução: 0,5 mL, 1,0 mL e 2,0 mL de hidróxido de amônio, correspondendo, respectivamente, às concentrações aproximadas de 0,077 $mol \cdot L^{-1}$, 0,153 $mol \cdot L^{-1}$ e 0,306 $mol \cdot L^{-1}$ de amônia na solução, como apresentado na seção 6.3.

Considerando a sensibilidade do sensor de NH_3 à umidade, os ensaios foram conduzidos em intervalos máximos de aproximadamente 6 horas, tempo suficiente para acompanhar a elevação da concentração de amônia na fase gasosa e sua tendência de estabilização nas condições estudadas. Embora o sistema também tenha monitorado o CO_2, a análise principal foi concentrada no NH_3, pois esse gás estava diretamente relacionado à solução preparada e ao objetivo de validação do sistema.

Figura 111 - Curvas experimentais de concentração de NH_3 e CO_2 no aparato experimental desenvolvido para soluções de hidróxido de amônio de 0,077, 0,153 e 0,306 $mol \cdot L^{-1}$.



Fonte: Autoria própria

Para o NH_3, observa-se um comportamento semelhante nas diferentes concentrações avaliadas, caracterizado por uma elevação inicial mais acentuada, seguida de oscilações e posterior aumento gradual até a tendência de estabilização. Essa resposta indica que a liberação de amônia da solução para a fase gasosa não resulta, nos instantes iniciais, em uma distribuição homogênea no interior da caixa. Assim, as variações observadas nas curvas podem ser associadas à formação e ao deslocamento de plumas gasosas de NH_3, que atingem temporariamente a região próxima ao sensor. Como o sensor mede a concentração em um ponto específico do aparato, e não a média de todo o volume interno, a posição do sensor em relação ao

recipiente contendo a solução influencia diretamente a leitura registrada, principalmente no início dos ensaios.

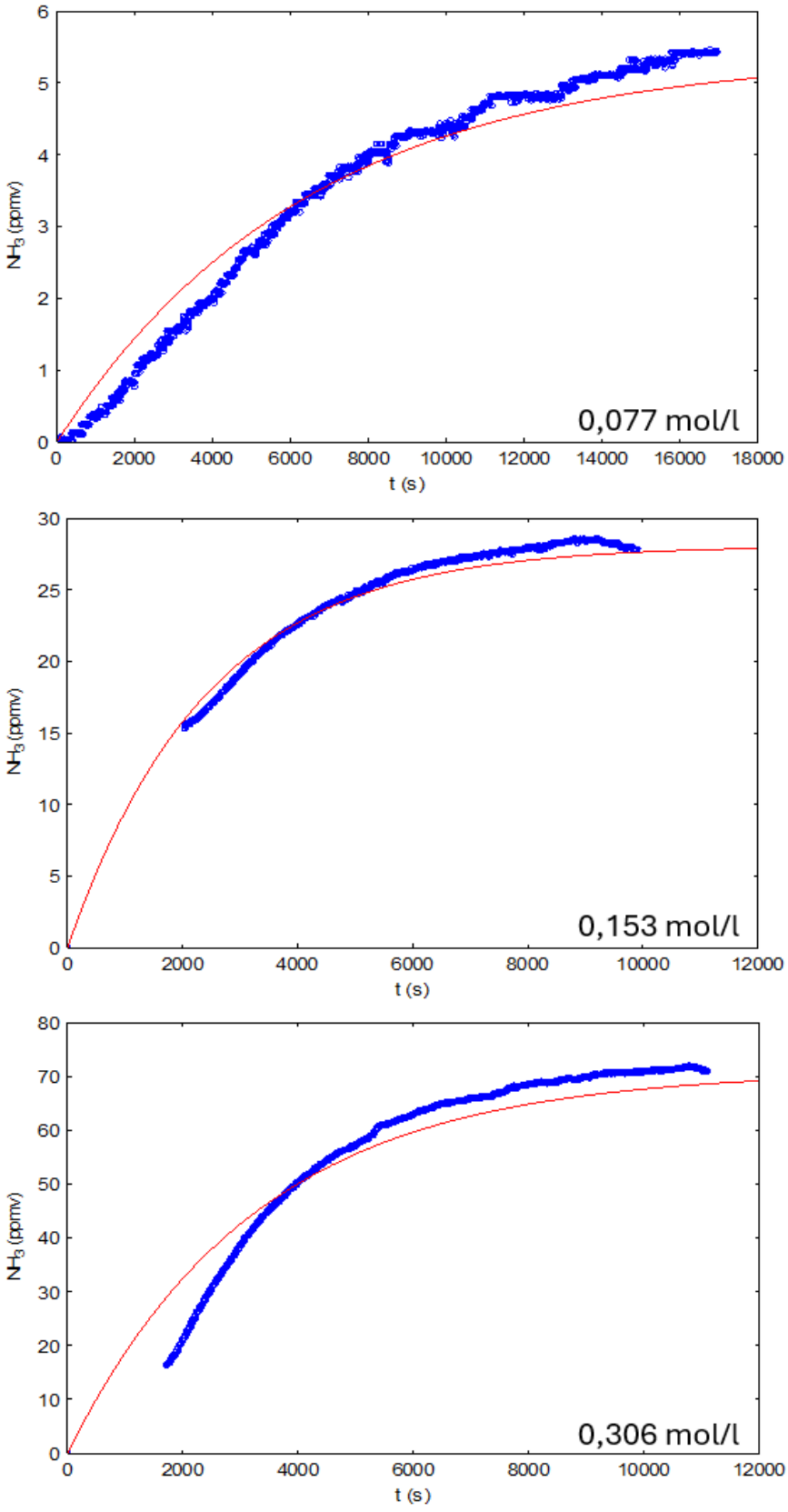
Em relação ao CO_2, observa-se uma tendência geral de redução da concentração ao longo do tempo, comportamento diferente daquele verificado para o NH_3. Essa redução pode estar associada à alteração da composição da fase gasosa, uma vez que o aumento da pressão parcial de NH_3 contribui para modificar as frações relativas dos gases presentes no volume fechado. Além disso, o CO_2 presente no ar confinado pode ser transferido para a fase líquida, devido à sua solubilidade em água, sendo esse processo favorecido pelo caráter alcalino da solução de hidróxido de amônio. Nessas condições, parte do CO_2 dissolvido pode reagir no meio líquido, formando espécies como bicarbonatos e carbonatos, o que contribui para a diminuição da concentração medida na fase gasosa. Portanto, os gráficos indicam que o aparato foi capaz de registrar tanto a dinâmica de liberação e acúmulo de NH_3 quanto a variação simultânea do CO_2 no ambiente interno, evidenciando a sensibilidade dos sensores às mudanças físico-químicas ocorridas no sistema fechado.

7.3.2 Análise dinâmica da liberação de NH_3 e estimativa de parâmetros aparentes do sistema

Com base na metodologia apresentada nas Seções 6.2 e 6.2.1, este tópico apresenta o ajuste não linear das curvas experimentais de NH_3, conforme a Eq. 6(modelo dinâmico de 1° ordem), e o cálculo do fluxo aparente inicial de amônia, conforme a Eq. 9. A análise foi realizada para as soluções de hidróxido de amônio de 0,077 $mol \cdot L^{-1}$, 0,153 $mol \cdot L^{-1}$ e 0,306 $mol \cdot L^{-1}$, permitindo determinar os parâmetros k e Japp,0. Nesse contexto, k representa a velocidade aparente de aproximação da concentração de NH_3 ao patamar de estabilização, enquanto Japp,0 indica o fluxo aparente inicial de acúmulo de amônia na fase gasosa. A determinação desses parâmetros é relevante para a verificação do sistema IoT desenvolvido, pois permite avaliar quantitativamente se a plataforma é capaz de registrar diferenças entre as condições experimentais e acompanhar a dinâmica de liberação e acúmulo de NH_3 no interior do aparato.

Antes da realização dos ajustes, os pontos experimentais associados à formação e ao deslocamento inicial das plumas gasosas foram retirados das curvas. Essa remoção foi necessária porque esses pontos representam variações locais de concentração próximas ao sensor, e não necessariamente o comportamento médio da concentração de NH_3 no volume interno da caixa. Dessa forma, o ajuste não linear foi aplicado à região da curva que melhor representa a tendência global de acúmulo e estabilização da amônia na fase gasosa, reduzindo a influência dos picos e quedas transitórias observados nos instantes iniciais dos ensaios. A Figura 112 apresenta os gráficos com os ajustes realizados.

Figura 112 - Ajuste não linear das curvas experimentais de concentração de NH_3 para soluções de hidróxido de amônio de 0,077, 0,153 e 0,306 $mol \cdot L^{-1}$



Fonte: Autoria própria

Observa-se, na Figura 112, que o modelo de primeira ordem (Eq.6), descreve de forma adequada a tendência geral de aumento da concentração de NH_3 ao longo do tempo. Em todas as condições avaliadas, a concentração apresenta comportamento crescente, seguido de aproximação gradual a um patamar de estabilização. Esse comportamento é compatível com o acúmulo progressivo de amônia na fase gasosa do sistema fechado. As diferenças entre os dados experimentais e as curvas ajustadas podem ser atribuídas à própria dinâmica do aparato, principalmente à ausência de mistura forçada, à medição em um ponto específico da caixa e à distribuição não totalmente homogênea do gás no volume interno.

Após a obtenção dos ajustes, os parâmetros k e $J_{app,0}$ foram organizados na Tabela 2, juntamente com as informações auxiliares do ajuste, para permitir a comparação entre os ensaios.

Tabela 2 – Parâmetros para análise de transferência de massa e do ajuste não linear.

Ajuste	**E1 (0,077 mol/l)**	**E2 (0,153 mol/l)**	**E3 (0,306 mol/l)**
C_∞	5,4 ppmv	28,1 ppmv	71 ppmv
k (s^{-1})	$1{,}56 \times 10^{-4}$	$4{,}13 \times 10^{-4}$	$3{,}06 \times 10^{-4}$
R (Coeficiente de Correlação)	0,9808	0,9831	0,9565
$\left(\frac{dC_g}{dt}\right)_{t=0}$ (Eq. 8)	0,00078 ppmv·s^{-1}	0,011536 ppmv·s^{-1}	0,02142 ppmv·s^{-1}
$J_{ap,0}$ (Eq. 9)	$1{,}05\times10^{-6}$ g·m^{-2}·s^{-1}	$1{,}55 \times 10^{-5}$ g·m^{-2}·s^{-1}	$2{,}82 \times 10^{-5}$ g·m^{-2}·s^{-1}

A partir da Tabela 2, observa-se que o parâmetro k, expresso em s^{-1}, não apresentou aumento diretamente proporcional à concentração da solução. Esse comportamento reforça sua interpretação como uma constante aparente do sistema, associada à velocidade com que a concentração de NH_3 se aproxima do patamar de estabilização. A variação de k pode estar relacionada às condições internas do aparato, como a posição do sensor, a formação de regiões localmente mais concentradas e a mistura não forçada da fase gasosa na caixa. Apesar dessas influências, os coeficientes de correlação obtidos indicam boa aderência do ajuste não linear à tendência experimental dos dados.

Por outro lado, o fluxo aparente inicial $J_{app,0}$ apresentou aumento com a elevação da concentração da solução de hidróxido de amônio. Esse comportamento é coerente com a condição experimental imposta, pois, soluções mais concentradas tendem a disponibilizar maior quantidade de NH_3 para transferência à fase gasosa. Dessa forma, o aumento de Japp,0 indica que o sistema IoT foi sensível às diferentes condições de liberação de amônia, registrando maiores taxas aparentes de acúmulo nos ensaios com maior concentração da solução.

Portanto, os resultados da modelagem dinâmica indicam que os parâmetros obtidos devem ser tratados como parâmetros aparentes do aparato experimental, e não como coeficientes universais de transferência de massa. Mesmo assim, a análise é importante para a validação do sistema IoT, pois demonstra que a plataforma foi capaz de diferenciar as condições experimentais avaliadas, registrar a evolução temporal da concentração de NH_3 e fornecer parâmetros quantitativos para comparar a resposta do sistema em diferentes concentrações de solução.

7.4 Observações acerca dos experimentos

Inicialmente conhecia-se a vulnerabilidade do sensor eletroquímico a umidade (NH_3), porém não se imaginava que o mesmo era tão sensível a tal variável. Seria realizado mais uma batelada de experimentos com as mesmas três concentrações, em 24horas de coleta, porém ao final do primeiro teste, o sensor de NH_3 parou de funcionar completamente, não sendo possível continuar.

Foi possível observar que todo o experimental ultrapassou os 95% de umidade previstos pelo fabricante, pois acumulou condensação nas paredes do aparato experimental. É possível que a membrana que separa os dois eletrodos do sensor possa ter falhado e oxidado o eletrodo de referência, danificando-o.

Este problema ocorreu em recipientes fechados, sendo assim, a aplicação deste para sistema aberto, como contêiners seria mais indicado.

8 CONCLUSÃO

O presente trabalho teve como objetivo desenvolver e validar funcionalmente uma plataforma baseada em Internet das Coisas para o monitoramento contínuo das concentrações de dióxido de carbono (CO_2) e amônia (NH_3), com possibilidade de aplicação em ambientes agrícolas controlados, como sistemas de cultivo vertical em contêineres. A arquitetura proposta, composta por um microcontrolador ESP32 como unidade de aquisição de dados, um Raspberry Pi 3B como servidor local, comunicação via protocolo MQTT e interface gráfica desenvolvida no Node-RED, mostrou-se funcional para a aquisição, transmissão, visualização e armazenamento dos dados experimentais.

A plataforma desenvolvida permitiu a construção de dashboards para acompanhamento em tempo real das concentrações medidas, bem como o armazenamento local dos dados em arquivos CSV, possibilitando a análise posterior do comportamento temporal dos gases monitorados. Dessa forma, o sistema atendeu ao propósito de integrar hardware, software e comunicação de dados em uma solução de monitoramento ambiental de baixo custo e com potencial de aplicação em ambientes confinados.

A validação experimental com amônia, realizada a partir de soluções de hidróxido de amônio em diferentes concentrações, permitiu avaliar a resposta dinâmica do sistema frente à liberação controlada de NH_3 para a fase gasosa. Os resultados indicaram que a plataforma foi capaz de registrar diferenças entre as condições experimentais avaliadas, com aumento dos valores de concentração de estabilização e do fluxo aparente inicial conforme a concentração da solução foi elevada. Os ajustes não lineares apresentaram boa aderência à tendência experimental dos dados, com coeficientes de correlação satisfatórios, demonstrando que a plataforma consegue acompanhar a evolução temporal da concentração de NH_3 no aparato experimental.

Entretanto, os parâmetros obtidos, como a constante aparente de velocidade e o fluxo aparente inicial de transferência de massa, devem ser interpretados como parâmetros aparentes do aparato experimental, e não como coeficientes universais de transferência de massa. Essa limitação decorre das condições específicas do ensaio, como ausência de mistura forçada, posição fixa do sensor, formação de

regiões localmente mais concentradas e distribuição não homogênea do gás no interior da caixa. Assim, os resultados são adequados para comparação entre os ensaios realizados no mesmo sistema, mas não devem ser extrapolados diretamente para outros volumes ou geometrias sem nova validação experimental.

Também foi observado que o sensor eletroquímico de amônia apresentou elevada sensibilidade à umidade. Durante os ensaios, a umidade relativa no interior do aparato ultrapassou o limite operacional previsto pelo fabricante, resultando em condensação nas paredes da caixa e posterior falha do sensor de NH_3. Esse comportamento interrompeu a continuidade dos ensaios de longa duração e evidenciou que, embora a plataforma IoT tenha se mostrado funcional, a aplicação prática em ambientes úmidos não é possível com este sensor. Para correção destes problemas em trabalhos futuros, recomenda-se o uso de volumes menores de solução de hidróxido de amônio, associado a um volume maior do ambiente confinado.

Assim, foi possível concluir que a plataforma desenvolvida representa uma contribuição relevante para o monitoramento de gases em ambientes controlados, pois demonstrou capacidade de aquisição, transmissão, armazenamento e visualização dos dados em tempo real. Além disso, a metodologia experimental proposta permitiu avaliar a resposta dinâmica do sistema para diferentes condições de liberação de amônia. Contudo, para aplicação em sistemas reais de agricultura vertical, ou para monitoramento ambiental, ainda é necessário a calibração com padrões de referência.

Como perspectivas para trabalhos futuros, recomenda-se o desenvolvimento de um aparato experimental de baixo custo que possibilite a calibração dos sensores de CO_2 e NH_3, aumentando a precisão e a confiabilidade dos dados obtidos. Também se propõe a integração da plataforma a sistemas de controle, como exaustores, alarmes, sistemas de renovação de ar e acionamento controlado de fontes de CO_2, permitindo manter a concentração desse gás em faixas adequadas para contêiners destinados ao cultivo de plantas. Além disso, o desenvolvimento de um sistema de decomposição térmica controlada do bicarbonato de amônio, capaz de simular condições mais próximas às encontradas em um contêiner de agricultura vertical, promovendo a geração controlada de CO_2 e o devido tratamento do NH_3 produzido.

REFERÊNCIAS

[1] PEREIRA, Antonio. Agricultura vertical: o futuro da produção agrícola. Agropecnews, 15 jul. 2024. Disponível em: https://www.agropecnews.com.br/agricultura-vertical-o-futuro-da-producao-agricola/.Acesso em: 01 dez. 2025.

[2] Silveira, A. M. F.; Marenco, R. A. (2023). O CO2 elevado induz a regulação negativa da fotossíntese e alivia o efeito do déficit hídrico em Ceiba pentandra (Malvaceae). *Revista Árvore*, 47, Disponível em: https://revistaarvore.ufv.br/rarv/article/view/263496. Acesso em 08 dez.2025.

[3] Assessment of air quality in poultry facilities/ Avaliação da qualidade do ar em instalações agrícolas.(2018). *Caderno De Ciências Agrárias*, 10(1), 68-72. Disponível em: https://periodicos.ufmg.br/index.php/ccaufmg/article/view/3017. Acesso em 08 dez.2025.

[4] WATT CONSULTORIA. Soluções em automação com ESP32 e Raspberry Pi. Watt Consultoria, 2024. Disponível em: https://wattconsultoria.com.br/solucoes-em-automacao-com-esp32-e-raspberry-pi/. Acesso em: 08 dez. 2025.

[5] WEG. O que são sensores industriais e por que eles são importantes para a automação industrial.Blog WEG Digital & Sistemas, 06 fev. 2024. Disponível em: https://www.weg.net/digital/blog/o-que-sao-sensores-industriais/. Acesso em: 08 dez. 2025.

[6] CHINAGALVO. *What is analog and digital signal. Chinagalvo* – News, s.d. Disponível em: http://pt.chinagalvo.com/news/what-is-analog-and-digital-signal-66445726.html. Acesso em: 08 dez. 2025.

[7] RENKEER. *How to Measure Carbon Dioxide (CO2)? Renkeer*, 13 jun. 2025. Disponível em: https://www.renkeer.com/how-to-measure-carbon-dioxide/. Acesso em: 09 dez. 2025.

[8] SENSOR1STOP. Análise de sensores de CO_2 NDIR, semicondutores e condutores térmicos. Sensor1Stop – Knowledge Base, s.d. Disponível em:https://sensor1stop.com/pt/knowledge/analysis-of-ndir-semiconductor-and-thermal-conductor-co2-sensors/. Acesso em: 09 dez. 2025.

[9] WINSEN SENSOR. What is CO_2 sensor. Winsen Sensor – Knowledge, s.d. Disponível em: https://pt.winsen-sensor.com/knowledge/what-is-co2-sensor.html. Acesso em: 10 dez. 2025.

[10] ELSEVIER. Thermal conductivity of gases. ScienceDirect Topics, s.d. Disponível em: https://www.sciencedirect.com/topics/chemical-engineering/thermal-conductivity-of-gases. Acesso em: 10 dez. 2025.

[11] NEWTON C. BRAGA. Sensor de condutividade térmica STC31-C para CO_2 (COMD153). Newton C. Braga – Novos Componentes. Disponível em: https://www.newtoncbraga.com.br/novos-componentes/35997-sensor-de-condutividade-termica-stc31-c-para-co2-comd153.html. Acesso em: 10 dez. 2025.

[12] LIMA, Ana Luiza Lorenzen. Amônia (NH_3): características, precauções. Manual da Química, 2024. Disponível em: https://www.manualdaquimica.com/quimica-inorganica/amonia.htm. Acesso em: 15 dez. 2025.

[13] WINSEN SENSOR. NH_3 sensor: working principle, applications and benefits. Winsen Sensor – Knowledge, s.d. Disponível em: https://pt.winsen-sensor.com/knowledge/nh3-sensor-working-principle-applications-and-benefits.html. Acesso em: 22 dez. 2025.

[14] DOL SENSORS. Ammonia measurement sensors. DOL Sensors – Ammonia Measurement Sensors, s.d. Disponível em: https://www.dol-sensors.com/ammonia-measurement-sensors/. Acesso em: 22 dez. 2025.

[15] AG SOLVE. Detecção de gases: sensores por fotoionização e combustão catalítica. Ag Solve – Notícias, 17 jun. 2016.Disponível em: https://www.agsolve.com.br/noticias/10269/deteccao-de-gases-sensores-por-fotoionizacao-e-combustao-catalitica/. Acesso em: 22 dez. 2025.

[16] GEOACQUA. O que é um detector PID e o que ele mede. GeoAcqua – Blog, s.d. Disponível em: https://www.geoacqua.com.br/o-que-e-um-detector-pid-e-o-que-ele-mede. Acesso em: 22 dez. 2025.

[17] VICTOR VISION. O que é um microcontrolador? Victor Vision – Blog, s.d. Disponível em: https://victorvision.com.br/blog/o-que-e-um-microcontrolador/. Acesso em: 22 dez. 2025.

[18] VICTOR VISION. Placa ESP32: o que é e para que serve? Victor Vision – Blog, s.d. Disponível em: https://victorvision.com.br/blog/placa-esp32/. Acesso em: 22 dez. 2025.

[19] PROMOBIT. O que é um mini PC e para que serve esse tipo de computador. Promobit – Blog, s.d. Disponível em: https://www.promobit.com.br/blog/o-que-e-um-mini-pc-e-para-que-serve-esse-tipo-de-computador/. Acesso em: 22 dez. 2025.

[20] LSTECNOLOGIA. Micro/Mini PC – Mini CPU Intel Dual Core SSD Wi-Fi NF-E. LSTecnologia – Produtos. Disponível em: https://www.lstecnologia.net/micro-mini-pc-mini-cpu-intel-dual-core-ssd-wifi-nf-e. Acesso em: 22 dez. 2025.

[21] PI MY LIFE UP. The Different Versions of the Raspberry Pi. Atualizado em: 24 jul. 2024. Disponível em: https://pimylifeup.com/raspberry-pi-versions/. Acesso em: 22 dez. 2025.

[22] SHIMABUKURO, Igor; TOLEDO, Victor. O que é o Raspberry Pi? Saiba para que serve e como funciona o minicomputador. Tecnoblog, 2025. Disponível em: https://tecnoblog.net/responde/o-que-e-o-raspberry-pi/. Acesso em: 22 dez. 2025.

[23] VIRÍSSIMO, Gabriel. Desenvolvimento de um sistema de monitoramento de gases para cultivo agrícola em ambiente controlado utilizando ESP32 e Raspberry Pi. 2024. Trabalho de Conclusão de Curso (Bacharelado em Engenharia) — Instituto Federal de educação, ciência e tecnologia de Goiás, Curso de Engenharia Elétrica, Itumbiara, 2024.

[24] MAGNA SISTEMAS. Editores de código-fonte. Doc Magna Sistemas – Arquitetura / Ferramentas. Disponível em: https://doc.magnasistemas.com.br/arquitetura/ferramentas/editores-codigo-fonte/. Acesso em: 22 dez. 2025.

[25] MQTT. MQTT – *MQ Telemetry Transport: The Standard for IoT Messaging.* Disponível em: https://mqtt.org/. Acesso em: 22 dez. 2025.

[26] XAVIER, João. Protocolo MQTT: um pilar da comunicação em IoT. LinkedIn Pulse, s.d. Disponível em: https://pt.linkedin.com/pulse/protocolo-mqtt-um-pilar-da-comunica%C3%A7%C3%A3o-em-iot-jxmsf. Acesso em: 22 dez. 2025.

[27] HOSTGATOR BRASIL. O que é Node-RED? HostGator Blog, 2025. Disponível em: https://www.hostgator.com.br/blog/o-que-e-node-red/. Acesso em: 8 jan. 2026.

[28] NODE-RED. Node-RED – Flow-based development tool for visual programming. Disponível em: https://nodered.org/. Acesso em: 08 jan. 2026.

[29] WIFEED. O que é IoT? Conheça a Internet das Coisas. Wifeed – Conteúdo, s.d. Disponível em: https://www.wifeed.com.br/conteudo/o-que-e-iot-conheca-internet-das-coisas/. Acesso em: 08 jan. 2026.

[30] ORACLE. Internet of Things (IoT) – Internet das Coisas. Oracle Brasil. Disponível em: https://www.oracle.com/br/internet-of-things/. Acesso em: 08 jan. 2026.

[31] USINA INFO. ESP32 Wi-Fi: comunicação com a Internet. Usina Info – Blog, s.d. Disponível em: https://www.usinainfo.com.br/blog/esp32-wifi-comunicacao-com-a-internet/. Acesso em: 09 jan. 2026.

[32] GOOGLE. Wi-Fi STA + AP *Concurrency. Android Developers* — Documentação Android.Disponível em: https://source.android.com/docs/core/connect/wifi-sta-ap-concurrency?hl=pt-br. Acesso em: 09 jan. 2026.

[33] ELETROGATE. Automação residencial com broker MQTT local com Raspberry Pi. Blog Eletrogate, s.d. Disponível em: https://blog.eletrogate.com/automacao-residencial-com-broker-mqtt-local-com-raspberry-pi/. Acesso em: 12 jan. 2026.

[34] BRAGA, Newton C. Como funcionam os sensores de gases tóxicos (ART2860). Newton C. Braga, 2007. Disponível em: https://www.newtoncbraga.com.br/index.php/como-funciona/12105-como-funcionam-os-sensores-de-gases-toxicos-art2860. Acesso em: 10 jun. 2026.

[35] GEANKOPLIS, Christie J. Transport Processes and Unit Operations. 3. ed. Englewood Cliffs: Prentice Hall, 1993.

[36] KUTZBACH, L. et al. CO_2 flux determination by closed-chamber methods can be seriously biased by inappropriate application of linear regression. Biogeosciences, v. 4, p. 1005-1025, 2007.

[37] MONTES, F.; ROTZ, C. A.; CHAOUI, H. Process modeling of ammonia volatilization from ammonium solution and manure surfaces: a review with recommended models. Transactions of the ASABE, v. 52, n. 5, p. 1707-1719, 2009.

APÊNDICE A – CÓDIGO DA CONEXÃO WIFI NO ESP32

```
// ---------------- Declarações para conexão Wi-Fi ---------------- //
const char* ssid  = "NOME_DA_REDE";   // Nome da rede Wi-Fi (SSID)
const char* password = "SENHA_DA_REDE";  // Senha da rede Wi-Fi
const int Led_Azul = 2;              // Pino GPIO do LED indicador
// ------------------------------------------------------------ //
// ----------- Função responsável pela conexão Wi-Fi ------------- //
void Conectar_WiFi() {
    if (WiFi.status() == WL_CONNECTED) // Verifica se o ESP32 já está conectado à rede Wi-Fi
     return;
    digitalWrite(Led_Azul, LOW); // Desliga o LED enquanto não houver conexão
   // Exibição de informações no monitor serial
   Serial.println();
   Serial.print("Endereco MAC do dispositivo: ");
   Serial.println(WiFi.macAddress());
   Serial.print("Tentando conectar a rede Wi-Fi: ");
   Serial.println(ssid);
   // Inicia a conexão com a rede Wi-Fi
   WiFi.begin(ssid, password);
   // Aguarda até que a conexão seja estabelecida
   while (WiFi.status() != WL_CONNECTED) {
     delay(500);
```

```
        Serial.print(".");
    }
    // Acende o LED indicando conexão estabelecida
    digitalWrite(Led_Azul, HIGH);
    // Mensagens de confirmação
    Serial.println();
    Serial.print("Conectado com sucesso a rede Wi-Fi: ");
    Serial.println(ssid);
    Serial.print("Endereco IP atribuido ao dispositivo: ");
    Serial.println(WiFi.localIP());
    Serial.println();
}
// ---------------------------------------------------------- //
// ---------------------- Função setup ---------------------- //
void setup() {
    // Inicializa a comunicação serial
    Serial.begin(115200);
    // Define o pino do LED como saída
    pinMode(Led_Azul, OUTPUT);
}
// ---------------------------------------------------------- //
// ---------------------- Função loop ----------------------- //
 void loop() {
    // Garante que o dispositivo permaneça conectado à rede Wi-Fi
```

```
    Conectar_WiFi();
}
// ------------------------------------------------------------ //
```

APÊNDICE B – CÓDIGO DA CONEXÃO DO BROKER NO ESP32

const char* MQTT_ID = "ESP32-PUBLICADOR"; // Identificador único do cliente MQTT

const char* MQTT_Broker = "192.168.1.187"; // Endereço IP do broker MQTT (servidor)

const int MQTT_Porta = 1883; // Porta utilizada pelo protocolo MQTT (1883 = padrão sem criptografia)

const char* MQTT_Usuario = "usuario"; // Nome de usuário para autenticação no broker MQTT

const char* MQTT_Senha = "senha"; // Senha para autenticação no broker MQTT

const char* Topico_Amonia = "Amonia"; // Categoria MQTT para envio da concentração de amônia (NH3)

const char* Topico_CO2 = "CO2"; // Categoria MQTT para envio da concentração de dióxido de carbono (CO2)

WiFiClient espClient; // Cria um cliente Wi-Fi para comunicação TCP/IP

PubSubClient MQTT(espClient); // Cria o cliente MQTT utilizando a conexão Wi-Fi

// Vetor de caracteres para armazenar o valor da amônia e dioxído de carbono em formato texto

char Amonia[10];

char CO2[10];

//--------- Função para conexão MQTT --------//

void Conectar_MQTT() // Função responsável por conectar o ESP32 ao broker MQTT

```
{
  MQTT.setServer(MQTT_Broker, MQTT_Porta); // Define o endereço e a porta do broker MQTT
  while (!MQTT.connected()) // Enquanto o ESP32 não estiver conectado ao broker
{
  // Exibe mensagem de tentativa de conexão
  Serial.print("Tentando conexão com o Broker MQTT: ");
  Serial.println(MQTT_Broker);
  // Tenta conectar ao broker usando ID, usuário e senha
  if (MQTT.connect(MQTT_ID, MQTT_Usuario, MQTT_Senha))
{
    // Mensagem exibida quando a conexão é bem-sucedida
    Serial.println("Conectado ao broker MQTT!");
    Serial.println();
  }
  else {
    // Exibe mensagem de erro caso a conexão falhe
    Serial.print("Falha na conexão. Codigo de erro: ");
    Serial.println(MQTT.state());
    // Informa que uma nova tentativa será realizada
    Serial.println("Nova tentativa em 5 segundos.");
    Serial.println();
    // Aguarda 2 segundos antes de tentar novamente
    delay(2000);
```

```
    }
  }
}
//----------------------------------------//
//------ Função para publicar no broker -----//
// Função responsável por enviar os dados via MQTT
void Publicar_MQTT() {
  // Converte o valor numérico da concentração de amônia para String
  String str1 = String(ConcentracaoAmonia);
  // Converte o valor numérico da concentração de CO2 para String
  String str2 = String(ConcentracaoCO2);
  // Converte a String de amônia para vetor de caracteres (char[])
  str1.toCharArray(Amonia, 10);
  // Converte a String de CO2 para vetor de caracteres (char[])
  str2.toCharArray(CO2, 10);
  // Publica o valor da amônia no tópico correspondente
  MQTT.publish(Topico_Amonia, Amonia);
  // Publica o valor do CO2 no tópico correspondente
  MQTT.publish(Topico_CO2, CO2);
}
//----------------------------------------//
// Função setup: executada apenas uma vez ao iniciar o ESP32
void setup() {
  // Inicializa a comunicação serial para monitoramento
```

```
  Serial.begin(115200);
}
// Função loop: executada continuamente
void loop() {
  // Garante que o ESP32 esteja conectado ao broker MQTT
  Conectar_MQTT();
  // Publica os dados dos sensores no broker
  Publicar_MQTT();
  // Define um intervalo entre as publicações
  delay(2000);
}
```

APÊNDICE C – CÓDIGO COMPLETO DO ESP32

```
#include <Arduino.h>
#include <WiFi.h>
#include <PubSubClient.h>
#include <ESPmDNS.h>
// ------------------------------------------------
// DADOS WI-FI
// ------------------------------------------------
const char* ssid  = "tccbirosca";
const char* senha = "537280cr";
// LED indicador
const int Led_Azul = 2;
// ------------------------------------------------
// DADOS MQTT
// ------------------------------------------------
const char* MQTT_ID       = "ESP32-PUBLICADOR";
const char* MQTT_Broker   = "10.0.0.3";
const int   MQTT_Porta    = 1883;
const char* MQTT_Usuario  = "conteiner";
const char* MQTT_Senha    = "171201";
const char* Topico_Dados  = "conteiner/gases";
// ------------------------------------------------
// PINOS DOS SENSORES
```

```
// ------------------------------------------------
#define PIN_CO2     34
#define PIN_AMONIA  35
// ------------------------------------------------
// VARIÁVEIS GLOBAIS
// ------------------------------------------------
WiFiClient espClient;
PubSubClient MQTT(espClient);
char payload[64];
float Amonia = 0;
float CO2    = 0;
// ------------------------------------------------
// CONTROLE DE TEMPO
// ------------------------------------------------
unsigned long tempoAnterior = 0;
const unsigned long intervaloEnvio = 10000; // 10 segundos
// ------------------------------------------------
// DECLARAÇÃO DAS FUNÇÕES
// ------------------------------------------------
void Conectar_WiFi();
void Conectar_MQTT();
void Publicar_MQTT();
void Ler_Sensores();
// ------------------------------------------------
```

```
// SETUP
// ------------------------------------------------
void setup() {
  Serial.begin(115200);
  delay(500);
  pinMode(Led_Azul, OUTPUT);
  digitalWrite(Led_Azul, LOW);
  pinMode(PIN_CO2, INPUT);
  pinMode(PIN_AMONIA, INPUT);
  Conectar_WiFi();
  IPAddress ip;
if (WiFi.hostByName(MQTT_Broker, ip)) {
  MQTT.setServer(ip, MQTT_Porta);
  Serial.println("Broker resolvido via mDNS:");
  Serial.println(ip);
} else {
  Serial.println("Falha ao resolver mDNS");
}
}
// ------------------------------------------------
// LOOP
// ------------------------------------------------
void loop() {
  Conectar_WiFi();
```

```
  Conectar_MQTT();
  MQTT.loop();
  unsigned long agora = millis();
  if (agora - tempoAnterior >= intervaloEnvio) {
    tempoAnterior = agora;
    Ler_Sensores();      // <<< LEITURA REAL
    Publicar_MQTT();     // <<< ENVIO MQTT
  }
}
// ------------------------------------------------
// FUNÇÃO: LEITURA DOS SENSORES (MÉDIA 50)
// ------------------------------------------------
void Ler_Sensores() {
  const int numAmostras = 50;
  long somaCO2 = 0;
  long somaAmonia = 0;
  for (int i = 0; i < numAmostras; i++) {
    somaCO2 += analogRead(PIN_CO2);
    somaAmonia += analogRead(PIN_AMONIA);
    delay(5);
  }
  float mediaADC_CO2 = somaCO2 / (float)numAmostras;
  float mediaADC_Amonia = somaAmonia / (float)numAmostras;
  // Converte ADC (0-4095) para tensão (0-3.3V)
```

```
  float tensaoCO2 = (mediaADC_CO2 * 3.3) / 4095.0;
  float tensaoAmonia = (mediaADC_Amonia * 3.3) / 4095.0;
  // ----------- CONVERSÃO CO2 -----------
  // 0.6–3.0V → 0–2000 ppm
  if (tensaoCO2 < 0.6) tensaoCO2 = 0.6;
  if (tensaoCO2 > 3.0) tensaoCO2 = 3.0;
  CO2 = ((tensaoCO2 - 0.6) / (3.0 - 0.6)) * 2000.0;
  // ----------- CONVERSÃO AMÔNIA -----------
  // 0–3.3V → 0–100 ppm
  if (tensaoAmonia < 0) tensaoAmonia = 0;
  if (tensaoAmonia > 3.3) tensaoAmonia = 3.3;
  Amonia = (tensaoAmonia / 3.3) * 100.0;
  Serial.println("Leitura média dos sensores:");
  Serial.print("CO2 (ppm): ");
  Serial.println(CO2);
  Serial.print("Amonia (ppm): ");
  Serial.println(Amonia);
}
// ------------------------------------------------
// FUNÇÃO: CONEXÃO WI-FI
// ------------------------------------------------
void Conectar_WiFi() {
  if (WiFi.status() == WL_CONNECTED)
    return;
```

```
  digitalWrite(Led_Azul, LOW);
  Serial.println();
  Serial.print("Conectando ao Wi-Fi: ");
  Serial.println(ssid);
  WiFi.begin(ssid, senha);
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  digitalWrite(Led_Azul, HIGH);
  Serial.println();
  Serial.println("Wi-Fi conectado com sucesso");
  Serial.print("Endereco IP: ");
  Serial.println(WiFi.localIP());
}
// ------------------------------------------------
// FUNÇÃO: CONEXÃO MQTT
// ------------------------------------------------
void Conectar_MQTT() {
   if (MQTT.connected())
    return;
  Serial.print("Conectando ao Broker MQTT: ");
  Serial.println(MQTT_Broker);
```

```
  if (MQTT.connect(MQTT_ID, MQTT_Usuario, MQTT_Senha)) {
    Serial.println("Conectado ao broker MQTT!");
  }
  else {
    Serial.print("Falha MQTT. Codigo: ");
    Serial.println(MQTT.state());
    Serial.println("Tentará novamente no próximo loop...");
     }
}
// ------------------------------------------------
// FUNÇÃO: PUBLICAÇÃO MQTT
// ------------------------------------------------
void Publicar_MQTT() {
  snprintf(payload, sizeof(payload),
           "{\"Amonia\": %.2f, \"CO2\": %.2f}",
           Amonia,
           CO2);
  MQTT.publish(Topico_Dados, payload);
  Serial.println("JSON enviado via MQTT:");
  Serial.println(payload);
}
```