UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ
PROGRAMA DE PÓS-GRADUAÇÃO EM INFORMÁTICA

ANDRÉ GEORGE LAUER

SISTEMA EM NÉVOA PARA SENSORIAMENTO DE COLÔNIAS DE ABELHAS SEM FERRÃO COM VALIDAÇÃO NA ESPÉCIE MANDAÇAIA OU MELIPONA QUADRIFASCIATA

DISSERTAÇÃO DE MESTRADO

CORNÉLIO PROCÓPIO
2023

ANDRÉ GEORGE LAUER

SISTEMA EM NÉVOA PARA SENSORIAMENTO DE COLÔNIAS DE ABELHAS SEM FERRÃO COM VALIDAÇÃO NA ESPÉCIE MANDAÇAIA OU MELIPONA QUADRIFASCIATA

Fog system to sensing stingless bee colonies and validation in Mandacaia species or Melipona quadrifasciata

Dissertação apresentada ao Programa de Pós-Graduação em Informática da Universidade Tecnológica Federal do Paraná – UTFPR, como requisito parcial para obtenção do título de Mestre em Informática.

Orientadora: Profa. Dra. Natássya Barlate Floro da Silva

CORNÉLIO PROCÓPIO
2023

Ministério da Educação
Universidade Tecnológica Federal do Paraná
Campus Cornélio Procópio

ANDRE GEORGE LAUER

SISTEMA EM NÉVOA PARA SENSORIAMENTO DE COLÔNIAS DE ABELHAS SEM FERRÃO COM VALIDAÇÃO NA ESPÉCIE MANDAÇAIA OU MELIPONA QUADRIFASCIATA

Trabalho de pesquisa de mestrado apresentado como requisito para obtenção do título de Mestre Em Informática da Universidade Tecnológica Federal do Paraná (UTFPR). Área de concentração: Computação Aplicada.

Data de aprovação: 04 de Agosto de 2023

Dra. Natassya Barlate Floro Da Silva, Doutorado - Universidade Tecnológica Federal do Paraná

Daniel Fernando Pigatto, - Universidade Tecnológica Federal do Paraná

Dr. Rafael Augusto Gregati, Doutorado - Universidade Estadual do Centro Oeste (Unicentro)

RESUMO

LAUER, André George. SISTEMA EM NÉVOA PARA SENSORIAMENTO DE COLÔNIAS DE ABELHAS SEM FERRÃO COM VALIDAÇÃO NA ESPÉCIE MANDAÇAIA OU MELIPONA QUADRIFASCIATA. 123 f. Dissertação – Programa de Pós-Graduação em Informática, Universidade Tecnológica Federal do Paraná. Cornélio Procópio, 2023.

No Brasil, a atividade de criação de abelhas sem ferrão pertencentes à tribo Meliponini (Hymenoptera, Apidae), como fonte de renda, tem experimentado um crescimento significativo. São mais de 300 espécies catalogadas no Brasil. Infelizmente, é comum ocorrerem problemas com as colmeias que podem dizimar criações inteiras. No entanto, a maioria das pesquisas realizadas até o ano de 2022 concentram-se na criação de abelhas com ferrão como a *Apis mellifera*, que possuem comportamentos e características distintas das abelhas sem ferrão. Esse trabalho tem como objetivo o desenvolvimento de um sistema de monitoramento de colmeias de abelhas sem ferrão, mais especificamente para a abelha mandaçaia (*Melipona quadrifasciata*). O sistema consiste na implementação de sensores em conjunto com um microcontrolador conectado a uma infraestrutura de névoa, onde é feita a adequação, armazenamento e análise dos dados. Para garantir uma maior tolerância a falhas e um acesso remoto, esses dados são também sincronizados com um serviço em nuvem. Além disso, o sistema foi integrado a uma aplicação Web, que possibilita a visualização dos dados coletados, a identificação de anomalias e o acionamento de gatilhos para auxiliar os criadores nas tomadas de decisão relacionadas à intervenção na colônia monitorada. O sistema desenvolvido possui a capacidade de verificar os valores das grandezas coletadas nas caixas de abelhas sem ferrão, a fim de determinar se estão dentro da faixa considerada normal. Esses valores são apresentados em um dispositivo específico, também parte do sistema. Caso algum dado esteja fora da faixa normal, o sistema é capaz de acionar um alarme no dispositivo, exibindo o valor e a grandeza correspondente que requer verificação.

Palavras-chave: Meliponicultura de precisão, Abelhas Sem Ferrão, Internet das Coisas, Criação de Abelhas, Mandaçaia

ABSTRACT

LAUER, André George. Fog system to sensing stingless bee colonies and validation in Mandacaia species or Melipona quadrifasciata. 123 f. Dissertação – Programa de Pós-Graduação em Informática, Universidade Tecnológica Federal do Paraná. Cornélio Procópio, 2023.

In Brazil, the activity of breeding stingless bees belonging to the Meliponini tribe (*Hymenoptera, Apidae*), as a source of income, has experienced significant growth. There are over 300 cataloged species in Brazil. Unfortunately, it is common for problems to occur in the hives that can decimate entire colonies. However, most of the research conducted until 2022 has focused on breeding bees with stingers, such as *Apis mellifera*, which have different behaviors and characteristics compared to stingless bees. This work aims to develop a monitoring system for stingless bee hives, specifically targeting the mandaçaia bee (*Melipona quadrifasciata*). The system consists of sensors implemented along with a microcontroller connected to a fog infrastructure, where data adaptation, storage, and analysis were performed. To ensure greater fault tolerance and remote access, this data is also synchronized with a cloud service. Additionally, the system was integrated into a web application, enabling the visualization of collected data, anomaly identification, and triggering mechanisms to assist breeders in making decisions related to intervention in the monitored colony. The developed system can verify the values of the measured variables collected from the stingless bee hives to determine if they fall within the defined normal range. These values are displayed on a specific device that is also part of the system. If any data falls outside the normal range, the system can trigger an alarm on the device, displaying the corresponding value and physical quantity that require verification.

Keywords: Precision Beekeeping, Stingless bees, Internet of Things, Beekeeping, Mandaçaia

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE SIGLAS

ASF	Abelhas Sem Ferrão
IoT	*Internet of Things*
MIT	*Massachusetts Institute of Technology*
RFID	*Radio-Frequency Identification*
ITU	*International Telecommunication Union*
MQTT	*Message Queuing Telemetry Transport*
Wi-Fi	*Wireless Fidelity*
INPA	Instituto Nacional de Pesquisas da Amazônia
CCD	*Colony Collapse Disorder*
TI	Tecnologia da Informação
MCU	*Micro Controller Unit*
IIC	*Inter-Integrated Circuit*
I2C	*Inter-Integrated Circuit*
I²C	*Inter-Integrated Circuit*
WPAN	*Wireless Personal Area Network*
WLAN	*Wireless Local Area Network*
WMAN	*Wireless Metropolitan Área Network*
WWAN	*Wireless Wide Area Network*
RAN	*Radio Access Network*
NFC	*Near field Comunication*
LPWAN	*Low Power, Wide-Area Networks*
M2M	*Machine To Machine*
QoS	*Quality of Service*
DUP	*Duplicate delivery of a PUBLISH packet*
SGBD	Sistema Gerenciador de Banco de Dados
SVM	*Support Vector Machine*
GPRS	*General Packet Radio Services*
PWM	*Pulse Width Modulation*
WDT	*Watch Dog Timer*
NTP	*Network Time Protocol*
RAM	*Random Access Memory*
SD	*Secure Digital*
ID	Identificador
SQL	*Structured Query Language*
CSV	*Comma Separated Values*
IP	*Internet Protocol*
HTTPS	Hypertext Transfer Protocol Secure
HTTP	Hypertext Transfer Protocol
DNS	Domain Name System
TCP	*Transmission Control Protocol*
SSH	*Secure Socket Shell*

SUMÁRIO

1 INTRODUÇÃO

Desde o ensino fundamental, é estudada a importância das abelhas para a vida na Terra. Leite et al. (2016) e Pinto et al. (2018) ressaltam os valores pedagógicos de explanar sobre as abelhas e sua relevância em um ecossistema no ensino fundamental. Os benefícios oferecidos pelas abelhas vão além da produção de mel, pólen, própolis, entre outros. Quando a abelha faz a busca pelo seu alimento, ela poliniza as flores, levando pólen de uma flor até a outra, desempenhando um papel essencial na perpetuação da flora (FRESINGHELI et al., 2019). Segundo Schuwart et al. (2019), cerca de 80% das espécies de plantas dependem da polinização por abelhas. Além disso, é apontado que diversas lavouras aumentam sua produção por área se não forem utilizados defensivos que afetem as abelhas.

Existem mais de 20.000 espécies de abelhas que, em grande maioria são abelhas solitárias, ou seja, não vivem em colônias. Há espécies com ferrão e espécies que possuem seu ferrão atrofiado, sendo estas últimas chamadas de Abelhas Sem Ferrão (ASF) (NOGUEIRA-NETO et al., 1986). As ASF estão presentes no mundo inteiro. No Brasil, há mais de 300 espécies de abelhas da tribo Meliponini (Hymenoptera, Apidae) catalogadas, sendo espécies de ASF. Venturieri (2004b) destaca a importância das abelhas nativas na polinização da fauna brasileira, devido a sua adaptação.

Os criadores das espécies de abelhas Meliponini, são chamados de meliponicultores. Assim como na criação de abelhas africanizadas, as colônias de ASF podem sofrer com determinados tipos de ataques que podem devastar completamente a colônia, levando a morte de todos os indivíduos. São inimigos a se temer: as moscas ligeiras também conhecidas como forídeos (Diptera: Phoridae), pica-paus, formigas, abelhas ladras, entre outros. Tais inimigos naturais costumam

atacar locais de criação de abelhas nativas sem ferrão, enfraquecendo ou até mesmo dizimando as colônias. Desse modo, criadores podem sofrer perdas que impactam financeiramente, considerando que esses ataques nem sempre são perceptíveis em tempo de resguardar as colônias (GOMES et al., 2022; NOGUEIRA-NETO et al., 1986; CONTRERA; VENTURIERI, 2008).

Em criação particular de abelhas sem ferrão, e em contato com outros criadores, houve relatos ou observações de anomalias que dizimaram colônias. Conforme é exposto por Nogueira-Neto et al. (1986) e Venturieri (2004b), certas perdas de colônia podem ser evitadas se o criador tiver a ciência que haja anomalia acontecendo na colônia e tenha tempo de executar uma intervenção.

Utilizando tecnologias disponíveis em 2022, pode-se coletar e enviar dados de sensores pela Internet, permitindo um monitoramento em tempo real para diferentes ambientes em numerosos cenários de aplicações. Esse conceito se encaixa na definição de Internet das Coisas, do inglês *Internet of Things* (IoT).

O termo Internet das Coisas foi inicialmente utilizado pelos anos 2000 no Auto-ID Labs do *Massachusetts Institute of Technology* (MIT), enquanto se trabalhava com *networked Radio-Frequency Identification* (RFID) (WORTMANN; FLÜCHTER, 2015). Outra definição de Internet das Coisas é apontada desde 2012 pelo *The International Telecommunication Union* (ITU) como sendo coisas físicas e virtuais interagindo entre si conectados por meio de tecnologias da informação (KAFLE et al., 2016).

Na literatura, são encontradas outras definições para Internet das Coisas propostas. Algumas dessas definições exibem uma ênfase nas coisas que se conectam. Outras definições focam em aspectos da Internet das Coisas relacionados à Internet, como protocolos de Internet e tecnologia de rede. Um terceiro tipo centra-se nos desafios semânticos da Internet das Coisas relacionados com, por exemplo, o armazenamento, pesquisa e organização de grandes volumes de informação (WORTMANN; FLÜCHTER, 2015). Será tratado nesse trabalho a definição de Internet das Coisas como apresentada pela ITU.

Logo, é possível associar o conceito de IoT para beneficiar meliponicultores com o monitoramento de ASF. Para os criadores, é importante o conhecimento sobre

o processo de criação e manejo, além do conhecimento de aspectos morfológicos e características de ninhos de ASF (NOGUEIRA-NETO et al., 1986; VENTURIERI, 2004b).

Na literatura, são mais comuns os estudos relacionando dados coletados por meio de sensores IoT com as abelhas com ferrão, popularmente chamadas de africanizadas, pertencentes ao gênero *Apis mellifera* (BRAGA et al., 2019; ZGANK, 2020; ZABASTA et al., 2019; SHAGHAGHI et al., 2019), além de poucos estudos em que IoT é relacionado com as ASF (YUSOF et al., 2019; MURPHY et al., 2015b).

Conforme é observado por Braga et al. (2020), existem padrões nos dados de temperaturas coletados em colônias de abelhas africanizadas que podem ser mapeados e auxiliam no tratamento de colmeias, possibilitando até mesmo fazer predições. Braga et al. (2019) definem uma forma de fazer a coleta, identificação e o tratamento dos dados anômalos para reduzir a quantidade a ser armazenada. Dessa forma, minimizando o tráfego desnecessário na rede e beneficiando locais onde as mensagens trocadas são pagas por tráfego. Assim há o benefício de redução de custos quando é utilizado um plano de Internet móvel para a transmissão dos dados. Estas e outras pesquisas encontradas deixam lacunas a serem exploradas. Por exemplo, no trabalho de Mahamud et al. (2019) não são armazenados os dados e no de Fitzgerald et al. (2015) foram elaborados testes somente em laboratório.

Assim, há um vasto número de criadores que não são contemplados com os benefícios da coleta dos dados efetuada pelos estudos existentes. Conforme é apontado por Nogueira-Neto et al. (1986), a forma da coleta e os índices a serem monitorados diferem para as ASF. Além disso, os estudos apresentam coletas em períodos inferiores a uma semana, tendo o foco somente no modo em que é efetuada a coleta, sem disponibilização dos dados (ANUAR et al., 2019; JIANGYI et al., 2019).

1.1 OBJETIVOS

O objetivo deste trabalho é desenvolver um sistema de névoa e nuvem para monitorar colônias de abelhas criadas em caixas racionais, com o intuito de auxiliar os criadores na identificação de anomalias, bem como na tomada de decisão para

intervenção na colônia monitorada. Para isto, o sistema foi desenvolvido e aplicado em caixa de abelha da espécie mandaçaia (*Melipona quadrifasciata quadrifasciata*). O escopo desse sistema abrange a integração de *hardware* e *software*, possibilitando a coleta, armazenamento e visualização de dados em sistema em névoa, integração desses dados com a nuvem, além da verificação dos dados para o acionamento de gatilhos.

Para realizar tais funcionalidades, o sistema foi composto por sensores de temperatura e de umidade relativa, instalados próximos e internos à caixa de abelhas monitorada. Os dados provenientes desses sensores foram enviados para um servidor em névoa para armazenamento e verificação da necessidade de acionamento de gatilho e sincronizados com um servidor em nuvem por meio do protocolo *Message Queuing Telemetry Transport* (MQTT) e conexões *Wireless Fidelity* (Wi-Fi). Também fez parte do sistema o armazenamento dos dados e exibição em uma aplicação Web para visualizar os dados, além de disparar avisos ao criador. Por fim, foi realizada a contribuição de divulgação pública de um *dataset* contendo os dados coletados para permitir futuros estudos.

Além do objetivo geral, pode-se mencionar também os seguintes objetivos específicos:

1. A definição de um *dataset*, que possui informações de temperatura e umidade relativa do ar, coletados dentro da caixa de abelha e na parte externa à caixa;
2. O projeto e implementação de um módulo coletor de fácil acoplamento à caixa de abelha INPA, com microcontroladores e sensores, capazes de enviar as informações observadas por Wi-Fi;
3. A validação da arquitetura de comunicação com o protocolo MQTT em uma aplicação real de IoT, pelo uso das bibliotecas MicroPython *Asynchronous* MQTT e *broker* Mosquitto.

1.2 ORGANIZAÇÃO DO TEXTO

Esta Dissertação de Mestrado é composta de 5 capítulos. No Capítulo 2 foram apresentadas características das ASF, em especial da abelha mandaçaia (*Melipona quadrifasciata*), com exemplos de ameaças que uma colônia de abelhas pode enfrentar, junto da definição de termos, tecnologias e paradigmas que foram aplicados neste trabalho. No Capítulo 3 foi detalhado o desenvolvimento do sistema de *hardware* e *software* para a coleta, armazenamento e exibição de dados de caixa de abelha sem ferrão mandaçaia (*Melipona quadrifasciata*). Já no Capítulo 4 foram demonstradas as execuções e resultados dos experimentos para a validação o sistema. No Capítulo 5 estão as conclusões sobre a execução da presente dissertação.

2 REVISÃO DA LITERATURA

A agropecuária de precisão é um termo abrangente que emprega o uso de tecnologias para monitorar e executar ações, melhorando a gestão das atividades agrícolas. A exemplo do uso de técnica de agropecuária de precisão está a criação de abelhas, meliponicultura ou apicultura de precisão. A apicultura de precisão é dividida em 3 fases de implementação: coleta dos dados, análise e a aplicação (ZACEPINS et al., 2012). Na meliponicultura de precisão estas fases são as mesmas. A precisão no manejo de abelhas é relacionada com o monitoramento de meliponários, apiários e colmeias, visando a redução na mortalidade e o aumento da produção. Em geral, os sistemas de precisão são divididos em 3 etapas: coletas de dados e medições das colônias, análise dos dados coletados e tomadas de ações apoiadas pela análise dos dados (MAHAMUD et al., 2019; MACHHAMER et al., 2020).

Na literatura científica atual, são encontrados trabalhos de apicultura de precisão que relacionam o uso de IoT com a criação de abelhas para atingir seus objetivos. Porém, a grande maioria desses trabalhos possuem como foco as *apis melliferas*, as abelhas africanizadas. O trabalho apresentado foi voltado a uma espécie de abelha sem ferrão, a Mandaçaia. Logo, são apresentados nesse capítulo as características das abelhas Mandaçaias e do seu manejo na meliponicultura, os principais conceitos de IoT e, por fim, os trabalhos relacionados a este.

2.1 ABELHA MANDAÇAIA

A espécie *Melipona quadrifasciata* é encontrada ao longo da costa brasileira desde a Paraíba até o Rio Grande do Sul, com duas subespécies: *Melipona quadrifasciata anthidioides* (MQA) e *Melipona quadrifasciata quadrifasciata* (MQQ). A

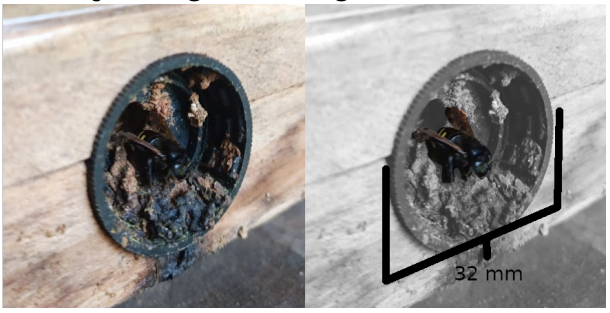
subespécie MQQ observável na Figura 1, sendo mais comum a ocorrência no Sul de São Paulo, Paraná e Santa Catarina, especialmente em regiões mais altas e frias. Em Minas Gerais, a subespécie mais comum é a MQA, mas também foram encontrados ninhos de MQQ em altitudes mais elevadas. Na região norte de Minas Gerais, foi encontrada a subespécie MQQ habitando regiões baixas e quentes de 500 a 700 m (PASSOS, 2010; AIDAR, 2010).

Figura 1: Meliponário Reual, mandaçaia MQQ sobre a cera do ninho.

Fonte: Autoria própria.

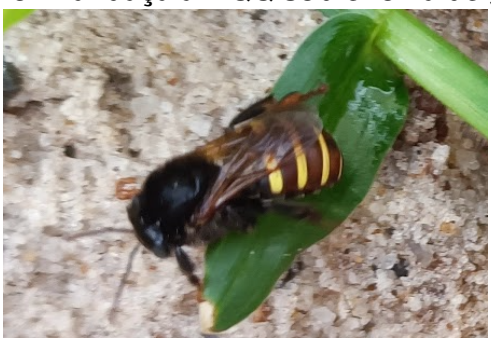
Mandaçaia é um nome de origem indígena que significa vigia bonito. Ela recebe este nome por ser possível observar na colmeia uma abelha que fica guardando a entrada, conforme pode ser observado na Figura 2. Os indivíduos da espécie *Melipona quadrifasciata* medem entre 10 e 11 mm de comprimento, como pode ser observado na Figura 3. Possuem tórax e cabeça pretos, abdômen com quatro faixas amarelas e asas em um tom de marrom (NOGUEIRA-NETO et al., 1986; SANTOS et al., 2021).

Figura 2: Mandaçaia, vigia bonito guardando a entrada da caixa.



Fonte: Autoria própria.

Figura 3: Mandaçaia MQQ sobre folha de grama.



Fonte: Autoria própria.

As abelhas Mandaçaias são criadas por meliponicultores em caixas ninho, para facilitar o manejo, na Figura 4 pode ser observada uma caixa ninho habitada por um enxame da espécie mandaçaia MQQ. A criação da Mandaçaia é voltada para a produção do mel, utilizado como alimento ou como remédio (MOURA et al., 2021). Além disso, pode ser feita a coleta do geoprópolis produzido pela espécie, que também é estudado para uso farmacológico (FERREIRA, 2021; MOURA et al., 2021). Outros benefícios da criação de abelhas estão relacionados com seus efeitos em plantações, uma vez que as colmeias podem ser dispostas próximas a elas, sendo utilizadas para a polinização e beneficiando certas culturas, como o tomate (SARTO, 2005).

Figura 4: Caixa ninho com abelhas da espécie mandaçaia MQQ.

Fonte: Autoria própria.

Há diversas espécies de abelhas, e estas espécies sofrem com inimigos naturais, sendo um aspecto também preocupante para as abelhas Mandaçaias. As ameaças podem prejudicar a produção ou até mesmo dizimar a colônia (CAESAR,

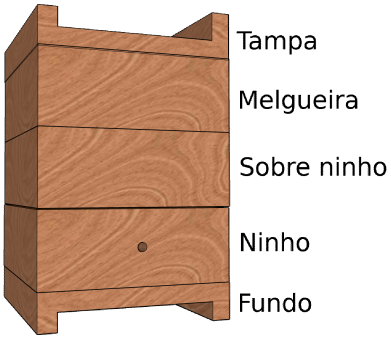
2020). Essas ameaças serão mais bem discutidas adiante na Seção 2.3.

2.2 CAIXA RACIONAL MODELO INSTITUTO NACIONAL DE PESQUISAS DA AMAZÔNIA

Uma caixa racional é uma caixa desenvolvida com razão ou lógica para a criação de abelhas. As caixas racionais são utilizadas de modo que facilitem o manejo para a multiplicação de enxames e a colheita de mel (VENTURIERI, 2004a). O ninho é colocado na caixa, por esse motivo, na literatura são encontradas menções a caixa ninho. O modelo de caixa racional Instituto Nacional de Pesquisas da Amazônia (INPA) foi proposto pelo pesquisador Fernando Oliveira quando trabalhou no INPA (OLIVEIRA, 2000). É um projeto modular, que visa a facilidade para a divisão de colônias e a coleta de mel.

As caixas INPA possuem duas partes apenas estruturais, o fundo e a tampa, e 3 módulos básicos funcionais, que podem ser vistos na Figura 5: o ninho, sobre ninho e melgueira. A postura dos ovos pela rainha é feita em células de cria. Estas células são construídas pelas operárias e dispostas uma ao lado da outra, formando disco. Os discos são construídos uns sobre os outros, e, se não houver limitador, a rainha continua a postura para o módulo superior. Esse ato de a postura ser levada ao módulo superior é chamado de subir a postura.

Figura 5: Modelo da caixa racional INPA com seus principais módulos destacados.



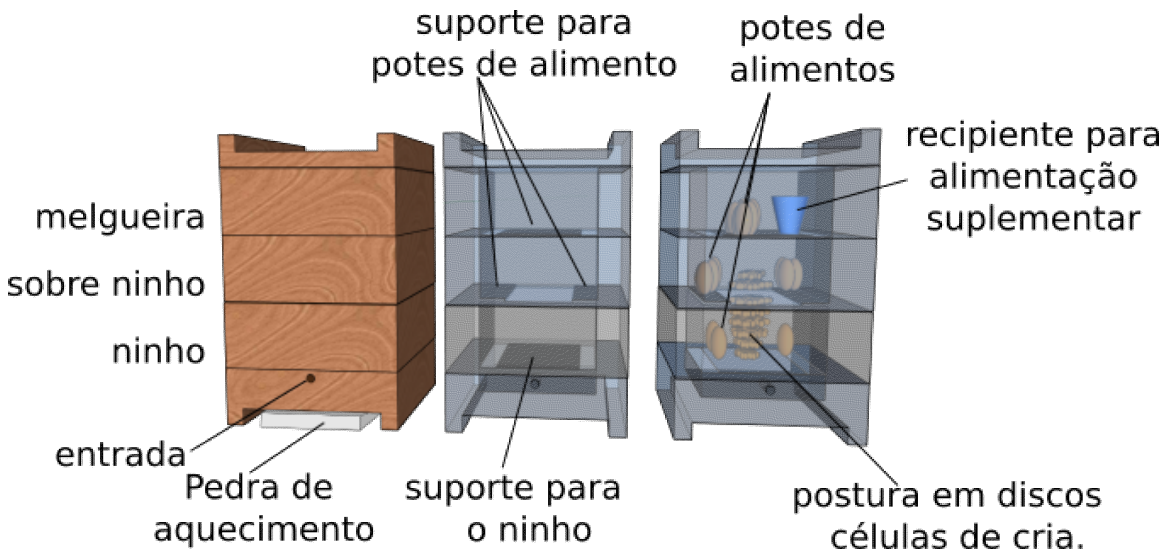
Fonte: Adaptado de (OLIVEIRA, 2000).

As Mandaçaias coletam as resinas de plantas lenhosas e barro, e os misturam gerando o geoprópolis. Essa mistura é utilizada para vedar orifícios, ou até mesmo

separar espaços em um ninho. Certas vezes, ocorre de o criador inserir um novo módulo em uma caixa e as abelhas isolarem esse novo módulo. O termo “propolizar” é utilizado pelos criadores quando as abelhas utilizam o própolis ou o geoprópolis para vedar algo. Nas Mandaçaias, é comum observar um módulo bem colado ao outro, ou um módulo isolado por ter suas entradas propolizadas (NOGUEIRA-NETO et al., 1986).

Como as abelhas Mandaçaia constroem o ninho ao centro do módulo, o módulo do sobre ninho possui uma abertura central, que, em uma colônia forte, permite que a postura ocupe os dois módulos. Se houver postura em ambos os módulos, uma divisão da colônia pode ser feita. Conforme pode ser observado na Figura 6, o módulo de melgueira possui um fundo que isola o meio do módulo, impedindo que seja dada continuidade na postura até ele. Para as abelhas transitarem entre os módulos sobre ninho e melgueira, existem frestas nas laterais do fundo do módulo melgueira. Assim, no módulo melgueira, as abelhas depositam apenas alimento, deixando as células de cria nos outros módulos (NOGUEIRA-NETO et al., 1986; SANTOS et al., 2021; VILLAS-BÔAS, 2012).

Figura 6: Visão interna com a disposição dos elementos da caixa racional INPA.



Fonte: Adaptado de (OLIVEIRA, 2000).

2.3 AMEAÇAS

Há relatos, nas últimas décadas, da redução na população de abelhas. Nos anos de 2005 e 2006 apicultores europeus e norte americanos passaram a chamar de desordem do colapso de colônias, do inglês *Colony Collapse Disorder* (CCD), que compreendem fatores que levam uma colônia a colapsar pela perda repentina de população (ZACEPINS et al., 2012; STOKSTAD, 2007). Esses fatores constituem em diversas ameaças documentadas também na literatura. Mesmo em uma caixa racional, as colônias sofrem ameaças. (BELLOS et al., 2021) categorizam estas ameaças para as criações em 3 grupos:

1. roubo e intervenção humana;
2. ataques de mamíferos;
3. ataques de vespas, ácaros ou incidentes de adaptação de colmeias, como morte de rainha, condições extremas de calor ou frio, fome ou até mesmo sede.

O grupo 1 cita situações de intervenção humana. Podem ser atos de vandalismo, ou roubo das colônias. Os ladrões são atraídos pelo valor do mel, que podem ser deixados nas melgueiras aguardando a colheita. Também pode acontecer o roubo da colônia inteira, pois colmeias povoadas tem o seu valor de mercado (NOGUEIRA-NETO et al., 1986; WOLFF et al., 2006; JIANGYI et al., 2019).

No grupo 2 estão as ameaças causadas por animais mamíferos representados por bois, cavalos, porcos, cabras, entre outros. Esses animais podem entrar em contato direto com as caixas ao se esfregar, derrubando-as e danificando as colmeias. Também fazem parte desse grupo casos de animais como iraras que se alimentam do mel, entram em uma caixa e depredam o ninho (NOGUEIRA-NETO et al., 1986; WOLFF et al., 2006; JIANGYI et al., 2019).

No grupo 3 estão os incidentes de adaptação biológica das abelhas. Ataques de vespas, formigas, abelhas, pilhadoras, ácaros, certos tipos de moscas e besouros. Também estão dentro desse grupo os incidentes causados por condições climáticas tais como calor ou frio extremos, seca ou umidade perduradora e causados pela falta

de alimento, resinas, pólen, néctar ou água (NOGUEIRA-NETO et al., 1986; JIANGYI et al., 2019).

Inimigos naturais das abelhas sem ferrão, que podem dizimar uma colônia, são formigas, abelhas ladras, determinadas espécies de moscas, ácaros e besouros (NOGUEIRA-NETO et al., 1986). As formigas atacam uma colônia em busca de alimento, ou para fazer o ninho no local. Certas espécies de formigas e cupins vivem em conjunto com as abelhas, mas outras podem atacar, comendo as crias e matando as abelhas (NOGUEIRA-NETO et al., 1986; CAESAR, 2020; JIANGYI et al., 2019).

A espécie de abelha *Lestrimelitta limao*, popularmente conhecida como irati, iratin ou abelha limão, não produzem mel ou cera, vivem de pilhagem de outras espécies, comportamento chamado de cleptoparasitismo (ZUBEN, 2012). Um ataque de abelha limão começa com uma ou duas campeiras de *Lestrimelitta* tentando encontrar a entrada do ninho. Quando a entrada é encontrada, a abelha faz uma marca cujo odor irá atrair outras de sua espécie. Após isso se dá a chegada de um número grande de indivíduos de *Lestrimelitta* que exalam um odor de limão forte. No mesmo momento as operárias dominam a entrada da colônia atacando as forrageadoras que tentam entrar ou sair (ZUBEN, 2012; NOGUEIRA-NETO et al., 1986). Então as abelhas limão invadem o ninho matando as operárias originais que tentam defender a colônia do ataque. No saque, buscam mel, pólen, materiais do ninho, e o alimento larval. Após o ataque que dura de um a trinta e quatro dias, as *Lestrimelitta* começam a partir. As abelhas limão, assim como as formigas, exalam ésteres que podem ser identificados com sensor específico (ZUBEN, 2012).

Além dos três grupos mencionados anteriormente, pesquisadores também apontam para a preocupante diminuição na população de abelhas como resultado do uso de agrotóxicos. As abelhas operárias, ao partirem para os campos em busca de néctar e pólen, enfrentam uma ameaça crescente. Quando essas incursões ocorrem em áreas tratadas com agrotóxicos, as abelhas correm um grande risco de transportar substâncias tóxicas de volta à colmeia. Esse fenômeno resulta em sérios problemas classificados em subletais ou letais, como a diminuição no número e no peso das larvas, podendo até mesmo levar à morte das abelhas dentro da colônia. Na área urbana é comum o uso de produtos conhecidos como “mata-mato”, herbicidas

aplicados sobre plantas com flores visitadas pelas abelhas. Também o “fumacê”, veículo utilizado para aspersar um inseticida não seletivo para controle do mosquito da dengue levam colônias ao declínio e morte (GRANDO, 2022; REGO et al., 2022; GEMIM et al., 2022; LIMA; ROCHA, 2012).

Outra ameaça às abelhas é a falta de alimentos na natureza. Logo, uma maneira de auxiliar as colônias nessa condição é suplementando sua alimentação. Uma das possibilidades para a suplementação é por meio de um xarope produzido ao se cozinhar açúcar em água. Esse xarope é servido em recipientes que as abelhas tenham acesso. Elas recolhem o xarope e fazem o processamento para o consumo (NOGUEIRA-NETO et al., 1986). Na Figura 6 é mostrado um pote para alimentação suplementar, que pode ser um copo de iogurte, com pedras ou palitos dentro, dispostos de forma que abelhas possuam um apoio e não se afoguem.

Já existem sistemas de segurança contra roubos, baseados em células de carga, sistemas de câmera e rastreamento via GPS que transmitem via GPRS ou IEEE 802.15.4 (MAHAMUD et al., 2019; ANUAR et al., 2019; YUSOF et al., 2019). Contudo, são ineficazes no que se diz respeito ao consumo energético e custos de manutenção. Além do sistema de alarme de peso não apresentarem um bom potencial, pois vários fatores como chuva, vento, posicionamento das células de carga e desequilíbrio da carga podem afetar o peso da colmeia, ativando o alarme, mas não tendo relação com roubos (ANUAR et al., 2019). Kits de câmeras de segurança para colmeia também são utilizados na criação de abelhas para rastrear movimentos e captar fotos ou vídeos e enviar para o celular do criador, mas apresenta um consumo energético significativo (BELLOS et al., 2021; MURPHY et al., 2015a; KULYUKIN, 2021).

As abelhas são insetos sensíveis às variações do clima (temperatura, umidade, luminosidade e precipitação). No inverno, elas tendem a se aglomerar nas áreas de cria, vibrando os músculos torácicos na tentativa de gerar calor. Por isso, ao se criar abelhas artificialmente, é necessário fornecer um ambiente adequado para ocorrer produção satisfatória dos subprodutos, assim como sua reprodução (JESUS et al., 2017).

As abelhas mantêm a temperatura no interior da caixa entre 15°C e 35°C. Se a temperatura está acima de 35 ℃, as abelhas se posicionam de forma que batendo

as asas façam o ar circular, resfriando o ninho. Quando a temperatura está abaixo dos 15 ℃, as abelhas vibram para produzir calor, ou entram em diapausa. A diapausa é definida como uma baixa atividade metabólica em resposta a sinais ambientais que precedem condições ambientais desfavoráveis ou baixa disponibilidade de recursos biológicos (JESUS et al., 2017; LOLI, 2008; AIDAR, 2010). Esta variação pode ser observada coletando os dados de um sensor de temperaturas no interior colônia.

Um modo de economizar os recursos das abelhas é mantendo a temperatura interna da caixa confortável para elas. Criadores costumam aquecê-la colocando um termostato que verifica a temperatura de uma caixa vazia e ativa resistências colocadas embaixo das caixas, como representado na Figura 6 (JESUS et al., 2017). Estas resistências podem ser feitas como uma pedra de aquecimento, que apresenta uma melhor dissipação do calor ajustada para a caixa. A pedra é construída com o auxílio de uma forma, em que são colocadas as resistências com seus terminais ligados cada um em um fio e estes fios acomodados fora da forma. Posteriormente, é despejado cimento sobre a resistência. Visando o controle da temperatura, a pedra é colocada sob a caixa, conforme visto na Figura 6 e acionada por um termostato. Utilizando o sensor de temperatura em conjunto com um relé para acionar uma pedra de aquecimento, a zona de conforto térmica das abelhas pode ser atingida mais facilmente.

2.4 INTERNET DAS COISAS

O termo Internet das Coisas, do inglês *Internet of Things*, é um paradigma que está tomando força desde o início dos anos 2000. Esse conceito designa processos que envolvam objetos conectados à Internet, responsáveis por produzir informações ou executar ações em tempo real, sem a necessidade de intervenção humana (MADAKAM et al., 2015; KRčO et al., 2014).

São exemplos de sistemas IoT: uma lâmpada ligada automaticamente após a consulta na Internet do horário do pôr do sol na região onde ela está instalada e desligada ao amanhecer; um par de tênis com sensores de pisada enviando o número de passos para um aplicativo que gera um plano de treino personalizado ao usuário;

e um vaso sanitário que mede a intensidade do jato de urina de um homem e analisa se existe possível alteração na próstata (FUJITA et al., 2019; VERMESAN; FRIESS, 2013).

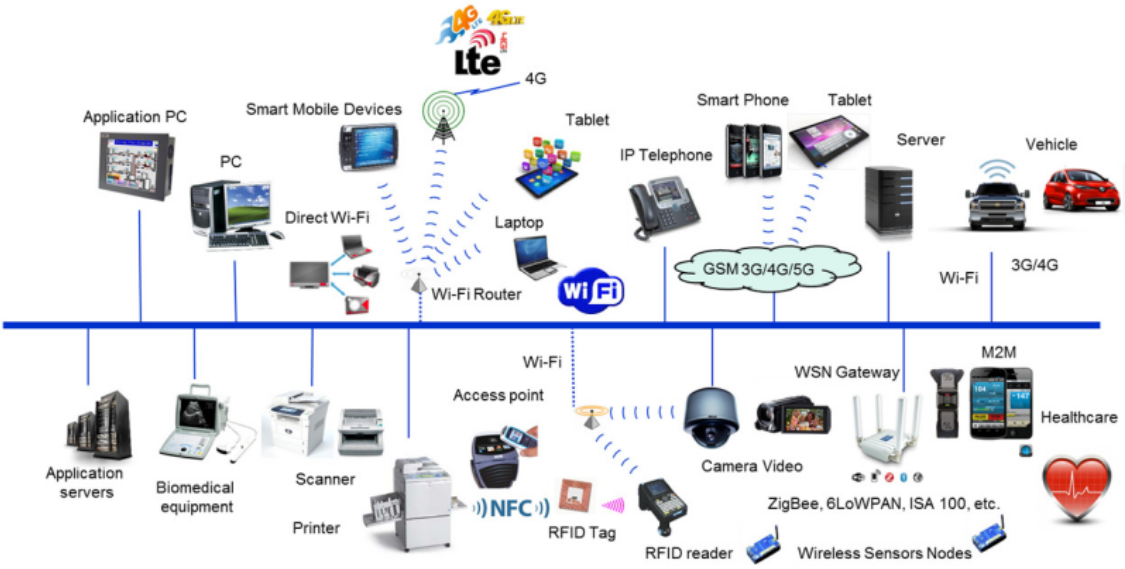
O paradigma da Internet das Coisas está em constante progresso e pode conectar outros objetos também, ampliando suas possíveis aplicações, como eletrodomésticos, aparelhos que monitoram sinais vitais, entre outros. Estes dispositivos podem ser sensores em uma residência, fornos, geladeiras, máquinas em uma fábrica, automóveis e outros locais (GALEGALE et al., 2016). Uma das formas dos objetos interagirem pode ser utilizando uma unidade de processamento onde estarão conectados sensores e/ou atuadores.

No início dos anos 2000, a computação em nuvem, do inglês *cloud computing*, era tida como tendência e também passou a ser utilizada em conjunto com os sistemas IoT. Na computação em nuvem, os recursos de Tecnologia da Informação (TI) são executados na Internet. Como exemplo são expostos: armazenamento de arquivos, coleta, tratamento e armazenamento dados, servidores web, *firewalls*, entre outros (ATLAM et al., 2018; YOUSEFPOUR et al., 2019).

Autores como Vermesan e Friess (2013) apresentam que, além dos dispositivos de IoT, outros aparelhos como câmeras de vigilância ou aparelhos de telefone, estão também convergindo para redes IP, como representado na Figura 7. O que mudaria o paradigma de Internet das Coisas para Internet de Tudo, com cada vez mais dispositivos conectados, ampliando o alcance e interação desses sistemas.

Outro paradigma, que é ainda mais recente, criado em 2012, que também está associado aos sistemas IoT, é o de computação em névoa, do inglês *fog computing*. Esse conceito surgiu para tratar o desafio de lidar com o volume de dados gerados num cenário de crescimento de dispositivos conectados à Internet. Em diferentes projetos de IoT, por limitações de largura de banda entre os dispositivos de IoT e a nuvem, é necessário que o mínimo de dados sejam trafegados pela Internet. Desta forma, na estrutura do projeto do sistema, antes da saída para a Internet, na borda, é feito um pré-processamento visando a redução do grande volume de dados recebido dos sensores, minimizando a demanda de largura de banda entre o local e a nuvem, conforme pode ser observado na Figura 8. Além disso, pode gerar respostas em um

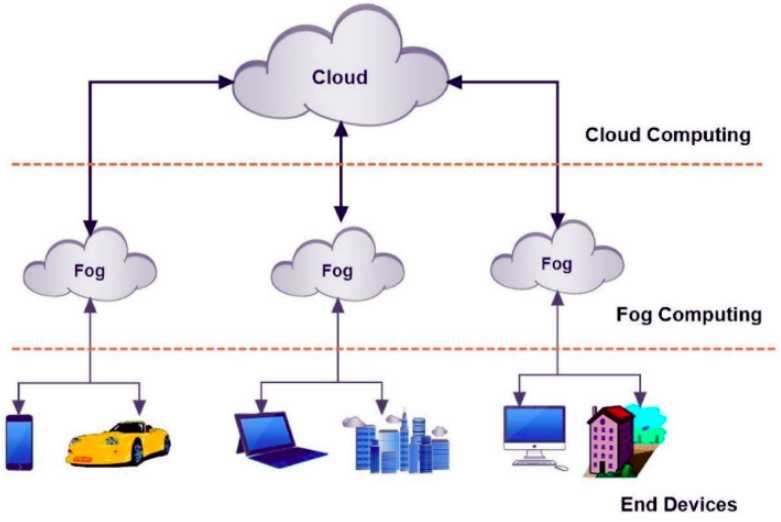
Figura 7: Possíveis aplicações de IoT.



Fonte: Vermesan e Friess (2013).

período menor do que o possível pela nuvem. O tratamento dos dados na névoa reduz o tráfego da conexão com a nuvem, economizando recursos de infraestrutura de comunicação. Também viabiliza aplicações que necessitem de respostas com baixa latência, mesmo quando a conexão à nuvem estiver indisponível (ATLAM et al., 2018; YOUSEFPOUR et al., 2019; BONOMI et al., 2012). (ATLAM et al., 2018; YOUSEFPOUR et al., 2019; BONOMI et al., 2012).

Figura 8: Representação da interação dos elementos na computação em névoa, sendo uma extensão da computação em nuvem, mais próxima aos dispositivos finais.



Fonte: (ATLAM et al., 2018).

Na computação em névoa, são propostas soluções para diferentes desafios da computação em nuvem aliada à IoT. É possível obter a redução da latência, da largura de banda necessária e do volume de dados a ser trafegado, e ainda podem ser incluídos serviços que não podem parar com o uso da redundância, resultando em uma melhor tolerância a falhas e melhoria da segurança dos dispositivos finais com o uso de protocolos de segurança que demandam maiores recursos ao serem gerenciados pelos equipamentos próximos aos objetos IoT (COUTINHO et al., 2016; ATLAM et al., 2018). Características e diferenças entre os paradigmas de computação em névoa e computação em nuvem são apresentados na Tabela 1. Aqui, podem ser observadas características como a disponibilidade de acesso na computação em névoa tem a particularidade do acesso local ou limitado, enquanto computação na nuvem pode ser acessado globalmente. Outras propriedades que diferem os paradigmas são de que a capacidade de processamento e armazenamento que em névoa são limitados, enquanto em nuvem são virtualmente ilimitadas, podem ser contratadas conforme a necessidade de uso (COUTINHO et al., 2016).

Tabela 1: Comparação de aspectos complementares da Computação em Nuvem e IoT.

Característica	Névoa	Nuvem
Modelo de computação	Distribuído ou pervasivo	Centralizado
Disponibilidade de acesso	Local ou limitado	Global ou ubíquo
Natureza dos componentes	Objetos físicos	Recursos virtuais
Capacidade de processamento	Limitada	Virtualmente ilimitada
Capacidade de armazenamento	Limitada ou nenhuma	Virtualmente ilimitada
Função da Internet	Ponto de convergência	Meio de prover serviços
Análise de dados	Análise em tempo real	Análise de Big Data

Fonte: (COUTINHO et al., 2016).

Existem diferentes modelos de arquiteturas para sistemas IoT definidos segundo o modelo em camadas, já utilizado previamente para a infraestrutura de redes de computadores. Segundo Al-Fuqaha et al. (2015), as arquiteturas em camadas para IoT deve possuir versatilidade para a adequação em um projeto. Porém, não existe uma padronização para a arquitetura de sistemas IoT. Dependendo das necessidades do escopo do projeto, ele é enquadrado em um modelo de arquitetura de camadas

diferente, já que existe mais de um modelo.

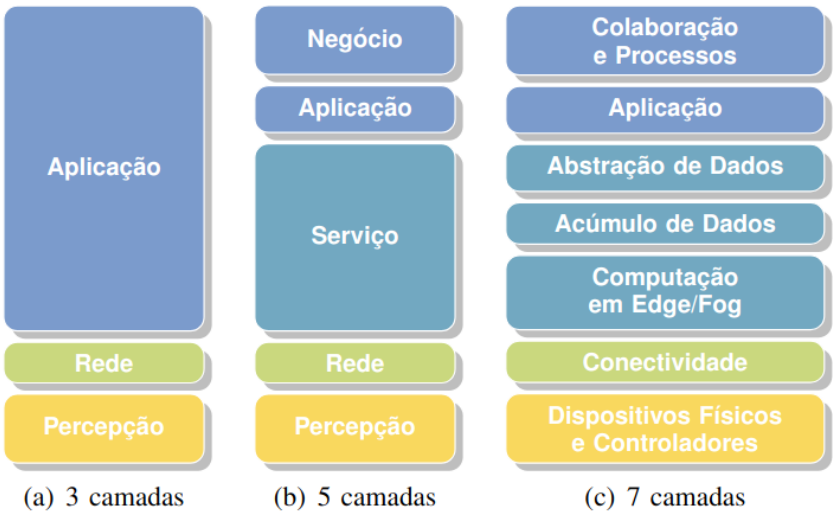
A arquitetura de 3 camadas é apresentada na Figura 9(a). Considerando uma visão *bottom-up*, as camadas são descritas da seguinte forma (KRčO et al., 2014; SANTANA et al., 2019; PASTÓRIO et al., 2020):

1. Camada de percepção, de sensoriamento ou de objetos: é a camada onde estão localizados os sensores, atuadores ou etiquetas RFID, ou seja, os objetos, ou as coisas da Internet das Coisas.

2. Camada de conexão ou de transporte: é a camada que faz a conexão entre os sensores e a camada de serviços. Para isso utiliza tecnologias de comunicação, como Bluetooth, Ethernet, Zigbee ou Wi-Fi.

3. Camada de aplicação: é a camada responsável por entregar a visualização dos dados ao usuário. Por exemplo, um aplicativo que mostra as informações de batimentos cardíacos coletados por um relógio inteligente.

Figura 9: Arquitetura de camadas para IoT: (a) arquitetura de 3 camadas, (b) arquitetura de 5 camadas e (c) arquitetura de 7 camadas.



Fonte: (PASTÓRIO et al., 2020).

Na arquitetura de 5 camadas, que pode ser vista na Figura 9(b), há as camadas 1 e 2, percepção e rede, com a definição igual ao modelo de 5 camadas. Já a camada 3 é subdividida em 3 novas camadas. Na camada 3, chamada de

camada de serviço, de processamento ou de *middleware*, é onde estão localizados os serviços utilizados, como exemplos: um servidor de banco de dados ou o *broker* MQTT. É a camada responsável também pelo armazenamento dos dados coletados. Já a camada 4, nomeada como camada de aplicação, é responsável por entregar ao usuário a visualização dos dados. Como, por exemplo, um aplicativo que mostra as informações de batimentos cardíacos coletados por um relógio inteligente. Por fim, existe a camada 5, conhecida como camada de negócio ou de gerenciamento, por gerenciar as camadas anteriores, além de ser a camada responsável por fornecer o modelo de negócio. Na camada de negócio é feito o processamento de dados das camadas anteriores, possibilitando o fornecimento de visualizações como gráficos e fluxogramas (KRčO et al., 2014; SANTANA et al., 2019; PASTÓRIO et al., 2020).

A arquitetura de 7 camadas é proposta pela Cisco, IBM e Intel (GREEN, 2014). Como pode ser observado na Figura 9(c), a camada nomeada como percepção nos modelos de 3 e 5 camadas, possui o nome de dispositivos físicos e controladores no modelo de 7 camadas. A camada de rede dos modelos de 3 e 5 camadas é chamada de conectividade no modelo de 7 camadas. Já a camada de serviço do modelo de 5 camadas, é dividida em 3 camadas. A primeira é a computação em *edge/fog*, que processa, filtra e faz a análise inicial dos dados coletados nas camadas inferiores. A segunda é a camada de acúmulo de dados, onde estão os serviços de armazenamento. Por fim, a camada de abstração de dados processa os dados recebidos das camadas inferiores a fim de reduzir os mesmos para passar para a camada de aplicação somente o necessário. Além dessas, existem mais 2 camadas com funções similares à de aplicação do modelo de 3 camadas. A camada de aplicação tem a mesma função da respectiva camada do modelo de 5 camadas. Na última camada, de colaboração e processos, estão os processos envolvidos na execução de uma tarefa.

Para a ligação entre as camadas 1 e 3 das arquiteturas de camadas apresentadas anteriormente, existe a camada 2. Nos modelos de 3 e 5 camadas, a camada 2 possui a denominação de rede, e no modelo de 7 camadas é a camada de conectividade. Nos modelos de arquitetura de camadas para IoT apresentados anteriormente, a segunda camada necessita de tecnologias que façam a conexão de

múltiplos sensores utilizando, preferencialmente, pouca energia com um alcance longo (CHETTRI; BERA, 2020; SHENG et al., 2015).

A comunicação entre sensores e atuadores com a infraestrutura da computação em nuvem é possível ser realizada utilizando microcontroladores e microprocessadores (PENIDO; TRINDADE, 2013; SANTANA et al., 2019).

Os sensores são equipamentos sensíveis a determinado tipo de energia. Exemplos dessas energias são magnéticas, térmicas, elétricas, luminosas e cinéticas. O sensor consegue informar, convertendo a grandeza a ser mensurada em sinais digitais, como pressão atmosférica, umidade do solo, intensidade de luz, entre outros. Os atuadores são dispositivos que quando acionados, executam funções, como exemplos: acender uma lâmpada, abrir uma válvula e ligar um motor elétrico (AL-FUQAHA et al., 2015; SANTANA et al., 2019).

Um microcontrolador ou *Micro Controller Unit* (MCU) é uma unidade eletrônica programável com portas digitais configuráveis como entradas ou saídas. Podem conter portas de comunicação, portas analógicas e outras funções (MUCKENFUHS, 2020; MALINOWSKI; YU, 2011). Exemplos de microcontroladores são o Arduino, ESP 8266, ESP 32, MSP430F235, Tiva C TM4C123G, microcontroladores PIC, entre outros (TARGA et al., 2019; THOMAZINI; ALBUQUERQUE, 2020; SANTANA et al., 2019).

Microprocessadores ou unidades de processamento, são *hardwares* com capacidade de processamento, possuem capacidade computacional, quantidade de memória superior que os microcontroladores. Também possuem portas de expansão que possibilitam a ligação de periféricos tais como teclado, mouse, unidades de armazenamento, monitores de vídeo entre outros. Podem ser chamados de mini computadores, e como exemplo podem ser citados Banana PI, Raspberry PI, CubieBoard e computadores pessoais (CRISP, 2004; SANTOS et al., 2016; MALINOWSKI; YU, 2011).

O número de portas digitais pode ser um limitador para a instalação de sensores em um microcontrolador ou microprocessador. Existem barramentos e protocolos que podem compartilhar uma porta digital com mais de um dispositivo.

O *Inter-Integrated Circuit* (IIC, I2Cou I²C) é um barramento de comunicação desenvolvido pela Philips, utilizado para conectar múltiplos dispositivos com diferentes endereços de *hardware* em uma única linha de dados. Além disso, cada dispositivo também deve ter duas conexões de energia. O barramento I²C permite a comunicação entre dispositivos com diferentes endereços e é compatível com uma variedade de dispositivos, incluindo sensores de temperatura, pressão, *displays* de cristal líquido e sensores de corrente elétrica. O barramento I²C está presente em microcontroladores Arduino Uno, ESP 8266 ou ESP32. O I²C está na versão 6 publicada a especificação no ano de 2014 (NXP SEMICONDUCTORS, 2021; NASCIMENTO et al., 2021).

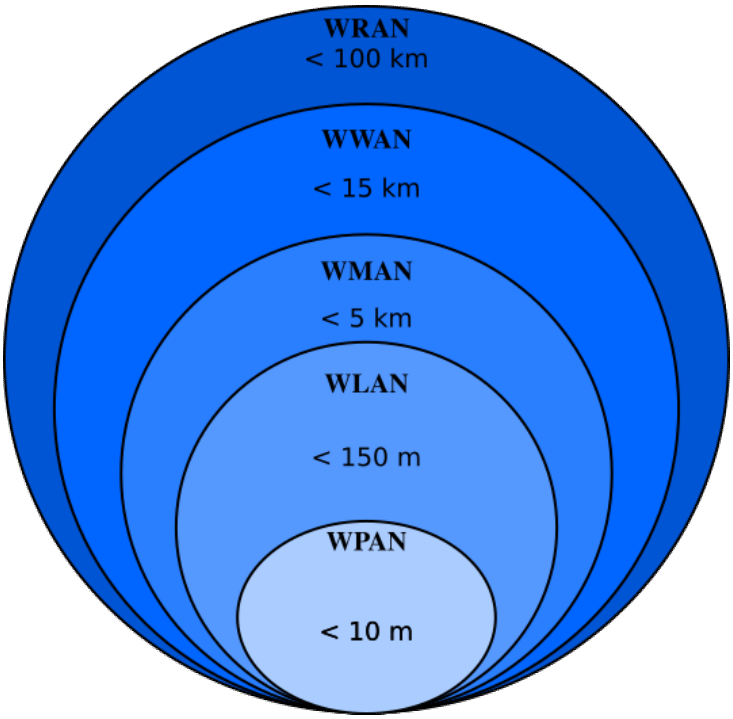
Outro exemplo de sistema de barramento para compartilhamento de porta digital é o 1-Wire. O protocolo 1-Wire foi criado pela *Dallas Semiconductor Corp.*, empresa que desde 2007 chama-se *Maxim Integrated Products*. O protocolo 1-Wire é diferente do I²C, pode ser configurado em uma porta digital disponível em um microcontrolador como o Arduino Uno ou o ESP 8266. O protocolo 1-Wire permite a conexão de até 256 dispositivos por barramento desde que seus endereços sejam diferentes (COLORADO; MARTÍINEZ-SANTOS, 2017; MAXIM INTEGRATED PRODUCTS, 2016).

A classificação por área de alcance de redes sem fio é apresentada na Figura 10. São elas: *Wireless Personal Area Network* (WPAN), rede sem fio de área pessoal, com alcances de até 10 m; *Wireless Local Area Network* (WLAN), rede sem fio de área local, com alcances de até 150 m; *Wireless Metropolitan Área Network* (WMAN), rede sem fio de área metropolitana, com alcances de até 5 km; *Wireless Wide Area Network* (WWAN), rede sem fio de área ampla, com enlaces de até 15 km; e *Radio Access Network* (RAN), rede de acesso de rádios, permite enlaces de até 100 km (CORDEIRO et al., 2005; NAITO, 2017).

Considerando diferentes áreas de alcance, também existem diferentes tecnologias de comunicação sem fio para IoT, como as expostas a seguir, cujas características também são apresentadas na Tabela 2 (NAITO, 2017; FAN et al., 2018):

- Bluetooth: funciona em distâncias de até 10 metros, baseado no padrão IEEE

Figura 10: Área de redes sem fio para IoT.



Fonte: Adaptado de (CORDEIRO et al., 2005).

802.15.4.

- *Near field Comunication* (NFC): em tradução livre comunicação de campo próximo, é uma tecnologia que evoluiu do RFID. O NFC comunica em distâncias abaixo de 4 cm. Aplicado em situações como autenticação de identidade, rastreamento logístico e pagamentos por aproximação;

- LoRa: é um padrão baseado em tecnologias sem fio proposto pela LoRa *Alliance*. Foi desenvolvido para otimizar as comunicações *Low Power, Wide-Area Networks* (LPWAN), com o intuito de cobrir grandes áreas consumindo pouca energia;

- ZigBee: também baseado no padrão IEEE 802.15.4, criado para utilizar pouca energia. Estima-se que um par de pilhas alcalinas AA durem entre 6 meses e dois anos, além de possibilitar enlaces de até 100 m;

- Wi-Fi: tecnologia criada em 1997 baseada no padrão IEEE 802.11. Fornece altas taxas de transmissão se comparadas com outras tecnologias. Tem um

consumo médio de energia e possibilita enlaces de aproximadamente 100 m com baixa latência e uma grande largura de banda.

Também é possível observar diferentes características de tecnologias de comunicação sem fio para sistemas IoT na Tabela 2 (SIKIMIC et al., 2020). Logo, a escolha da tecnologia mais adequada depende dos requisitos da aplicação entre as possibilidades disponíveis.

Tabela 2: Tecnologias de comunicação sem fio para IoT.

Tecnologia	Alcance	Latência	Consumo de energia	Cobertura	Custo (USD)
Bluetooth Low Energy	< 10 m	< 10 ms	Baixo	Pequena	$\$2 \sim \$5$
NFC	< 4 cm	$100 \sim 250$ ms	Muito Baixo	Muito Pequena	Leitor $\$5 \sim \$40$ Tag $< \$1$
ZigBee	< 100 m	< 20 ms	Baixo	Pequena	$\$1 \sim \$5$
Wi-Fi	< 100 m	< 10 ms	Médio	Pequena	$\$1 \sim \$10$
LoRaWAN	5 km Urbano 20 km Rural	$1 \sim 10$ s	Baixo	Ampla	$\$2 \sim \$25$
3G/4G/LTE	> 30 km	100 ms ~ 1 s	Baixo a Alto	Ampla	$\$4 \sim \$35$

Fonte: Adaptado de (SIKIMIC et al., 2020).

2.5 CONTÊINERES E DOCKER

Na área de informática, há uma técnica de virtualização conhecida como contêiner, que consiste em uma unidade de software capaz de criar um ambiente independente e isolado para executar uma aplicação com seus próprios recursos de sistema, bibliotecas e dependências. Uma das tecnologias mais populares para virtualização de contêineres é o Docker, que automatiza o processo de criação, gerenciamento e execução desses ambientes de forma eficiente e escalável. Com o Docker, é possível criar imagens de contêineres portáteis e leves, que podem ser executadas em qualquer sistema operacional compatível com o Docker. Além disso, o Docker oferece ferramentas de gerenciamento de recursos, rede e armazenamento.

Por isso, o Docker é amplamente utilizado na indústria de desenvolvimento de software para fornecer ambientes de desenvolvimento consistentes e portáteis (BATISTA, 2022).

2.6 MQTT

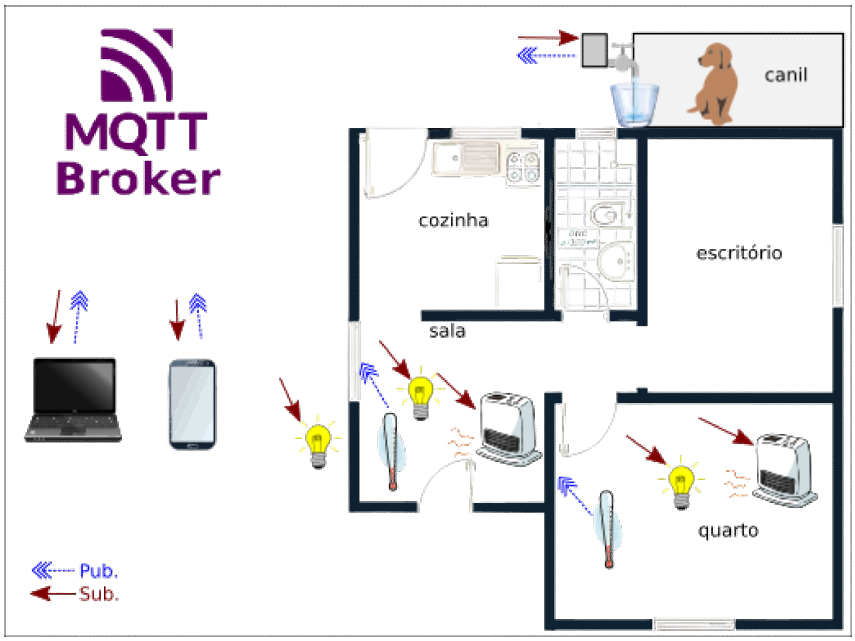
O protocolo da camada de aplicação *Message Queuing Telemetry Transport* (MQTT) foi criado pela IBM e a Eurotech no ano de 1999, com a premissa de ser simples, leve e de fácil implementação, para prover comunicação máquina para máquina, do inglês *Machine To Machine* (M2M) utilizando uma conexão TCP. Foi desenvolvido para coletar informações de monitoramento de oleodutos transmitindo os dados via satélite. Em 2010 foi liberado gratuitamente, sendo a versão informada como 3.1, ficando o padrão sob responsabilidade do OASIS *Open offers projects*. Em 2014 foi publicada a primeira versão sob responsabilidade da OASIS, a versão 3.1.1, e a sua versão 5 foi publicada no ano de 2019 (GEMIRTER et al., 2021; OASIS MQTT TECHNICAL COMMITTEE, 2019).

No cenário de comunicação do MQTT, um dos elementos mais importante é o serviço do MQTT, chamado de agente MQTT ou de *broker* MQTT, que tem a função de gerenciar os tópicos recebendo e disponibilizando as mensagens ou *payloads*. Os clientes têm as funções de *publisher* (publicador) e/ou *subscriber* (subscrito ou assinante). O tópico, ou também chamado assunto, pode ser comparado a uma variável. Como exemplo de tópico pode-se ter “casa/temperatura”. Nesse exemplo, os dados coletados periodicamente de um sensor de temperatura são publicados no *broker* MQTT. Outros clientes podem assinar o tópico e, quando houver publicação no tópico, o *broker* se encarrega de replicar a postagem para todos os clientes, que estão subscritos neste tópico (HARIPRIYA; KULOTHUNGAN, 2019; DINCULEANĂ; CHENG, 2019).

A Figura 11 possui a representação de um cenário de comunicação MQTT em uma casa com dispositivos do tipo *publisher* e *subscriber*. Conforme ilustrado, as setas em que estão voltadas para o *broker* MQTT levam a informação do dispositivo para ser escrita no tópico. Logo, é possível observar que o sensor de volume de água

do bebedouro no canil e os termômetros enviam seus dados para o *broker* MQTT. Se houver alteração nos tópicos, os dispositivos que os assinam serão notificados pelo *broker* MQTT. Estes dispositivos subscritos nos tópicos estão representados pela seta que inicia no *broker* MQTT e vai em direção ao dispositivo. Nesse exemplo, por meio do intermédio do *broker* MQTT, o bebedouro verifica o nível da água, e se estiver baixo, aciona a torneira para enchê-lo. Outro exemplo de comunicação acontece com os termômetros, que regulam a temperatura acionando o atuador do aquecedor para buscar o conforto térmico no cômodo. Por fim, o computador e o telefone podem consultar as informações dos tópicos e publicar em tópicos que acionem as lâmpadas, facilitando a interação do usuário. Logo, em apenas um único cenário, são claras diferentes vantagens e a flexibilidade desse protocolo para comunicação de sistemas IoT.

Figura 11: Exemplo de aplicação em casa automatizada utilizando MQTT.



Fonte: Autoria Própria.

Por ser um protocolo de mensagens leve e eficiente, o MQTT é amplamente utilizado em aplicações de IoT para transferência de dados entre dispositivos com largura de banda limitada e baixo consumo de energia. O MQTT é um protocolo de mensagens que se destaca por ser leve e eficiente, tornando-se uma escolha comum para aplicações de IoT que envolvem transferência de dados entre dispositivos com restrições de largura de banda e consumo de energia. O *broker* MQTT não

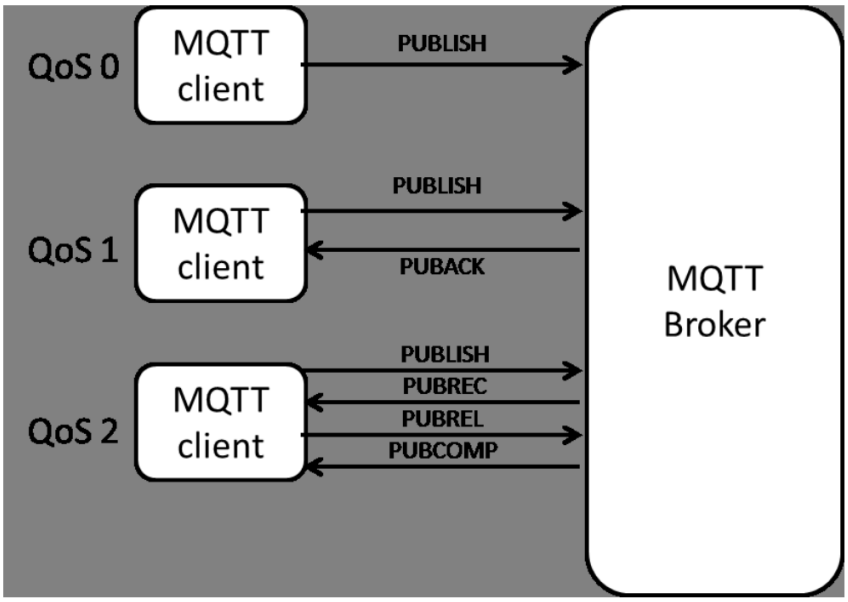
armazena as mensagens, dependendo da configuração, ele deixa o tópico com a última mensagem retida. Assim, se um cliente que assine o tópico tenha perdido a conexão na hora da publicação, ao se reconectar, é possível a verificação do último valor salvo (OASIS MQTT TECHNICAL COMMITTEE, 2019).

Outra característica presente no MQTT, é o *Quality of Service* (QoS) que, conforme é apresentado em sua especificação (OASIS MQTT TECHNICAL COMMITTEE, 2019), é dividido em 3 níveis ilustrados na Figura 12. O primeiro nível é o valor QoS 0, chamado de menor esforço, no qual o pacote *PUBLISH* QoS 0 é enviado apenas uma vez, sem a confirmação de recebimento e sem armazenamento da mensagem para a retransmissão. Já no segundo nível, com valor QoS 1, a mensagem deve ser recebida ao menos uma vez e deve ter a confirmação de entrega. Para tanto, o pacote *PUBLISH* QoS 1 é continuamente reenviado até o transmissor receber o pacote *PUBACK*, que é a confirmação da entrega. No terceiro nível, com o valor QoS 2, a mensagem deve ser recebida exatamente uma vez. O QoS 2 é o mais lento, pois quando o emissor envia um pacote *PUBLISH* QoS 2, o receptor deve enviar um pacote *PUBREC* que é a confirmação da entrega ao emissor. Ao receber este pacote o emissor deve enviar um pacote *PUBREL* confirmando o recebimento da mensagem de confirmação da entrega ao receptor, após isso o emissor envia um pacote *PUBCOMP* ao receptor que encerra a comunicação do QoS 2.

Cada nível de QoS do protocolo MQTT tem seus benefícios e usos. Um comparativo entre as diferenças no consumo de energia em nós IoT utilizando os diferentes níveis de QoS é discutido por Toldinas et al. (2019). Os autores utilizaram dois microcontroladores ESP 8266. O primeiro ESP 8266 configurado como *publisher* e o segundo configurado como *subscriber* foram conectados por meio de conexão Wi-Fi a um roteador e a um *broker* MQTT instalado em uma Raspberry PI. Os resultados dos testes apresentaram que o nível de QoS 0 consumiu 29% menos energia que o nível de QoS 1 e 87% menos energia que o nível de QoS 2. Já o nível de QoS 1 consumiu 45% menos energia que o nível de QoS 2. Além dessas diferenças, também existem outras como atraso e perda de pacotes que podem ocorrer conforme os níveis de QoS do MQTT (LEE et al., 2013).

O *keepalive* no MQTT é um mecanismo de manutenção de conexão entre o

Figura 12: Níveis de QoS no protocolo MQTT.



Fonte: (KODALI, 2016).

cliente e o *broker* MQTT. Ele é utilizado para garantir que a conexão entre o cliente e o *broker* permaneça ativa e saudável. Quando um cliente MQTT estabelece uma conexão com o *broker*, ele especifica um intervalo de tempo chamado de *keepalive*. Esse valor é definido pelo cliente durante o processo de conexão e indica ao *broker* por quanto tempo ele deve esperar por uma atividade do cliente antes de considerá-lo desconectado.

É importante mencionar que a desconexão por extrapolação do tempo de *keepalive* é uma medida de segurança para garantir a integridade da rede MQTT. A desconexão permite liberar recursos do *broker*, que estavam sendo alocados para uma conexão inativa.

O pacote MQTT é dividido em 3 partes (OASIS MQTT TECHNICAL COMMITTEE, 2019):

- Cabeçalho fixo: composto de 2 bytes representados na Figura 13, onde no primeiro byte os bits de 7 a 4 são utilizados para designar o tipo do pacote de controle. Os bits de 3 a 0 são utilizados como marcadores ou *flags*, que indicam a preferências que precedem o envio da mensagem, na versão 5 do MQTT, só são utilizados para se o tipo de pacote for 3, do tipo *PUBLISH*. O bit 3 faz referência

ao *Duplicate delivery of a PUBLISH packet* (DUP), que indica se o pacote é uma nova tentativa de entrega de um pacote. Os bits 2 e 1 indicam o nível de QoS sendo usado e o bit 0 se refere ao *retained message flag - RETAIN*, que indica se a mensagem deve ser armazenada pelo servidor. O comprimento restante é um byte inteiro e variável, que informa o número restante de bytes no pacote de controle atual;

- Cabeçalho variável: necessário para determinados tipos de pacotes de controle MQTT. Este cabeçalho variável é inserido entre o cabeçalho fixo e o *payload*, com o conteúdo variando conforme o tipo de pacote. Como exemplo, se o pacote de controle MQTT for do tipo *PUBLISH* e a configuração do QoS difere de zero, é inserido o cabeçalho variável com informações relacionadas às entregas;
- *Payload*: é a mensagem em si. Alguns pacotes de controle MQTT levam o *payload* ao final do pacote. O *payload* é opcional em um pacote de controle MQTT *PUBLISH*, necessário para pacotes de controle MQTT *CONNECT, SUBSCRIBE, SUBACK, UNSUBSCRIBE, UNSUBACK* e são inexistentes nos demais tipos de pacotes de controle MQTT.

Figura 13: Cabeçalho fixo do pacote MQTT.

Bit	7	6	5	4	3	2	1	0
byte 1	MQTT Control Packet type				Flags specific to each MQTT Control Packet type			
byte 2…	Remaining Length							

Fonte: (OASIS MQTT TECHNICAL COMMITTEE, 2019).

O MQTT na versão 5 possui 15 pacotes de controle, listados na Tabela 3, utilizados para o envio das informações pelas aplicações e para o gerenciamento da comunicação entre os clientes, servidores e o *broker* MQTT (OASIS MQTT TECHNICAL COMMITTEE, 2019).

2.7 TRABALHOS RELACIONADOS

Na literatura científica, foram encontrados estudos relacionados à coleta dos dados de colmeias de abelhas. Nesta seção são discutidas as principais características que podem ser observadas dos referidos trabalhos.

Tabela 3: Tipos de pacotes de controle do MQTT na versão 5.

Nome	Valor	Direção	Descrição
Reservado	0	Proibido	Reservado
CONNECT	1	Cliente para Servidor	Requisição conectar ao servidor
CONNACK	2	Servidor para Cliente	Reconhecimento da conexão
PUBLISH	3	Cliente para Servidor ou Servidor para Cliente	Publicar mensagem
PUBACK	4	Cliente para Servidor ou Servidor para Cliente	Reconhecimento da publicação (QoS 1)
PUBREC	5	Cliente para Servidor ou Servidor para Cliente	Publicação recebida (parte 1 do QoS=2)
PUBREL	6	Cliente para Servidor ou Servidor para Cliente	Publicação lançada (parte 2 do QoS=2)
PUBCOMP	7	Cliente para Servidor ou Servidor para Cliente	Publicação completa (parte 3 do QoS=2)
SUBSCRIBE	8	Cliente para Servidor	Pedido de inscrição
SUBACK	9	Servidor para Cliente	Reconhecimento de inscrição
UNSUBSCRIBE	10	Cliente para Servidor	Pedido cancelamento da inscrição
UNSUBACK	11	Servidor para Cliente	Reconhecimento de cancelamento da inscrição
PINGREQ	12	Cliente para Servidor	Requisição PING
PINGRESP	13	Servidor para Cliente	Resposta PING
DISCONNECT	14	Cliente para Servidor	Notificação de desconexão
AUTH	15	Cliente para Servidor ou Servidor para Cliente	Mecanismo de autenticação

Fonte: (OASIS MQTT TECHNICAL COMMITTEE, 2019).

Dispondo de um ESP 8266, Mahamud et al. (2019) criaram um sistema para a coleta de dados de sensores de umidade, temperatura, peso e frequência do som de colmeias de abelhas africanizadas. Esses dados são publicados via MQTT e podem ser consultados em um aplicativo para celular. Neste sistema não há o armazenamento de dados, além de não disparar gatilhos para alertar o criador. Também não foram reportados resultados da aplicação do sistema em uma colônia de abelhas.

Machhamer et al. (2020) realizaram a programação visual e elaboração de interface visual para a apresentação de dados coletados em abelhas africanizadas utilizando *Node-RED*[1]. Os sensores utilizados capturavam informações de umidade, temperatura, pressão, peso e gestos. O sensor de gestos foi utilizado para fazer a contagem de entrada e saída das abelhas. O sistema também possui uma placa de processamento chamada *IoT-Octopus*, que se trata de um *chip* ESP8266 com sensores já embutidos. Essas informações são publicadas em um *broker* MQTT em um Raspberry PI. Este Raspberry PI também possui um Sistema Gerenciador de Banco de Dados (SGBD) e a aplicação do *Node-RED*, que pode ser acessada por um navegador na rede local. Os autores também mencionam o treinamento de um algoritmo de aprendizado de máquina *Support Vector Machine* (SVM) para identificar anomalias a partir de dados simulados de uma colônia de abelhas com sucesso. Porém, não são apresentados os resultados dessa análise e não são disponibilizados os dados simulados gerados do sistema.

No trabalho de Anuar et al. (2019), é sugerida a coleta de dados de abelhas africanizadas e elaborado um procedimento para predizer problemas. Durante 36 horas foram coletados dados de um sensor de umidade, um sensor de temperatura e 4 células de carga para conseguir medir a massa em kg de uma colmeia. Os dados das células de carga apresentaram valores fora da distribuição, que foram lidos e armazenados erroneamente.

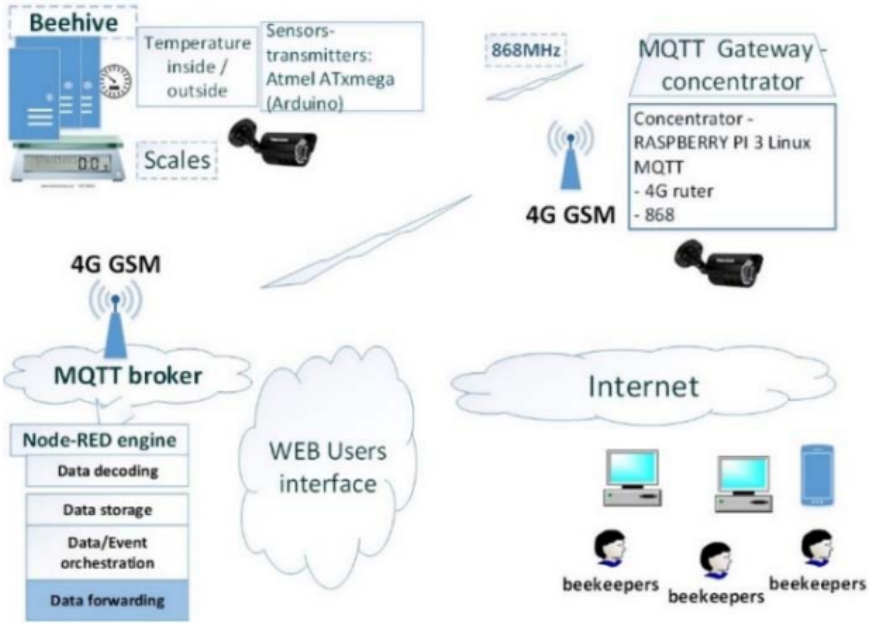
Para fazer um controle da produção das abelhas africanizadas, Fitzgerald et al. (2015) definem um protótipo que coleta informações de células de carga colocadas

[1] *Node-RED* é uma ferramenta de desenvolvimento com programação visual para modelar e manipular fluxo de dados e conectar dispositivos de *hardware* e APIs.

sob uma colmeia de abelhas. Esse sistema funciona como uma balança para verificar sua massa. O protótipo se conecta como um nó de uma rede de sensores *Zigbee* e possui bateria recarregada por uma célula solar. Porém, esta balança não ficou adequada para uso em colmeias ocupadas, por ainda não estar protegida contra intempéries. Os autores também não disponibilizaram os dados medidos em um *dataset*.

Zabasta et al. (2019) propõem um protótipo de um sistema autônomo de monitoramento de colmeias de abelhas africanizadas. Utilizando sensores de temperatura, umidade, carga e pressão, evidenciado a apicultura autônoma. Conforme pode ser observado na Figura 14, foi empregado um Arduino juntamente com os sensores que coletam os dados da colmeia de abelha e os transmitem via LoRa para um *broker* MQTT. Este *broker* MQTT publica as informações utilizando a rede de dados móvel em um *broker* MQTT em servidor na nuvem, que os encaminha para um servidor. Este servidor armazena os dados, além de possuir a ferramenta *Node-RED* que permite a visualização dos dados e a possibilidade de criar gatilhos.

Figura 14: Esquema do sistema de monitoramento da colmeia (ZABASTA et al., 2019).



Fonte: (ZABASTA et al., 2019).

Yusof et al. (2019) apresentam um protótipo para coleta de dados de umidade, temperatura, concentração de gás e massa de uma colmeia de abelhas sem ferrão que

não informam a espécie. Foi utilizado o MQTT para enviar as informações coletadas pelos sensores para uma aplicação *Node-RED* no servidor na nuvem. Os dados são exibidos pela interface do usuário criada no *Node-RED*, demonstrando que uma colmeia de abelhas pode ser monitorada com um sistema IoT.

Já Murphy et al. (2015c) propõem um sistema de coleta de dados de ruídos, temperatura e umidade de uma colônia de abelhas africanizadas. Para seu protótipo, foi construído um interruptor utilizando um microfone de baixo consumo de energia. Este interruptor é acionado com certas frequências de áudio, habilitando a captura de dados dos sensores, além de um microfone com boa qualidade. Também foi utilizado o ZigBee para transmitir os dados para um nó que envia as informações para serviço na nuvem utilizando a tecnologia *General Packet Radio Services* (GPRS), método de comunicação utilizado em redes móveis de segunda geração 2G.

Focados em garantir autonomia energética, Murphy et al. (2015b) utilizam células solares para elaborar um nó de uma rede de sensores sem fio. Os dados de abelhas africanizadas coletados de alguns sensores são transmitidos via Zigbee e posteriormente podem ser tratados e transmitidos via GPRS para servidor na nuvem. Coletam informações de dentro da colônia referentes a: dióxido de carbono, oxigênio, dióxido de nitrogênio, poluentes, temperatura, umidade relativa, pó, bateria e aceleração. Sugerem melhorar a eficiência energética e aumentar a rede de sensores.

Shaghaghi et al. (2019) buscam um sistema IoT viável que indique o momento da colheita de cada quadro móvel disposto para as abelhas produzirem o favo e armazenarem o mel, inseridos em uma melgueira vazia de uma caixa para abelhas africanizadas. Foram elaboradas estimativas de custo com 3 tipos de células de carga: medidor de tensão, de compressão e resistiva. Os autores optaram pela resistiva devido às suas dimensões, mas concluíram que o tipo de sensor não produz dados confiáveis para a aplicação.

O trabalho apresentado por Jiangyi et al. (2019) não foca em implementações de sistemas, mas aponta como tecnologias IoT podem auxiliar o criador de abelhas africanizadas. Nele, são indicadas possíveis aplicações de sensores para aferições de diferentes grandezas, como infravermelho, temperatura, umidade, entre outros, trazendo benefícios como redução de mão de obra e intervenção humana nas caixas

de abelhas.

Uma publicação que se destaca propondo a inserção de um giroscópio e um sensor magnético *reed* que verificam se a colmeia de abelhas africanizadas está sendo movimentada é a de Bellos et al. (2021). Os autores criaram uma base para a caixa de abelhas com um equipamento que transmite seus dados via LoRaWAN para um *gateway* LoRa, que os repassam pela rede de dados móvel para um servidor. O giroscópio consegue fornecer informações precisas sobre inclinação, permitindo identificar a ocorrência de eventos específicos, como a presença de animais em busca de alimento na caixa, oscilações causadas por ventos ou atividades suspeitas, como tentativas de roubo. O sensor magnético *reed* informa se a caixa foi retirada de cima do compartimento do equipamento, indicando uma queda ou roubo.

É descrito por Harun et al. (2015) que a existência de gatilhos que podem fazer uma colônia de abelhas sem ferrão abandonar o ninho. Entre os principais estão a identificação de fumaça tóxica ou a falta de alimentos. Estes gatilhos podem ser verificados utilizando sensores de temperatura, umidade e CO_2. Foi criado um sistema que captura dados de temperatura e umidade em colmeia de melíponas saudável, colmeia de melíponas com problema de saúde e do ambiente externo, para comparações. Os resultados são apresentados em gráficos comparativos, que tornaram visuais as diferenças entre as informações coletadas, apresentando a diferença na variação de temperatura e umidade entre o ambiente externo, a colmeia saudável e a com problema de saúde, indicando que essas informações podem ser usadas para avaliar possíveis problemas.

A utilização de sensores de temperatura, umidade e peso é questionada por Cousin et al. (2019) quanto à precisão na identificação de queda de população em uma colmeia de abelhas africanizadas. Para ter uma noção mais precisa, é proposto um sistema que conte as abelhas que entram e saem da caixa, microfones para a captação do áudio, que capta o som e analisa se existem sinais de enxameamento ou identifica ataque de vespas, também um sistema de alarme contra roubo. Os dados são enviados utilizando LoRa.

No trabalho de Zgank (2020), é apresentado o uso de uma base de dados

proveniente do projeto *Open Source Beehives*[2], em que áudios capturados em colmeias de abelhas africanizadas fornecem informações sobre comportamentos das abelhas. Os pesquisadores demonstraram que diferentes sons emitidos pelas abelhas estão relacionados a diferentes tipos de atividades. Os áudios coletados são submetidos à extração de características, utilizadas para comparações em etapas posteriores. Nesse contexto, os autores propõem a utilização de um sistema de IoT visando identificar o estado de saúde de uma colmeia, e eles demonstraram experimentalmente a viabilidade dessa abordagem. Esse estudo ressalta a importância do uso de técnicas avançadas, como processamento de sinais e análise de áudio, para monitorar e entender melhor as colônias de abelhas e seu comportamento.

Braga et al. (2019) utilizam uma base de dados de sensores de temperatura, umidade, som e de peso coletados por um projeto do Reino Unido chamado Arnia. O projeto Arnia coleta informações de colmeias de abelhas africanizadas monitoradas em 25 países. O artigo aponta a verificação da termorregulação do ninho, que pode informar a saúde da colmeia, e propõem o uso de um algoritmo que faz predições. Estas predições permitem a intervenção do apicultor antes que as abelhas pereçam por frio ou calor. Também foi realizado um tratamento prévio dos dados, reduzindo a quantidade a ser armazenada ou transferida.

Foram compiladas certas características das pesquisas acima, descritas na Tabela 4. Ao analisar a literatura especializada sobre coleta de dados em colmeias de abelhas, nota-se que os sensores mais utilizados para monitoramento são aqueles voltados à medição de temperatura, umidade e carga. No que tange à comunicação entre os dispositivos, observa-se que a tecnologia Wi-Fi e o protocolo MQTT são amplamente empregados. No entanto, é comum que os dados coletados sejam armazenados em cartões SD, os quais podem apresentar problemas de leitura e gravação que impossibilitem sua recuperação. Ademais, destaca-se que a maioria dos trabalhos investigados restringe-se à análise local dos dados, sem disponibilizar o *dataset* para identificação de anomalias nas colmeias. Cabe mencionar, ainda, que a maioria dos estudos se concentra em abelhas africanizadas.

[2]O *dataset*, pertencente ao projeto OSBH, antigamente era disponibilizado na página `https://www.osbeehives.com/`, porém, não está mais acessível.

Tabela 4: Principais características dos trabalhos relacionados.

Trabalho	Sensores	Comunicação	Armazenamento dos dados	Exibição dos dados	*Dataset* disponível	Tipo de Abelha
(MAHAMUD et al., 2019)	Temperatura, umidade, som, gás e carga	MQTT, LoRaWAN e Wi-Fi	Não informado	Aplicativo móvel	Não	Africanizadas
(MACHHAMER et al., 2020)	Temperatura, umidade, pressão e gestos	MQTT e Wi-Fi	Raspberry Pi com cartão SD	E-mail se anomalia é detectada	Não	Africanizadas
(ANUAR et al., 2019)	Temperatura, umidade e carga	Wi-Fi	Não informado	Aplicativo móvel	Não	Africanizadas
(FITZGERALD et al., 2015)	Temperatura, umidade, fumaça e carga	ZigBee	Não informado	Não	Não	Africanizadas
(YUSOF et al., 2019)	Temperatura, umidade, gás e carga	MQTT e Wi-Fi	Não informado	*Dashboard* NodeRED	Não	ASF
(MURPHY et al., 2015c)	Temperatura, umidade e som	ZigBee e GSM	Cartão SD no nó IoT	Não	Não	Africanizadas
(MURPHY et al., 2015b)	Temperatura, umidade, gás, carga de bateria, acelerômetro e poeira	ZigBee e GPRS	Cartão SD	Não	Não	Africanizadas
(SHAGHAGHI et al., 2019)	Carga	Wi-Fi	SGBD local	Não	Não	Africanizadas
(OCHOA et al., 2019)	Temperatura, umidade e carga	Wi-Fi e MQTT	Não informado	Gráficos e *dashboards* em tempo real	Não	Africanizadas
(ZABASTA et al., 2019)	Temperatura, umidade, carga e pressão atmosférica	MQTT, Wi-Fi e GSM	SGBD local	*Dashboard* NodeRED	Não	Africanizadas
(BELLOS et al., 2021)	Temperatura, umidade e giroscópio	LoRaWAN e GSM	Não disponível	Não informado	Não	Africanizadas
(BRAGA et al., 2019)	Não	Não	Não	Não	Não	Africanizadas
Sistema proposto	Temperatura e umidade	Wi-Fi e MQTT	Em cartão SD e servidor na nuvem	Dispositivo externo e *Dashboards* em aplicação Web	Público	ASF

Fonte: Autoria Própria.

Após concluir a pesquisa da literatura relacionada, foi constatado o uso de tipos de sensores específicos em caixas de abelhas africanizadas. A seguir são apresentados exemplos desses sensores e possíveis aplicações para o monitoramento de caixas de ASF (MAHAMUD et al., 2019; MACHHAMER et al., 2020; MURPHY et al., 2015b):

- Sensor de temperatura: pode ser utilizado para coletar os dados internos e os dados externos de uma caixa de ASF. Comparando as varições entre seus valores, espera-se que possam ser identificados anomalias na caixa ou a necessidade do aquecimento em períodos de clima frio;
- Sensor de umidade: da mesma forma que o sensor de temperatura, é esperado que, verificando ocorrências de alterações bruscas, sejam identificadas anomalias;
- Células de carga: podem ser utilizadas para verificar a massa de uma caixa de ASF. Caso o valor fique repentinamente próximo ao zero, estaria demonstrando que a caixa não estaria mais sobre a célula, desta forma acionando um alarme para o criador verificar se o vento ou um animal derrubou a caixa, até mesmo se a caixa foi furtada;
- Captação de áudio: um microfone que possa ser utilizado para transmitir o áudio de uma caixa de ASF de forma que possam ser extraídas características devem possibilitar a identificação de situações como enxameamento, ataques de predadores ou mudanças no comportamento das abelhas;
- Sensor de luz ou iluminação: pode ser elaborado um sistema de alarme com barreiras luminosas ou informar quando houver luz incidindo sobre a caixa;
- Sensor ultrassônico: com ele pode ser medida a altura do ninho utilizando um sensor deste tipo, mas depende de fatores para ser funcional, tais como: a necessidade de haver um sensor monitorando em cima e outro em baixo do ninho e as abelhas não fechem com cera ou outro material os módulos referentes ao sinal e leitura. Além disso, pode ser colocado para verificar se identifica movimento de abelhas na entrada da caixa;

- Sensor magnético: o sensor trabalha em conjunto com um ímã e podendo ser utilizado para verificar se a caixa continua no local ou foi movimentada.

No Capítulo 3, é apresentado o desenvolvimento do módulo de coleta, utilizando sensores que aferem a temperatura e a umidade relativa do ar. Além disso, foi desenvolvido um dispositivo externo dedicado à visualização das informações coletadas e apresentação de alarmes. Para as funções de armazenamento, sincronismo e ativação de alarmes, foram configurados sistemas de névoa e nuvem. Os detalhes sobre essas configurações são descritos, juntamente com os procedimentos adotados durante a implementação do sistema.

3 DESENVOLVIMENTO

Neste capítulo, são apresentados detalhes sobre o desenvolvimento do sistema para a coleta e gerenciamento de dados de caixas de ASF, incluindo informações sobre os componentes de *hardware* e *software* escolhidos, bem como a integração dos sensores com a névoa e com a nuvem.

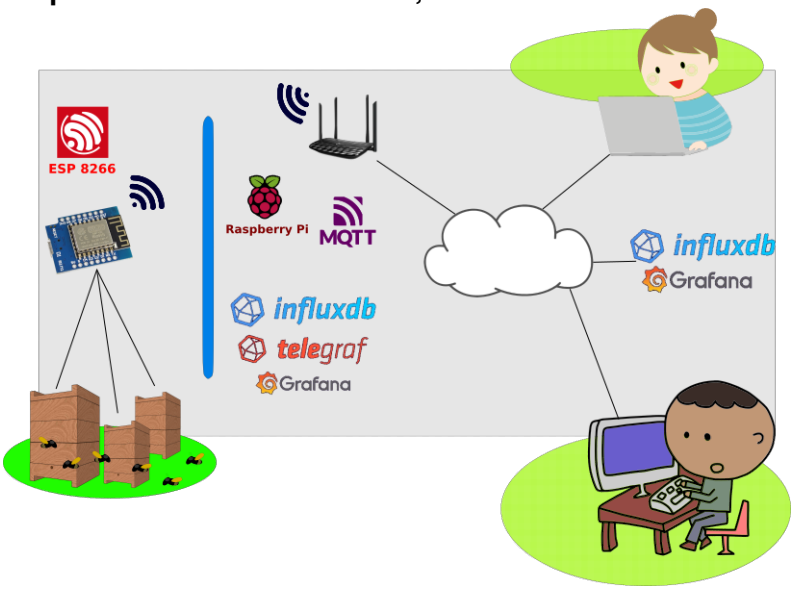
Optou-se pelo uso de conexões Wi-Fi no sistema devido ao custo reduzido em relação a outras tecnologias de comunicação sem fio, além do microcontrolador ESP D1 mini possuir conexão Wi-Fi nativa. Outro motivo que reforça essa escolha é o fato de se tratar de uma implantação em um meliponário urbano, onde o Wi-Fi é a tecnologia de comunicação sem fio mais comum em residências. O consumo energético da tecnologia não é uma restrição, uma vez que os dispositivos são alimentados por fontes de energia conectadas diretamente na rede elétrica.

Os testes do sistema foram realizados em um meliponário urbano chamado Reual (registro ADAPAR n° 142055616), localizado na cidade de Guarapuava, no estado do Paraná, aproximadamente a 25°23′36″ de latitude sul e 51°27′19″ de longitude oeste. Guarapuava é um município localizado no centro-sul do estado.

A elaboração inicial do sistema foi realizada seguindo o modelo apresentado na Figura 15. O início da implementação do sistema contemplou a elaboração de um módulo coletor, acoplável à caixa racional modelo INPA. Esse módulo foi desenvolvido para acomodar sensores conectados a um microcontrolador ESP D1 mini. As comunicações deste módulo são executadas via rede Wi-Fi utilizando o protocolo MQTT. O conceito de névoa, apresentado no Capítulo 2, é realizado pela Raspberry Pi, onde foram implementados serviços e aplicações que realizam o tratamento, armazenamento e transmissão de dados. Além disso, a Raspberry Pi também se comunica com a nuvem para manter os dados sincronizados. A exibição

dos dados é realizada por meio de uma aplicação Web utilizando o Grafana, que, após a configuração das necessidades, gera gráficos em painéis em um ou vários *dashboards*.

Figura 15: Arquitetura inicial do sistema, alterada durante o desenvolvimento.



Fonte: Autoria própria.

Durante o desenvolvimento, constatou-se que a arquitetura concebida para o sistema exigiria ajustes, os quais serão abordados ao longo do texto. Essas modificações contribuíram para aumentar a estabilidade e efetividade do sistema. A arquitetura final é descrita na Seção 3.4.

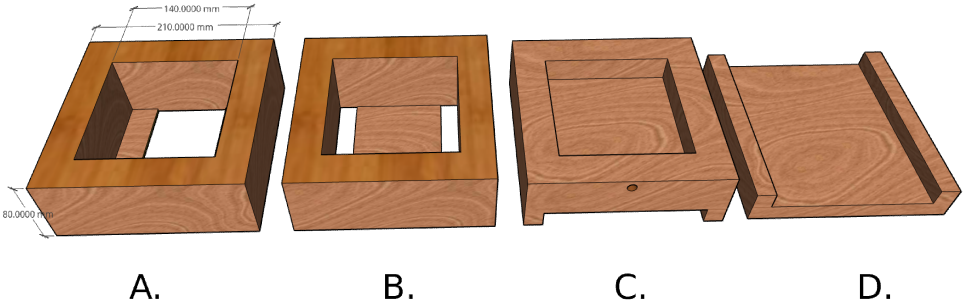
Ao considerar o escopo do projeto, percebeu-se que ele se enquadra na arquitetura IoT de 5 camadas. Os sensores, juntamente com os controladores, compõem a camada de percepção. Na camada de rede, a conexão entre a camada de percepção e a camada de serviço é realizada utilizando a conexão Wi-Fi. A camada de serviço é composta pelas aplicações de armazenamento de dados, pelo SGBD PostgreSQL e pelo *broker* MQTT. A camada de aplicação é composta pela plataforma de visualizações Grafana. Já na camada de negócios, são realizados os testes dos dados, acionamento de gatilhos e demais funções que envolvam o gerenciamento das camadas inferiores. Todos esses elementos serão discutidas nas seções seguintes.

3.1 CAIXAS DE ABELHAS COM SENSORES

As caixas de abelhas com sensores são responsáveis por abrigar as ASF e possuem sensores capazes de monitorar a colmeia. Os dados coletados pelos sensores são enviados para a aplicação em névoa que os armazena, testa, compara e executa gatilhos.

Optou-se pela utilização de uma caixa modelo INPA adaptada para a criação de abelhas Mandaçaia, por ser um modelo modelo de utilização comum na região, além da indicação de criadores regionais sobre quais medidas internas e espessura de parede utilizar. Então foi construída uma caixa INPA com as medidas internas dos módulos de 14 cm de largura e de profundidade e 8 cm de altura, totalizando um volume interno de 1568 cm³ por módulo. Foi utilizada madeira de pinus, seca naturalmente, sem aditivos químicos, com espessura de 3,5 cm. A adaptação foi feita da seguinte forma: a entrada é movida do módulo de ninho para o fundo, o que dá mais liberdade para mover o módulo do ninho. No fundo, é feita uma parede nas mesmas medidas internas e externas de um módulo, com 2 cm de altura. Os módulos podem ser vistos na Figura 16, que ilustra em A. o sobre ninho, em B. o ninho e a melgueira, em C. o fundo com a entrada e em D. a tampa.

Figura 16: Arquitetura modular de uma caixa modelo INPA adaptada.

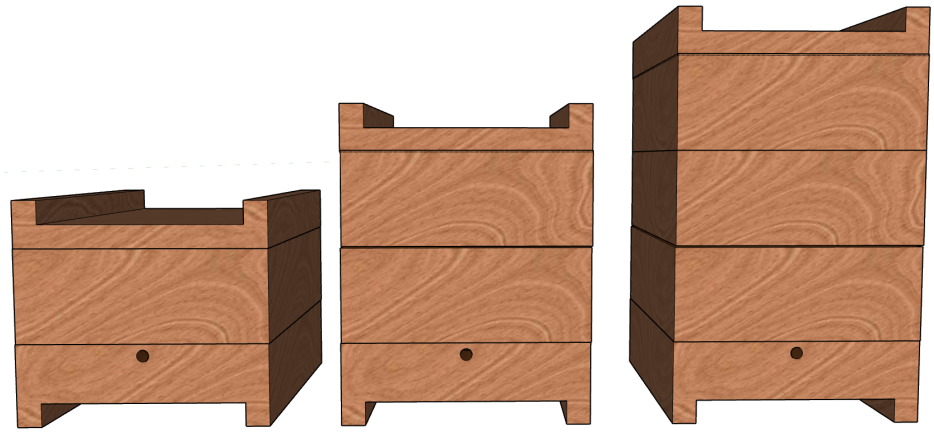


Fonte: Autoria própria.

Este modelo de caixa possui versatilidade, podendo ser utilizado tanto para a divisão de colônias quanto para a produção de mel. Quando o objetivo é a divisão de colônias, a caixa é geralmente iniciada com o fundo, módulo de ninho e tampa. À medida que a colônia se desenvolve ao longo do tempo, o sobre ninho é adicionado.

A postura das abelhas é transferida de um módulo para outro. Se houver discos de cria suficientes nos dois módulos, o enxame pode ser dividido (NOGUEIRA-NETO et al., 1986; VILLAS-BÔAS, 2012). A divisão é feita sendo alocado o sobre ninho em outra tampa e fundo. Dessa forma, se tudo correr bem, haverá duas colônias em desenvolvimento, como apresentada na Figura 17 na caixa à direita. Se o foco for produção de mel, é colocado um ou mais módulos de melgueira onde as abelhas depositam mel e pólen.

Figura 17: Caixas INPA com 1, 2 e 3 módulos instalados.



Fonte: Autoria própria.

Optou-se pelo uso de uma caixa modular devido à facilidade de instalação dos sensores nesse tipo de modelo. Um dos módulos da caixa foi adaptado para acomodar o microcontrolador, onde os sensores serão conectados. Nos experimentos, foram utilizadas duas caixas de ASF do modelo INPA. Cada caixa foi equipada com um módulo coletor. Na primeira caixa, uma colônia de abelhas da espécie MQQ foi estabelecida e classificada como “forte”, um termo comum na meliponicultura que indica a presença de mel e pólen disponíveis na caixa, ademais, essa colônia possui uma rainha que realiza posturas em discos que variam desde os mais novos até os nascentes, além de uma população de abelhas conforme as características da espécie. Enquanto isso, a segunda caixa foi mantida sem a presença de abelhas. O módulo de coleta da primeira caixa foi denominado caixa01 e contou com sensores posicionados tanto internamente quanto externamente à caixa, sendo as coletas externas designadas como externa. Já o módulo de coleta da segunda caixa, denominado caixa02, realizou coletas apenas internamente.

3.1.1 *HARDWARE* DO SISTEMA DE COLETA DE DADOS

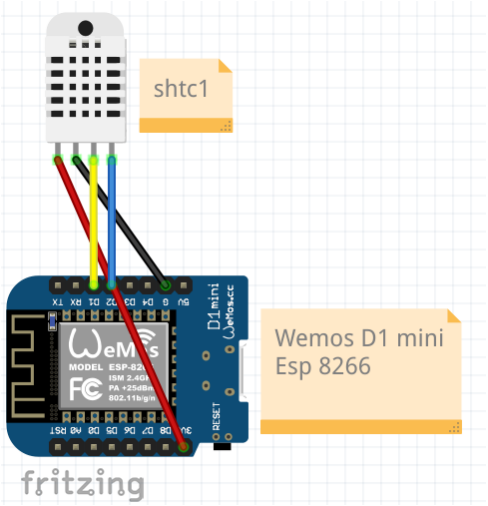
Para viabilizar a coleta dos dados no sistema proposto, foi instalado no módulo de coleta o sensor composto por um microcontrolador ESP 8266, que possui 1 porta analógica e 9 portas digitais *Pulse Width Modulation* (PWM) para as conexões com sensores ou atuadores e Wi-Fi embutido. Optou-se pelo modelo ESP D1 mini por ser uma plataforma de *hardware* e *software* com código aberto, que utiliza o controlador ESP12s, com 4 MB de memória *flash*, 11 portas digitais, 1 analógica, suporte a rede Wi-Fi 802.11b/g/n e conector micro-USB para programação e alimentação (DATASHEET, 2015). O modelo Weemos D1 mini pode ser alimentado com tensões de 3,3 ou 5 volts e possui um conversor serial USB acessível por meio de uma conexão micro USB.

Foram instalados sensores SHTC3 para coleta de dados sobre umidade e temperatura. Estes sensores possuem as seguintes características: faixa de operação de umidade de 0 a 100%, faixa de operação da temperatura entre -40 ℃ a 125 ℃ e margem de erro de ±2% RH / ± 0,2 ℃, (SENSIRION SMART SENSORS SOLUTION, 2021). Cada sensor SHTC3 é conectado em um microcontrolador ESP D1 mini, sendo utilizado um conjunto para a captura dos dados internos e um conjunto para os dados externos, pressupondo que desta forma possam ser extraídas informações relevantes para o apoio na identificação de anomalias na criação.

O sensor SHTC3 utiliza o protocolo I2C e por isso ele deve ser conectado nas portas D1 e D2 no ESP D1 mini. O esquema que mostra o sensor ligado na ESP D1 mini pode ser observado na Figura 18. O conjunto de ESP D1 mini conectado a um sensor SHTC3 é mostrado na direita da Figura 19, sendo apresentada também à esquerda o Raspbery Pi 3B+, usado para a infraestrutura de névoa.

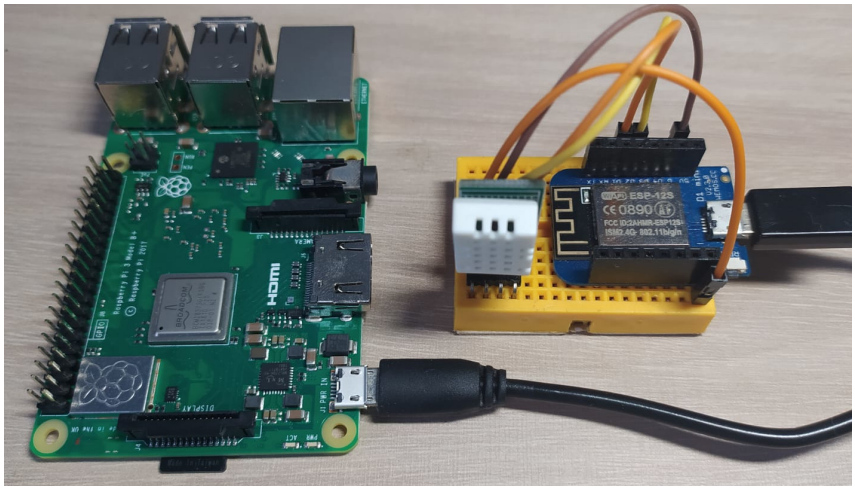
Foram realizados testes instalando sensores diretamente em um módulo de sobre ninho, que seria ocupado pelas abelhas. No entanto, essa prática permitiu que as abelhas isolassem os sensores com cera, o que tornou as leituras imprecisas e dificultou o manejo das abelhas em caso de divisão da colônia. Diante disso, essa estratégia foi alterada e um novo protótipo de módulo coletor foi elaborado. Esse novo módulo pode ser colocado entre a tampa e o módulo mais alto da caixa, evitando a

Figura 18: Ligação do sensor SHTC3 no ESP D1 mini.



Fonte: Autoria própria.

Figura 19: Raspberry Pi 3B+ e ESP D1 mini com sensor SHTC3.



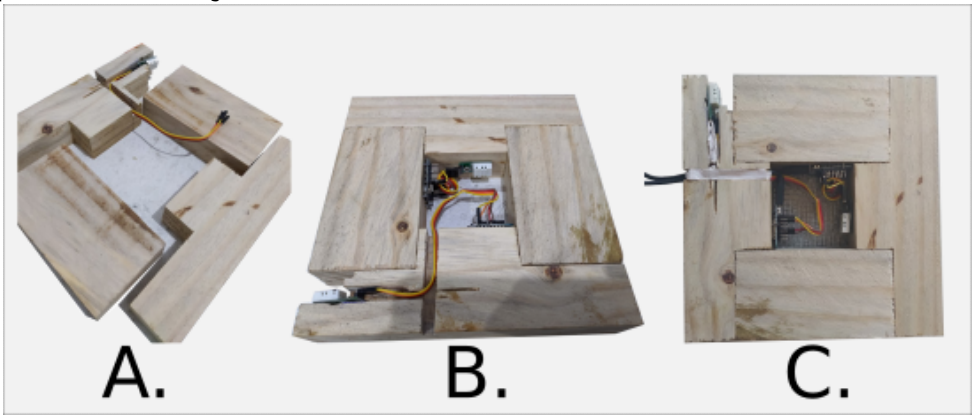
Fonte: Autoria própria.

interferência das abelhas e garantindo leituras mais precisas dos sensores.

3.1.2 MONTAGEM DO MÓDULO COLETOR

Para construir o novo módulo coletor, foi mantida a largura e a profundidade externa de 21 cm, correspondentes a um módulo da caixa modelo INPA onde o módulo coletor foi instalado. Buscando reduzir o volume onde as abelhas não possuem acesso, as medidas internas utilizadas foram 7 cm de largura e profundidade, e altura de 3,2 cm. Dessa forma, o espaço interno, utilizado para acomodar os ESP D1 mini e os sensores SHTC3, possui as dimensões de 7 cm por 7 cm por 3,2 cm, totalizando um volume de 157 ml. A Figura 20 apresenta as fases da montagem do módulo coletor: em A. estão as peças usinadas, em B. a adição dos sensores SHTC3, um posicionado internamente e outro posicionado para coletar as informações do lado externo da caixa. Os microcontroladores ESP D1 mini são exibidos em C. é apresentado o módulo coletor montado com cabeamento externo para a alimentação. Dessa forma, foi utilizado um kit ESP D1 Mini em conjunto com um sensor SHTC3 para capturar os dados internos, enquanto outro kit ESP D1 Mini em conjunto com um sensor SHTC3 foi empregado para capturar os dados externos. Essa abordagem permitiu a coleta de informações tanto do ambiente interno quanto do ambiente externo.

Figura 20: Fases da montagem do módulo coletor: em A. estão as peças usinadas, em B. a adição dos sensores e dos ESPs D1 mini e em C. o módulo montado com cabeamento externo para a alimentação.



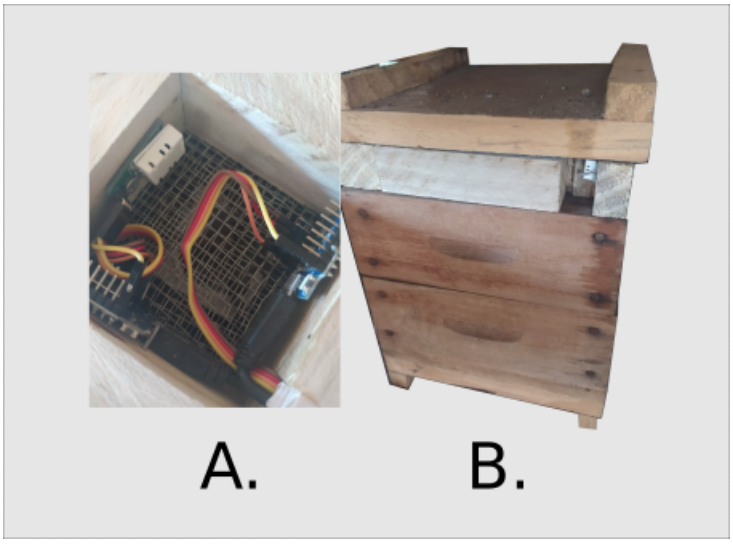
Fonte: Autoria própria.

Na parte inferior do módulo coletor, foi colocada uma tela para impedir que as abelhas ocupem o seu interior. Foi idealizado um módulo coletor que permita a

coleta de dados internos da caixa, bem como um sensor posicionado para coletar informações externas. Isso viabiliza um parâmetro para verificar esses dados. Para isso, foram utilizados 2 ESP D1 mini e 2 sensores SHTC3. Para a alocação do sensor SHTC3 responsável pela coleta de informações sobre temperatura e umidade externas, foi feita uma abertura na madeira, de forma que o sensor fique acomodado e protegido. Já dentro do módulo, encontra-se o sensor SHTC3 que coletará os dados internos da caixa.

Na Figura 21 pode ser vista em A. uma tela entre o módulo coletor e o segundo módulo da caixa, que evita que as abelhas tenham contato com os equipamentos eletrônicos. Em B. é visualizado o módulo coletor instalado na caixa, logo abaixo da tampa. Na possibilidade de haver mais caixas próximas fisicamente, pode ser feita a instalação de um módulo coletor que transmita somente os dados internos, pois os dados externos já estarão sendo coletados em um módulo em outra caixa.

Figura 21: Módulo coletor instalado na caixa INPA: A. apresenta a visão interna do módulo e B. apresenta o módulo coletor instalado na caixa.



Fonte: Autoria própria.

Para coletar dados de dois sensores SHTC3, foi necessário utilizar dois módulos ESP D1 mini. Cada sensor possui um endereço I2C fixo e o modelo adquirido não permite a alteração desse endereço. Como o ESP D1 mini possui apenas um barramento I2C, a solução foi utilizar dois módulos para conectar cada sensor a um barramento separado. Dessa forma, os dois sensores podem ser acessados independentemente pelos módulos ESP D1 mini, permitindo a coleta de

dados simultânea e sem conflitos.

3.1.3 COLETA DE DADOS DOS SENSORES

Para viabilizar a coleta, organização e publicação dos dados em um tópico MQTT provenientes dos sensores conectados nos ESP D1 Mini, foi necessário desenvolver e instalar uma aplicação específica em cada dispositivo ESP D1 Mini. Essa aplicação foi desenvolvida para capturar os dados dos sensores, organizá-los adequadamente e enviá-los para o respectivo tópico MQTT. Dessa forma, os dados coletados podem ser facilmente transmitidos e disponibilizados para outros dispositivos ou sistemas que estejam inscritos no mesmo tópico MQTT, permitindo uma comunicação eficiente e centralizada dos dados sensoriais.

Inicialmente o desenvolvimento da aplicação de coleta dos dados dos sensores, para utilização no ESP D1 mini, foi realizada utilizando-se a IDE do Arduino. Porém, as bibliotecas MQTT, PubSubClient e Adafruit MQTT, disponíveis para Arduino, não permitiam no início do ano de 2022, a execução da publicação em tópicos MQTT com o QoS diferente de zero se o microcontrolador utilizado for um ESP 8266, que é o microcontrolador do ESP D1 mini. Esse motivo levou a pesquisa de alternativas como o Tasmota, Arendst (2022), um *firmware* de código aberto para dispositivos ESP, mas que também não possibilita a publicação de tópicos MQTT utilizando QoS 1 no ESP D1 mini. Outra solução encontrada foi o MicroPython, George (2022), uma implementação de Python 3 compatível com dispositivos ESP. Com o Micropython, foi possível compilar uma biblioteca MQTT para o ESP D1 mini que permite a utilização do QoS nos níveis 0 e 1. Esses motivos levaram a escolha do MicroPython como alternativa para o desenvolvimento.

Durante a coleta de dados reais do trabalho, até março de 2023, os dados de temperatura e umidade de cada sensor eram publicados a cada 60 segundos no *broker* MQTT utilizando o QoS 0. No entanto, observou-se que os sensores deixavam de transmitir as coletas em intervalos que variavam entre 4 e 7 dias, voltando a funcionar apenas quando reiniciados. Esse problema ocorria devido a questões relacionadas à memória do dispositivo e foi resolvido com a função *collect* da biblioteca GC, presente nativamente no Micropython. Essa função é responsável por desfragmentar

a memória do microcontrolador e, portanto, passou a ser chamada no início da função responsável pela coleta dos dados dos sensores. Essa solução permitiu melhorar a estabilidade da coleta de dados e garantir o envio contínuo das informações ao *broker* MQTT.

Em março de 2023 foi obtido sucesso na implementação o QoS 1 no ESP D1 Mini utilizando a biblioteca MicroPython *Asynchronous* MQTT (HINCH, 2019). Após esta implementação, os sensores paravam de enviar dados de uma a duas vezes por dia, sendo necessário reinicializá-los manualmente. Em pesquisas foram observadas a utilização de uma função nomeada *watchdog timer* (WDT) para resolver o problema de travamentos em microcontroladores. Dechmunee et al. (2017) explicam a função WDT como sendo um temporizador utilizado em sistemas computacionais para detectar falhas e reiniciar o sistema em caso de erro. O WDT pode ser acionado por meio de uma porta de controle, por meio de uma instrução de linguagem de máquina ou por meio de um *driver* de dispositivo em sistemas operacionais como o GNU/Linux. A configuração do WDT pode ser feita por meio de bibliotecas de temporizadores, estabelecendo um valor inicial ao temporizador. Durante a execução do código o temporizador é reiniciado. Caso ocorra qualquer circunstância em que o temporizador não seja reiniciado, o dispositivo é reiniciado como medida de segurança.

O MicroPython possui uma classe nativa chamada WDT (*Watchdog Timer*). Ao tentar implementá-la no ESP D1 mini, foi identificada uma diferença na implementação em comparação a outras plataformas que suportam o MicroPython, como o ESP32, Raspberry Pi Pico, entre outras. Nessas outras plataformas, a classe WDT permite definir o valor do temporizador ao iniciá-la. No entanto, no ESP D1 mini, devido ao modelo do microcontrolador ser o ESP8266, o valor do tempo da classe WDT é fixo em 3 segundos.

Na lógica utilizada no sistema de coleta de dados dos módulos coletores, o temporizador do WDT é inicializado com o tempo definido para o período entre as coletas, somado de 5 segundos. Essa lógica é aplicada porque a chamada da função de reinício do temporizador é realizada na função de publicação de dados, que ocorre de forma contínua, com cada ciclo tendo o tempo definido no tópico MQTT

“sensores/frequencia” que está entre 5 e 3600 segundos. No entanto, o valor fixo de 3 segundos para o controlador ESP8266, na classe WDT do MicroPython, inviabilizou sua utilização. Como solução, foi desenvolvida uma função em MicroPython utilizando o exemplo de Kock (2018), com um funcionamento semelhante ao da classe WDT, utilizando interrupções do próprio ESP8266. Com essa função personalizada, foi possível definir o tempo do WDT para uma soma de 5 segundos com o tempo definido para o período de coleta. O contador do temporizador é reiniciado a cada coleta de dados, e o módulo coletor é reiniciado caso o temporizador não seja reiniciado. Essa abordagem possibilitou contornar as limitações do temporizador fixo do ESP8266 e garantir o correto funcionamento do sistema de coleta de dados.

Outro ponto importante que precisou ser observado foi o sincronismo dos relógios nos sensores. Na implementação inicial, utilizando o Telegraf, o módulo coletor publicava no tópico MQTT os valores das unidades de medida, utilizando o carimbo de data e hora do momento em que a publicação era realizada no *broker MQTT*. No entanto, considerando o uso do QoS 0, caso o módulo coletor parasse de se comunicar com a Raspberry Pi, os dados seriam perdidos e seriam somente enviados novamente após a reconexão, resultando na perda das coletas realizadas durante esse intervalo. Essa era uma característica não editável do Telegraf.

Após a alteração para o uso do QoS 1, a aplicação com o MicroPython no módulo coletor teve que ser adaptada. Essa aplicação é responsável por inserir um carimbo de data e hora no momento da coleta dos dados e, em seguida, publicá-los no tópico MQTT correspondente. A adequação da aplicação com o MicroPython envolveu a implementação de um mecanismo para garantir que cada dado coletado seja registrado com o carimbo de data e hora atualizado. Isso permite uma análise precisa e temporal dos valores coletados ao longo do tempo.

Além disso, foi desenvolvida uma função em Python no subsistema de névoa para conectar o tópico MQTT com a inserção dos dados no banco de dados. Essa função é responsável por receber os dados publicados no tópico MQTT e realizar a inserção dos mesmos no banco de dados. Para sincronizar os relógios dos módulos coletores, foi implementado na Raspberry Pi um servidor *Network Time Protocol* (NTP). Quando o módulo coletor conecta na rede Wi-Fi e também de hora em hora,

ele sincroniza seu relógio com o relógio da Raspberry Pi para publicar no tópico MQTT o horário em que ele efetuou a leitura.

3.2 SUBSISTEMA DA NÉVOA

Realizou-se o desenvolvimento uma aplicação em Python para o funcionamento do subsistema da névoa, com funções específicas. Uma dessas funções tem a finalidade de tratar e inserir no banco de dados as publicações provenientes do módulo coletor, por meio do tópico MQTT. Outra função é responsável por verificar as informações das coletas armazenadas no banco de dados e enviá-las para exibição no painel do meliponário. Além disso, uma função realiza testes nos valores de temperatura e umidade relativa das caixas, acionando alarmes no painel do meliponário caso algum valor esteja fora da faixa ideal. Por fim, há uma função dedicada à sincronização entre o banco de dados em névoa e o banco de dados em nuvem. As aplicações são executadas individualmente em contêineres Docker o que permitiu a alteração dos códigos isoladamente. Utilizam o protocolo MQTT para realizar a comunicação.

Para viabilizar as funções de computação em névoa, foi adotado um minicomputador Raspberry Pi versão 3B+, equipado com processador armv7l de 4 núcleos (600 a 1200 MHz) e 1 GB de memória *Random Access Memory* RAM. Para armazenamento local, foi utilizado um cartão *secure digital* (SD) de 32 GB classe 10. Durante os testes de conectividade Wi-Fi à rede interna, foram observadas perda de dados ao longo do tempo. Por outro lado, quando o dispositivo foi conectado pelo cabo Ethernet, não foram registradas perdas. Ainda assim seria possível utilizar uma conexão Wi-Fi, uma vez que os problemas do roteador específico tenham sido superados. Mas que isso não foi realizado e foi mantida a configuração com o cabo Ethernet para os experimentos. O sistema operacional selecionado foi o *Raspbian GNU/Linux 11 (bullseye)*. Além disso, foi instalado e configurado o *broker* MQTT Mosquitto, bem como o SGBD.

3.2.1 COMUNICAÇÃO COM MQTT

De fato, existem várias tecnologias de comunicação sem fio disponíveis para sensores, como LoRaWAN, ZigBee e Bluetooth. No entanto, no desenvolvimento do protótipo em questão, a conexão de rede entre os módulos coletores e o Raspberry Pi foi realizada utilizando a tecnologia Wi-Fi. Essa escolha foi feita levando em consideração a acessibilidade e a ampla disponibilidade dessa tecnologia. O Wi-Fi é uma tecnologia de rede sem fio amplamente utilizada e difundida, o que torna provável que o criador possua dispositivos com suporte a Wi-Fi em sua infraestrutura. Por esse motivo, a decisão de utilizar a conexão Wi-Fi para a comunicação entre os módulos coletores e a Raspberry Pi foi tomada.

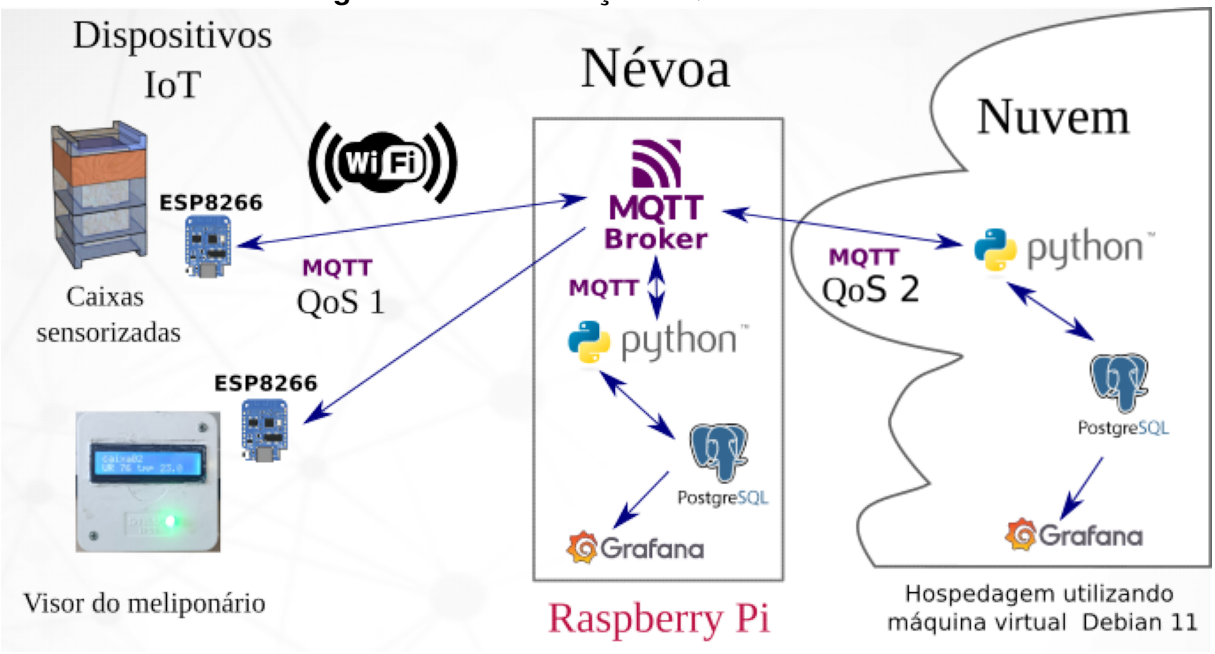
Para a comunicação na camada de serviços, optou-se pela exploração do protocolo MQTT, pelo protocolo possuir um *payload* leve, o que é importante ao utilizar conexões com baixa largura de banda, além de possuir diferentes níveis de QoS que são adequados à proposta do sistema.

A comunicação entre o microcontrolador e o *broker* MQTT local é estabelecida utilizando o QoS 1. Com o QoS 1 no MQTT, uma informação é publicada periodicamente até que o *broker* MQTT envie o pacote denominado PUBREC, confirmando o recebimento. Para enviar as informações da aplicação em névoa para o servidor na nuvem, é utilizado o protocolo MQTT com o QoS 2. Isso deve garantir que os dados sejam transmitidos de forma confiável e que não sejam perdidos durante o processo de comunicação.

Na Raspberry Pi está instalado o Sistema Operacional *Raspbian GNU/Linux* rodando o MQTT *broker* Mosquitto que gerencia a publicação e subscrição dos dados. No Raspberry Pi também foram instalados o Grafana, e o SGBD PostgreSQL. Na Figura 22 é apresentado um diagrama da comunicação entre os dispositivos, névoa e nuvem. Nela são apresentados os níveis de QoS utilizados nas comunicações entre os dispositivos locais e a névoa, também entre a névoa e nuvem.

O funcionamento da coleta dos dados dos módulos coletores, ocorrem da seguinte forma: os módulos coletores com o Microcontrolador ESP 8266, conectam através da rede Wi-Fi e assinam o tópico MQTT “/reual/sensores/frequencia” onde

Figura 22: Comunicação MQTT no sistema.

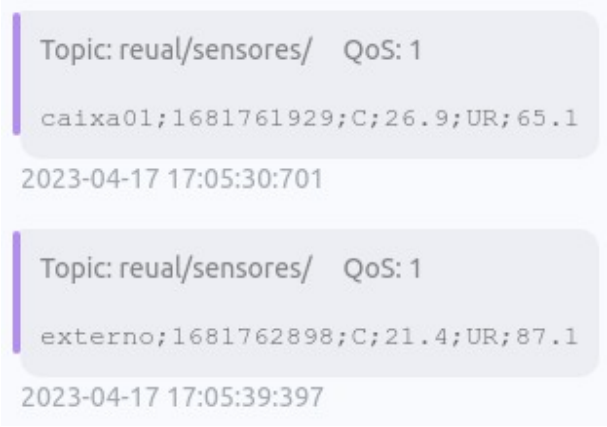


Fonte: Autoria própria.

é publicado um número inteiro entre 5 e 3600, usado para definir, em segundos, o período para determinar a frequência com que os sensores realizam as coletas. Em seguida, os módulos coletores publicam no *broker* MQTT o nome do módulo coletor, etiqueta temporal da coleta, unidade de medida e valor das leituras dos sensores. Exemplos dessas publicações podem ser observados na Figura 23.

A aplicação em névoa responsável pelo armazenamento dos dados foi desenvolvida em Python. Essa aplicação realiza a inscrição nos tópicos dos sensores por meio do protocolo MQTT para receber os dados coletados. Em seguida, os dados recebidos são tratados e armazenados no banco de dados em névoa. Ela recebe as publicações dos módulos coletores no *broker* MQTT e realiza o processamento necessário para extrair as informações relevantes, como o nome do módulo coletor, a etiqueta temporal, a unidade de medida e os valores das leituras dos sensores. Em seguida, esses dados são inseridos e armazenados no banco de dados para posterior consulta e análise.

Figura 23: Exemplo da estrutura de publicações MQTT realizadas pelo módulo de coleta.



Fonte: Autoria própria.

3.2.2 APLICAÇÃO DE ARMAZENAMENTO E MANIPULAÇÃO DOS DADOS

Para garantir a tolerância a falhas e permitir o acesso remoto do sistema, algumas funções presentes no subsistema de névoa também estão presentes na nuvem. A seguir, são descritas essas funções, indicando quando há uma versão correspondente na nuvem.

Visando possibilitar o funcionamento das funções de, armazenamento dos dados publicados pelos módulos coletores, apresentação no painel do meliponário, acionamento de alarme e sincronismo de bancos de dados em névoa e em nuvem, foi desenvolvida uma aplicação em Python, específicas para os subsistemas.

A primeira dessas aplicações tem a função de assinar os tópicos MQTT onde os módulos coletores publicam seus dados com QoS 1. Os dados publicados pelo módulo coletor no tópico, estão no seguinte formato: nome do módulo coletor; etiqueta temporal; unidade de medida 1; valor; unidade de medida 2; valor. Como exemplo, temos “caixa01;1681301010;C;22,1;UR;54.6”. A aplicação divide esses dados e utiliza-os para realizar duas inserções no PostgreSQL, com o índice, o identificador (ID) do módulo coletor, a etiqueta temporal, o ID da unidade de medida e o valor.

A segunda aplicação, responsável pela sincronização dos bancos de dados entre a névoa e a nuvem, é dividida em duas partes: a aplicação em névoa e a aplicação na nuvem. A aplicação em névoa assina um tópico MQTT no qual a nuvem publica o próximo índice a ser sincronizado. Se esse índice ainda não estiver presente

no banco de dados a aplicação em névoa aguarda. Quando o índice é encontrado no banco de dados, a névoa insere a compilação da inserção como uma linha no *broker* MQTT. A nuvem, por sua vez, assina o tópico que recebeu a publicação da névoa e realiza a inserção no banco de dados em nuvem os valores nos tópicos podem ser observados na Figura 24. As publicações e inscrições nos tópicos MQTT realizadas por essas aplicações são configuradas com o QoS 2. Essa escolha de QoS é feita para garantir uma comunicação confiável entre os dispositivos, especialmente no caso de erros de transmissão ou perda de pacotes. Essa abordagem permite que a aplicação Grafana na nuvem acesse os dados de forma remota, mesmo em cenários onde podem ocorrer falhas de transmissão ou perda de pacotes durante a comunicação MQTT.

Figura 24: Publicação MQTT com a comunicação entre névoa e nuvem.

Fonte: Autoria própria.

O Visor do Meliponário é um dispositivo eletrônico desenvolvido como parte do subsistema da névoa. O objetivo do Visor do Meliponário é fornecer informações visuais e em tempo real sobre o estado do meliponário. O *display* LCD exibe dados como temperatura e umidade das colmeias. O LED RGB é utilizado para indicar visualmente condições específicas, como níveis críticos de temperatura ou problemas identificados nas colmeias.

Ele é composto por um microcontrolador ESP D1 mini, um LED RGB e um display LCD de duas linhas e 16 colunas. Conforme mostrado na Figura 25, foram montados em uma caixa de passagem com dimensões de 8 cm x 8 cm x 5,5 cm. O dispositivo Visor do Meliponário é conectado à rede Wi-Fi e realiza a subscrição em

tópicos MQTT específicos para receber dados dos sensores, alerta de alarme e um tópico para ligar ou desligar os alarmes.

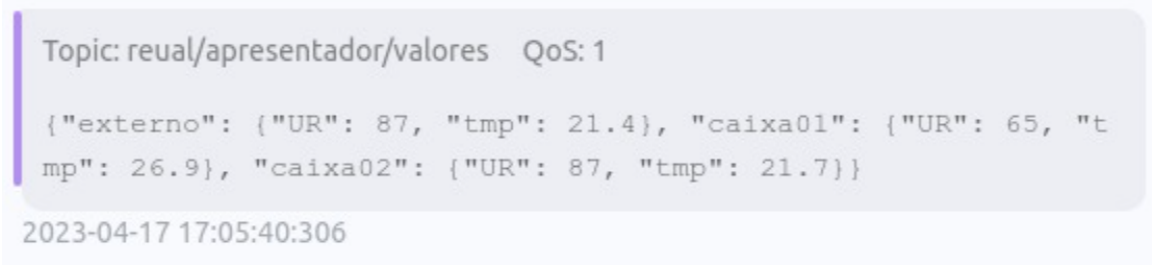
Figura 25: Protótipo do Visor do Meliponário.

Fonte: Autoria própria.

Quando o alarme está desligado, o dispositivo exibe as informações dos módulos coletores no display LCD e mantém o LED aceso na cor verde. Outra função realiza a verificação dos valores para determinar se estão dentro da faixa de normalidade ou se requerem intervenção do usuário. Caso sejam identificados valores que demandam atenção, essa função publica esses valores no tópico chamado “avisos”, além disso, ela altera o valor do tópico “alarme” de 0 para 1, indicando que o Visor do Meliponário deve apresentar as informações recebidas no tópico “avisos”, como exemplificado na Figura 26. Dessa forma, os usuários serão informados sobre os valores que necessitam de atenção ou ação. Quando recebe um comando para ligar o alarme, o Visor do Meliponário altera a cor do LED de verde para vermelho piscante, exibindo os dados que causaram o alarme.

Para enviar as informações ao Visor do Meliponário, foi desenvolvida uma aplicação em Python que realiza a leitura das últimas N linhas da tabela de dados armazenada no banco de dados. O valor de N é definido como o número de módulos coletores cadastrados, multiplicado pelo número de unidades de medidas cadastradas, e os dados são armazenados em um dicionário. Por exemplo,

Figura 26: Publicação MQTT enviando as informações para o Visor do Meliponário.



Fonte: Autoria própria.

considerando que os módulos coletores externo, caixa01 e caixa02 estão cadastrados, juntamente com as unidades de medida umidade relativa do ar e temperatura em graus Celsius, tem-se um valor de N igual a 6. Nesse caso, a aplicação lê os últimos 6 índices do banco de dados, que correspondem às leituras de temperatura e umidade dos três módulos coletores. Se um dos módulos coletores falhar e não publicar seus dados, a leitura é realizada considerando apenas os valores dos módulos coletores que estão ativos. O dicionário com os valores coletados é publicado em um tópico MQTT, assinado pelo Visor do Meliponário, que exibe as informações em tempo real. Essa abordagem permite uma visualização eficiente e atualizada dos dados coletados pelos módulos coletores, possibilitando a intervenção do criador ao identificar uma anomalia.

Em paralelo à função que alimenta as informações para o Visor do Meliponário, foi desenvolvida a função responsável por ativar os alarmes. Essa função verifica periodicamente os dados no banco de dados e analisa se os valores estão em um intervalo predefinido para cada tipo de unidade de medida. Para isso são desconsiderados os valores do módulo coletor que apresenta as informações externas, pois não há como agir diretamente sobre o clima.

3.3 APLICAÇÃO WEB

Os dados são exibidos na aplicação Web que utiliza a aplicação Grafana para gerar os gráficos. A aplicação Web foi instalada e configurada na Raspberry Pi para acesso em névoa, e também hospedada em nuvem. O banco de dados em nuvem possui os dados sincronizados com os dados armazenados no banco de dados em

névoa. Assim, o armazenamento de todas as coletas se torna mais seguro e há uma disponibilidade de espaço maior do que em um cartão de memória na Raspberry Pi. A aplicação em nuvem tem a possibilidade de receber os dados de sistemas instalados em locais físicos distantes, tendo a função de concentrar as consultas. Como exemplo, pode-se imaginar um criador com meliponários em mais de uma cidade.

Na camada de aplicação foram implementados serviços na Raspbery Pi, que executa o papel da névoa, também no servidor em nuvem. Nessa camada foram implementadas inicialmente as ferramentas Telegraf, InfluxDB e também a aplicação Grafana como aplicação Web responsável pela apresentação dos dados dos módulos coletores. No sistema final o Telegraf foi substituído por uma aplicação em Python e o InfluxDB substituído pelo PostgreSQL. Com os motivos apresentados posteriormente.

A aplicação Telegraf foi utilizada no início do projeto para fazer o registro dos dados que foram publicados no *broker* MQTT Mosquitto no banco de dados InfluxDB. O Telegraf é um agente voltado a *plugins* e pode ser configurado para coletar e enviar métricas e eventos de bancos de dados para sistemas IoT (APUROOP et al., 2021). O Telegraf assina os tópicos MQTT e, quando um dado é publicado no tópico MQTT, ele insere a informação no banco de dados.

O *InfluxDB* é um banco de dados de código aberto, não relacional, voltado a séries temporais, desenvolvido pela *InfluxDB Inc* em linguagem *Go*. Não possui dependências externas, sendo disponibilizado em um único arquivo binário. Utiliza o InfluxQL, uma linguagem de consulta personalizada do tipo *Structured Query Language* (SQL), com suporte a funções de agregação sobre dados de séries temporais (LEIGHTON et al., 2015; APUROOP et al., 2021).

As coletas de dados da fase inicial do trabalho foram começadas no dia 15 de fevereiro de 2022, mas em setembro de 2022 houve uma variação na rede elétrica com picos e surtos, que possivelmente corromperam o cartão SD que estava instalado na Raspberry Pi. O cartão SD, além de conter o sistema operacional e demais aplicações, armazenava os dados das coletas. Ao observar que as aplicações em névoa não respondiam mais, foi trocado o cartão SD e efetuada a nova implementação. Como ainda não existia a função de sincronismo do banco de dados em névoa e nuvem, o único local de armazenamento era o cartão SD.

O cartão danificado foi utilizado para tentativas de recuperação dos dados por aproximadamente um mês, obtendo sucesso ao criar uma imagem do cartão para um computador utilizando o comando DD do GNU/Linux. Esta imagem foi criada com a extensão de arquivo *img*. Esta imagem pôde ser aberta com um gerenciador de arquivos compactados chamado PeaZip (TANI, 2014), e com ele foi possível ignorar a estrutura corrompida do sistema operacional, extraindo as pastas de armazenamento de dados do InfluxDB. Em seguida, buscou-se formas de acessar os dados nos fóruns da Grafana Labs, mas os procedimentos encontrados não funcionavam, então foi dada continuidade às implementações faltantes no projeto. Porém, novas tentativas de acesso aos dados, realizadas em março de 2023, resultaram em sucesso para a recuperação dos dados da coleta inicial. Uma sugestão foi encontrada para excluir os índices e inserir os arquivos do banco de dados em um contêiner Docker com o InfluxDB em outra máquina. Os dados contidos nesses arquivos foram lidos por uma aplicação Python e posteriormente inseridos no banco de dados. Em seguida, esses dados foram mesclados com a coleta mais recente e exportados no formato *Comma Separated Values* (CSV) e armazenado em nuvem para garantir a sua integridade.

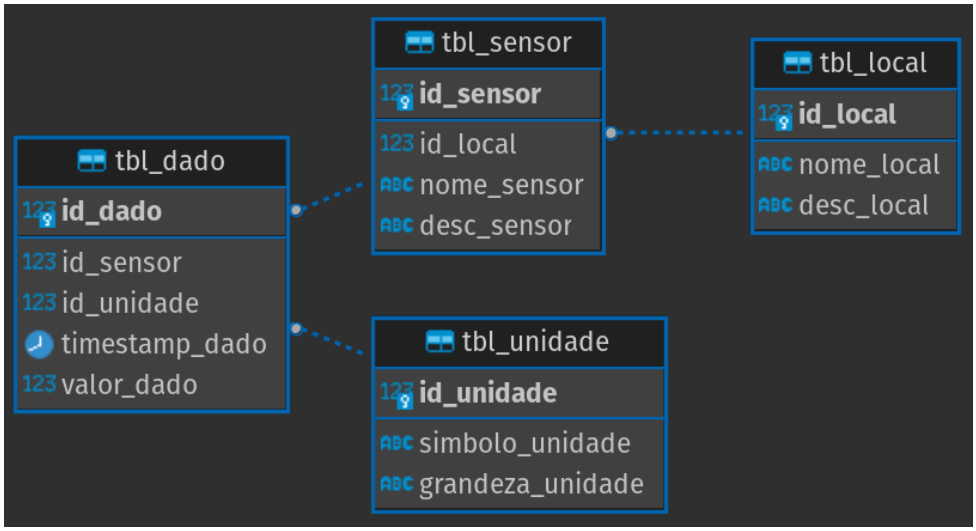
Após a implementação inicial verificou-se que o SGBD InfluxDB não possui funções nativas que tornassem possíveis sincronizar o banco de dados da névoa para a nuvem e descartar dados antigos armazenados na névoa. Ele apagaria nos dois lugares ou retornaria os dados da nuvem para a névoa. Outro ponto importante observado, foi que o Telegraf insere somente uma tupla com todas as informações publicadas em um tópico assinado do *broker* MQTT. Desta forma, não permite realização de normalização de dados, e com isso requer maior espaço de armazenamento. Além de que a aplicação Telegraf não permitiu de forma simples a interação com os índices dos dados. Por esses motivos optou-se por alterar o SGBD para PostgreSQL além do Telegraf ser substituído pela função da aplicação mencionada na Seção 3.2.2.

O PostgreSQL é um sistema de gerenciamento de banco de dados de código aberto, relacional e altamente extensível. É amplamente utilizado devido à sua confiabilidade, flexibilidade e recursos avançados. Desenvolvido pela comunidade *the PostgreSQL Global Development Group* desde 1996, mas suas origens são anteriores

a 1986, como parte do projeto PostgreSQL realizado na *University of California at Berkeley*. O PostgreSQL suporta transações ACID (Atomicidade, Consistência, Isolamento e Durabilidade) e oferece uma variedade de recursos avançados, como suporte a consultas complexas, índices avançados, integridade referencial, consultas complexas e replicação de dados. Com sua linguagem de consulta SQL completa, o PostgreSQL permite a manipulação eficiente e segura dos dados armazenados no banco de dados, atendendo às necessidades de aplicativos e sistemas de grande porte (GROUP, 2023).

Em março de 2023 a aplicação passou a armazenar os dados somente no SGBD PostgreSQL com a modelagem exposta na Figura 27, com um formato adequado tanto para a névoa quanto para a nuvem. São cadastrados o local, nome do módulo coletor e unidade de medida, além dos dados coletados por cada sensor. Cada dado coletado pelos módulos coletores é inserido em uma linha, armazenando um inteiro para referenciar o nome do módulo coletor e a unidade, reduzindo o espaço de armazenamento necessário tanto no cartão SD na Raspberry Pi quanto na nuvem.

Figura 27: Diagrama entidade relacionamento do banco de dados do sistema.



Fonte: Autoria própria.

O Grafana é um *software* de código aberto que é amplamente empregado para a exposição de painéis de visualização. Essa aplicação Web proporciona uma vasta gama de recursos destinados à visualização de dados, assim como admite a inclusão de *plugins*. O Grafana é uma aplicação Web que se dedica ao monitoramento

e à visualização de dados e eventos, podendo suportar múltiplas fontes de dados (GRAFANA LABS, 2022). Optou-se pela utilização do Grafana por ele ser um *software* de código aberto e por haver um conhecimento prévio da ferramenta.

Para obter acesso ao Grafana, é necessário utilizar um navegador e inserir o endereço IP (*Internet Protocol*) seguido da porta 3000. Vale ressaltar que essa é a porta padrão, mas pode ser alterada conforme a configuração específica. Após a instalação do Grafana, é necessário realizar a configuração inicial.

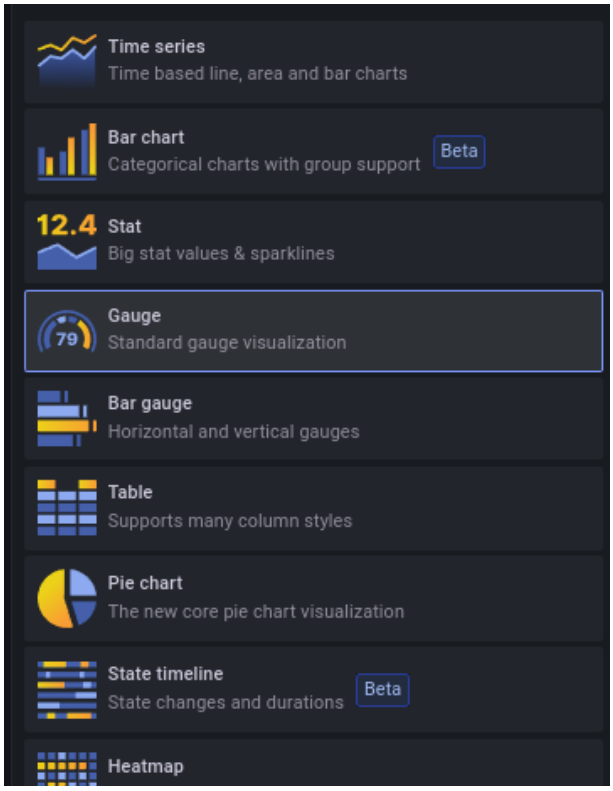
No contexto deste trabalho, a configuração da aplicação Web foi realizada seguindo uma sequência de passos descrita a seguir. Primeiramente, foi necessário adicionar uma fonte de dados. Nesse caso, o tipo de origem de dados escolhido foi o PostgreSQL. Para fazer isso, foram acessadas as configurações do Grafana e selecionado o tipo de origem de dados, informando o endereço e a porta do SGBD, bem como as configurações de autenticação e o nome da base de dados.

Após isso foi criado um *dashboard* e acessado para a adição dos painéis. Cada painel foi configurado para a exibição de um tipo de gráfico, exemplo de tipos de gráficos no Grafana são apresentados na Figura 28. Cada painel foi configurado com um título, tipo de gráfico e as consultas específicas ao banco de dados.

Nos painéis do tipo grau, para a apresentação da temperatura e umidade relativa do ar, foi inserido um gráfico por módulo coletor. A consulta ao banco de dados é realizada buscando o último dado inserido de cada sensor. Os limites de valores foram configurados para se o valor lido estiver fora da faixa normal, a cor de exibição do gráfico é alterada. A Figura 29 apresenta como esta configuração é realizada. No lado esquerdo superior é mostrado o exemplo do painel com os 3 mostradores de temperatura. No canto inferior esquerdo são visualizadas as consultas ao banco de dados utilizadas para criar cada um dos mostradores. No lado direito da figura estão as configurações, no inferior estão sendo exibidos os valores que a temperatura altera a cor de exibição. O painel de graus referente a umidade relativa do ar é configurado da mesma forma, alterando na consulta ao banco de dados o ID da unidade de 1 para 2.

A configuração dos painéis de gráficos de linha teve seu procedimento

Figura 28: Tipos de Gráficos no Grafana.

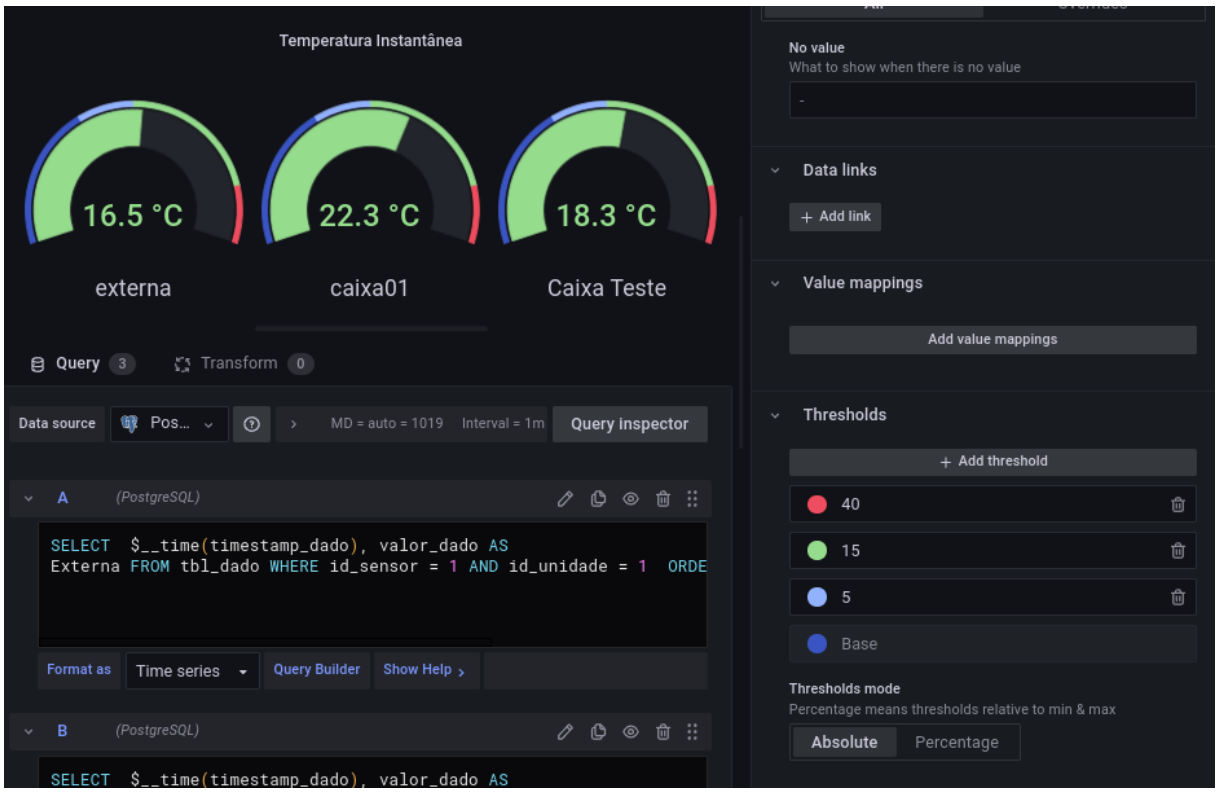


Fonte: Autoria própria.

semelhante aos painéis do tipo grau. A consulta ao banco de dados não define limites de leitura, pois o seu valor é alterado conforme o usuário altera o zoom. Essa configuração e organização dos painéis no Grafana permitiram uma visualização dos valores coletados pelos sensores, facilitando o monitoramento das informações relevantes ao sistema.

A configuração final resultou no *dashboard* apresentado na Figura 30. Na figura estão os 4 painéis configurados apresentando os valores coletados pelos sensores nas caixas de ASF. No *dashboard*, são exibidas informações de temperatura e umidade coletadas para os dispositivos denominados “externa”, caixa01 e caixa teste. É importante destacar que o painel no Grafana pode ser configurado para exibir nomes diferentes para os sensores. No caso mencionado, a representação “externa” refere-se aos dados de temperatura e umidade coletados no sensor posicionado externamente ao módulo coletor instalado na caixa01 e o nome do sensor é externo. Já a representação caixa01 apresenta os dados coletados internamente em uma caixa ocupada por abelhas MQQ. Além disso, o painel inclui os dados coletados pelo

Figura 29: Configurando a exibição do painel de temperatura na aplicação Web.



Fonte: Autoria própria.

sensor caixa02 representado no gráfico como caixa teste, que está posicionado em uma caixa de abelhas vazia, com um módulo ninho e o módulo coletor presentes. Os gráficos exibidos no painel representam os dados coletados pelos sensores dos módulos coletores, que estão localizados conforme explicado anteriormente.

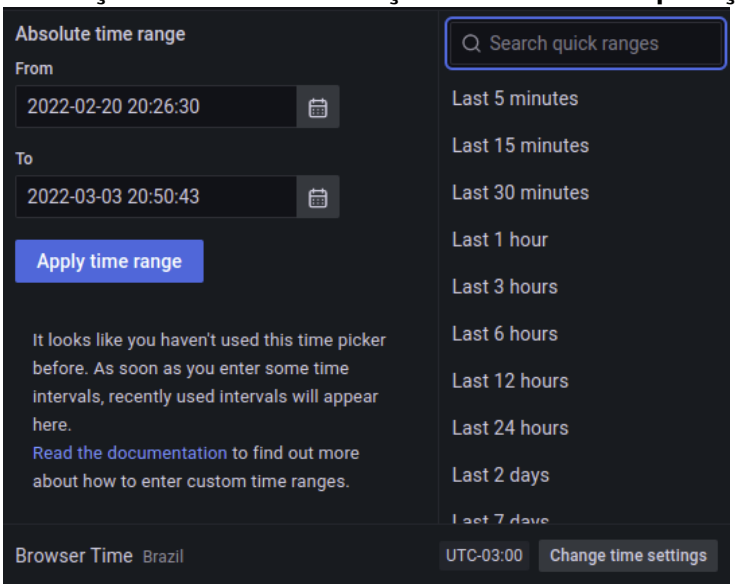
Figura 30: Painel do Grafana apresentando dados coletados e armazenados no banco de dados.

Fonte: Autoria própria.

Na aplicação Web resultante, pode-se selecionar o *zoom* para o período desejado conforme apresentado na Figura 31. Esta alteração permite que os gráficos apresentem os dados do período configurado. Esse período com os dados de temperatura e umidade relativa dos sensores são exibidos na Figura 32.

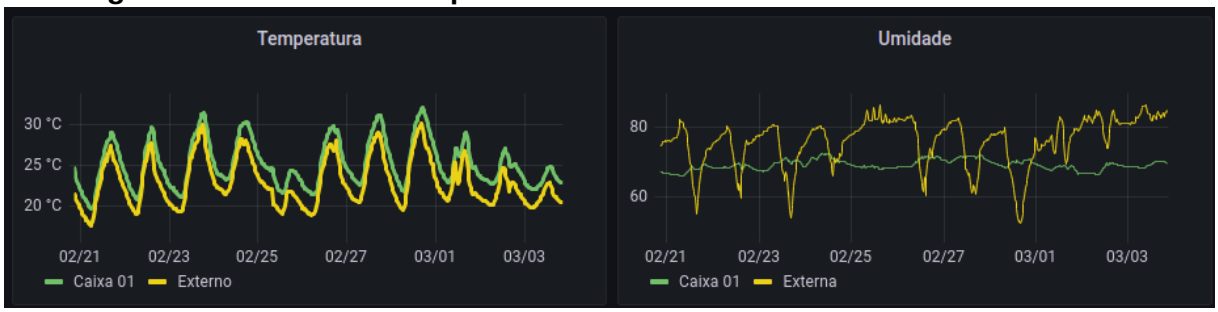
Para receber e distribuir a Internet no local de testes, foi utilizado um roteador TP-Link *Archer* C6 v2 com o *firmware* alterado para o OpenWRT. Optou-se pelo OpenWRT, por ser um *firmware* baseado em GNU/Linux que adiciona novas funcionalidades ao roteador. Como exemplo, IPv4, IPv6, *statefull firewall*, *Intruction Detection System*, rede 802.11s, 802.11r, *Virtual Private Network*, entre outras aplicações que podem ser instaladas. O projeto OpenWRT é desenvolvido por uma comunidade de voluntários (OPENWRT, 2022). Optou-se pelo *firmware* Openwrt por permitir um gerenciamento mais avançado da rede e ofereceu recursos adicionais para otimizar o acesso à Internet, proporcionando um ambiente estável e seguro para o sistema de testes.

Figura 31: Seleção de *zoom* de exibição dos dados na aplicação Web.



Fonte: Autoria própria.

Figura 32: Gráficos de temperatura e umidade entre 20/02/2022 e 03/03/2022.



Fonte: Autoria própria.

O arquivo de configuração da aplicação Web foi alterado para aceitar as conexões utilizando o protocolo *Hypertext Transfer Protocol Secure* (HTTPS) e não *Hypertext Transfer Protocol* (HTTP), para garantir as propriedades de segurança de confiabilidade, integridade e autenticação do servidor fornecidas pelo protocolo TLS, além de permitir proteção contra ataques de *Man-in-the-Middle* pelo uso dos certificados digitais (STALLINGS, 2020). Visando facilidade para o acesso da aplicação Web resultante, foi criado um nome de domínio no servidor *Domain Name System* (DNS) dinâmico Duck DNS. No roteador *OpenWRT* foi criado o direcionamento da porta TCP 3000 recebida, apontando para o endereço IP do Raspberry Pi. Na página do Duck DNS existem diversas instruções de implantação do serviço. Optou-se por utilizar A função do DuckDNS disponível no roteador com o *firmware OpenWRT*. Após realizadas as configurações, a aplicação Web ficou acessível com um certificado válido, no endereço `https://lauer77.duckdns.org:3000/`.

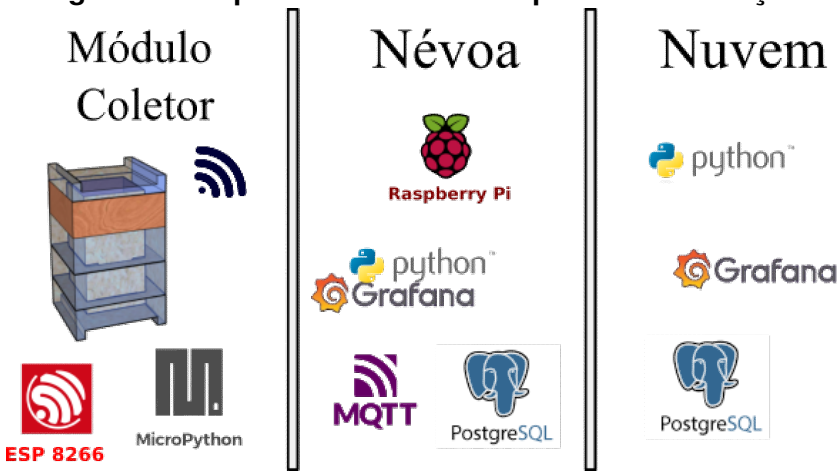
3.4 ARQUITETURA FINAL DO SISTEMA

O roteador Wi-Fi utilizado no início do projeto precisou ser substituído por um Xiaomi Mi Router AX3000, que possui um alcance maior com menos perda de pacotes que o TP-Link citado anteriormente. Em novembro de 2022 o *firmware OpenWRT* ainda não havia sido portado para o roteador Xiaomi Mi Router AX3000. Diante disto foi configurada a nova rede utilizando o *firmware* original do roteador. Desta forma foram perdidas funções como um *firewall* com capacidade de filtragem ou liberação de faixas de endereços IP, a atualização do Duck DNS que utiliza a porta 80, entre outras características e funções. Pelo novo roteador não possuir uma filtragem mais ampla de pacotes, a acessibilidade externa a aplicação Web com o gerador de certificado válido, precisou ser interrompida, pois se as portas *Transmission Control Protocol* (TCP) 80 e 3000 permanecessem liberadas, a conexão com a Internet era perdida em um período inferior a um dia. Essa limitação pode ser sanada em novas atualizações do *firmware* do roteador, assim, o sistema em névoa poderá ser liberado para acesso pela Internet novamente.

A arquitetura proposta inicialmente para o sistema de *hardware* e *software*

passou por modificações durante a implementação do projeto, relatadas nas seções anteriores. A Figura 33 apresenta a arquitetura do sistema após essas modificações, onde o módulo coletor está instalado em uma caixa de abelhas MQQ. No módulo coletor, há um microcontrolador ESP D1 mini com Micropython que coleta os dados dos sensores SHTC3. Esses dados são enviados por Wi-Fi para o *broker* MQTT implementado em uma Raspberry Pi. A Raspberry Pi, além de executar o *broker* MQTT, também executa aplicações de armazenamento dos dados, que recebe as informações dos tópicos MQTT e as insere no banco de dados. Essa aplicação gerencia um índice sequencial, para evitar transações não concluídas e linhas nulas. Optou-se por gerenciar os índices com a aplicação ao invés de utilizar um campo de incremento automático do banco de dados para facilitar o sincronismo entre névoa e nuvem. Os dados coletados são acessíveis por meio da aplicação Web.

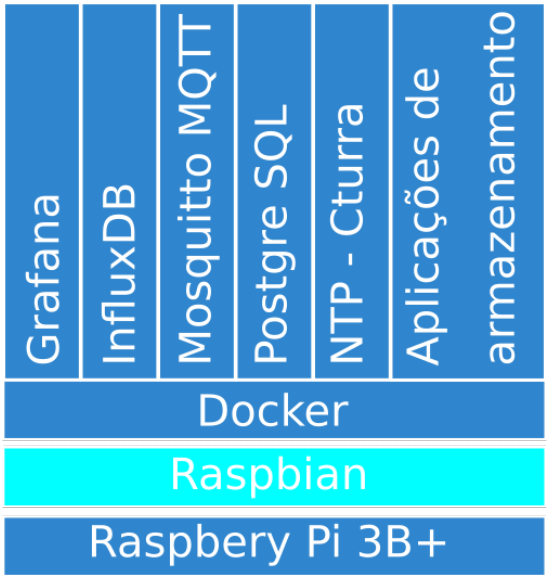
Figura 33: Arquitetura do sistema após as modificações.



Fonte: Autoria própria.

Tanto na nuvem, quanto na Raspberry Pi, que atua como dispositivo de névoa, os serviços e aplicações foram implementados utilizando o Docker Contêiner. Na Figura 34, é possível visualizar os contêineres empregados na Raspberry Pi. Na nuvem o funcionamento é semelhante, substituindo-se a Raspberry Pi por uma máquina virtual com o sistema Debian 11.

Figura 34: Contêineres empregados na Raspberry Pi.



Fonte: Autoria própria.

3.5 EXPERIMENTOS CONTROLADOS

Com o objetivo de avaliar o funcionamento do sistema de coleta e armazenamento de dados proposto, foram realizados experimentos em um ambiente controlado. Esses experimentos permitiram aferir a qualidade dos dados coletados pelos sensores com a exibição dos dados no Visor do Meliponário, o acionamento de alarmes e a apresentação do dado que requer atenção, bem como avaliar a eficácia do software desenvolvido para a comunicação e armazenamento desses dados. Nesta seção, são apresentados os detalhes dos experimentos realizados, incluindo a descrição do ambiente experimental, dos procedimentos de coleta de dados, dos testes de gatilhos, da exibição no Visor do Meliponário e do sincronismo dos dados em névoa e em nuvem.

Para a execução dos experimentos de validação foi construída uma nova caixa e adicionado um novo módulo coletor. Essa caixa não possui abelhas e é apresentada como caixa teste nos gráficos da aplicação Web. Na caixa teste, são coletados os dados da mesma forma que estão sendo coletados nos casos dos sensores externo e caixa01.

Inicialmente, para validar a comunicação entre o sensor e o subsistema de

névoa, o Raspberry Pi foi desconectado da rede por um período e ao reconectar foram verificados o funcionamento do QoS 1 implementado no sensor e se esses dados foram publicados com o carimbo de hora corretamente.

Uma vez que a comunicação ocorre corretamente entre os sensores e a aplicação responsável pelo armazenamento dos dados, foi executado o experimento para verificar o armazenamento correto desses dados no banco de dados. Essa verificação considerou também a conferência dos dados e de data e hora informada pelo sensor no tópico MQTT com a tupla inserida no banco de dados. Também foi executado o teste de desconexão e reconexão da rede da Raspberry Pi, para efetivar o funcionamento do QoS 1 do MQTT.

Visando confirmar a efetividade da sincronização de dados entre o banco de dados em nuvem e o banco de dados em névoa, o contexto do próximo experimento motivou-se pela necessidade de manter a base de dados atualizada em ambos os ambientes, envolvendo a sincronização dos dados entre os bancos de dados. A avaliação do sistema de sincronismo da base de dados em ambiente de névoa e nuvem consistiu na comparação dos índices inseridos em cada local de armazenamento e o campo de data e hora armazenados em névoa e em nuvem. Adicionalmente, foram testadas as aplicações de sincronismo desenvolvidas, avaliando se ao interromper e reiniciar a aplicação em névoa e nuvem, haveria correta sincronização dos dados após os dois sistemas estarem ativos. Dessa forma, é possível verificar se o sistema realmente é resiliente para cenários que envolvem a perda de conectividade temporária.

Além disso, para o monitoramento de recursos da Raspberry Pi 3B+ durante a operação como névoa de dados, uma aplicação foi instalada para gerar gráficos que mostram a utilização dos recursos computacionais da placa, além de verificar a temperatura de funcionamento. Com base nessas informações, foi possível concluir a viabilidade do uso da Raspberry Pi 3B+ para executar as funções de nuvem no sistema.

Também foram realizados testes para confirmar se os dados extraídos do banco de dados correspondiam aos exibidos nos painéis da aplicação Web, verificando-se sua precisão e atualização em tempo real. Essas características são

importantes porque a identificação das anomalias também depende que os dados armazenados nos banco de dados em névoa sejam apresentados de forma clara e atualizada ao usuário.

Em seguida, foi verificado o comportamento do sistema em cenários de temperatura e/ou umidade crítica em uma caixa. Foi forçada a alteração de temperatura e umidade no interior da caixa teste, primeiramente apontando o fluxo de ar de um secador de cabelos, desta forma aumentando a temperatura e reduzindo a umidade relativa do ar no sensor caixa teste. Com isso, a aplicação responsável pelo gerenciamento dos alarmes, que faz a análise das faixas normais de temperatura e umidade, deve publicar nos tópicos de avisos e ativação do alarme. Com esta ação pode ser testado se o Visor do Meliponário desempenha corretamente a função de piscar o LED em vermelho e exibir, no *display* LCD, qual sensor e qual unidade de medida está fora do valor normal. Da mesma forma é testado se ao estabilizarem a temperatura e a umidade, o LED altera sua cor para verde e se os valores lidos em todos os sensores são apresentados no *display* LCD. No painel da aplicação Web, o mostrador deve apresentar o valor em cores diferentes de verde, apresentada quando a informação está em faixa normal de operação.

Quanto à geração e disponibilização do *dataset*, durante o processo de desenvolvimento do sistema, foram coletados dados que foram inicialmente armazenados no InfluxDB e, posteriormente, no PostgreSQL. Esses dados foram coletados para a caixa que possui o módulo coletor com os conjuntos de sensores externo e caixa01 e com as abelhas MQQ, sendo dados reais que permitem observar o comportamento da colmeia em um período de 388 dias. Esses dados foram compilados em um *dataset* para ser disponibilizado publicamente.

Este capítulo apresentou as etapas envolvidas no desenvolvimento do sistema, abrangendo a construção do módulo coletor, bem como a utilização de dispositivos de *hardware* e *software* para estabelecer a comunicação entre os sensores e o ambiente de névoa. Além disso, foram abordados os softwares empregados na comunicação entre a névoa e a nuvem. No capítulo subsequente, serão discutidos os resultados decorrentes dos experimentos realizados.

4 RESULTADOS

Neste capítulo serão apresentados os resultados obtidos com os experimentos executados, apontando particularidades e benefícios do uso realizados para validar o sistema de monitoramento de colmeias de abelhas utilizando a computação em névoa e em nuvem. Além do resultado, serão discutidas as particularidades e objetivos de cada experimento.

4.1 COMUNICAÇÃO ENTRE SENSOR E NÉVOA

O experimento da comunicação entre sensor e névoa pretende validar a publicação dos dados coletados pelos sensores e seu redirecionamento pela inscrição feita pela aplicação responsável pelo armazenamento dos dados com o *broker* MQTT. Para realizar este experimento, o período de leituras dos sensores foi alterado de 60 segundos para 20 segundos. A Raspberry Pi foi conectada em um monitor e teclado, para possibilitar a visualização do terminal, pois durante o experimento, o cabo Ethernet será desconectado impossibilitando o acesso por terminal remoto. Com acesso ao terminal do sistema operacional Raspbian, foi realizada a conexão no PostgreSQL e executada uma consulta de seleção das 15 últimas linhas inseridas na tabela *tbl-dado*. Após repetir a execução e confirmado o funcionamento, foi retirado o cabo Ethernet, que conecta o dispositivo à rede, por 70 segundos.

No exemplo mencionado, o valor *keepalive* estava configurado para 60 segundos. Isso significa que o cliente precisa enviar um pacote MQTT PINGREQ, para o *broker* ao menos uma vez a cada 60 segundos para manter a conexão ativa. Se o *broker* não receber esse pacote nesse intervalo de tempo, ele considera que o cliente está inativo ou desconectado.

Para realizar o experimento, simulando uma perda momentânea de conexão, o cabo Ethernet da Raspberry Pi foi desconectado por 35 segundos e, após este tempo, foi reconectado. O módulo coletor estava configurado com o *keepalive* no valor de 60 segundos, além de, realizar a leitura e publicação dos dados no período de 20 segundos. Conforme pode ser observado na Figura 35, o sensor com ID 2, inseriu os dados que ficaram na fila, em contrapartida, o *broker* mostrou em seus registros a perda de conexão por *keepalive* extrapolado com o sensor com ID 1, e os dados não foram coletados até o sensor reestabelecer a conexão com o *broker*.

Figura 35: Dados inseridos no banco de dados após a desconexão e reconexão de 35 segundos.

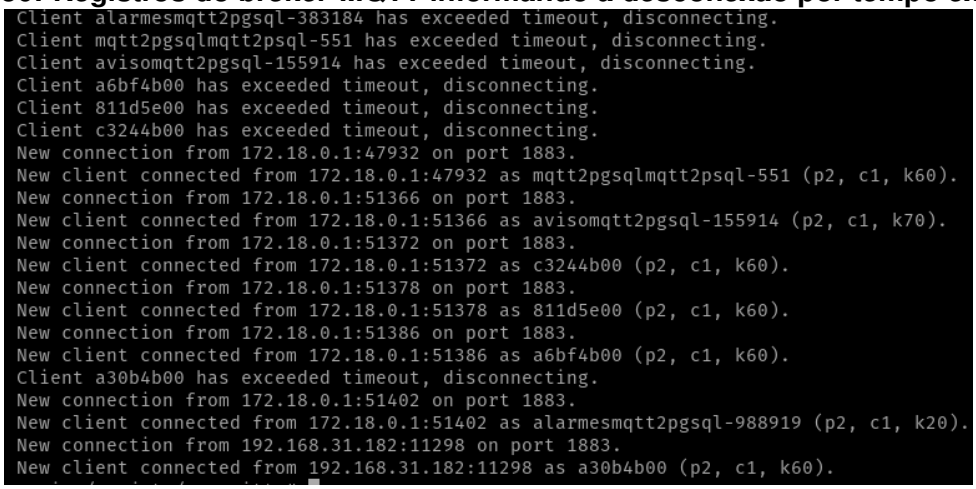
```
MQTTbee=# SELECT * from tbl_dado ORDER BY id_dado DESC limit 13;
 id_dado | id_sensor | id_unidade |       timestamp_dado       | valor_dado
---------+-----------+------------+----------------------------+------------
 1029934 |         2 |          2 | 2023-05-15 20:24:52+00     |         58
 1029933 |         2 |          1 | 2023-05-15 20:24:52+00     |         29
 1029932 |         2 |          2 | 2023-05-15 20:24:31+00     |         58
 1029931 |         2 |          1 | 2023-05-15 20:24:31+00     |         29
 1029930 |         1 |          2 | 2023-05-15 20:24:17+00     |         44
 1029929 |         1 |          1 | 2023-05-15 20:24:17+00     |         23
 1029928 |         2 |          2 | 2023-05-15 20:24:11+00     |         58
 1029927 |         2 |          1 | 2023-05-15 20:24:11+00     |         29
 1029926 |         1 |          2 | 2023-05-15 20:23:57+00     |         43
 1029925 |         1 |          1 | 2023-05-15 20:23:57+00     |         23
 1029924 |         2 |          2 | 2023-05-15 20:23:51+00     |         58
 1029923 |         2 |          1 | 2023-05-15 20:23:51+00     |         29
 1029922 |         1 |          2 | 2023-05-15 20:23:36+00     |         45

(13 rows)
```

Fonte: Autoria própria.

Quando o cabo Ethernet é desconectado por 70 segundos, os clientes MQTT não conseguem enviar o pacote PINGREQ para o *broker* no tempo de *keepalive*. Como resultado, o *broker* MQTT reconhece que os clientes excedem o tempo e os desconecta. Este evento pode ser observado na Figura 36, que apresenta as informações do arquivo de registros do *broker* MQTT.

O módulo coletor, funcionando como cliente MQTT, foi configurado para tentar se reconectar automaticamente ao *broker*. Dessa forma, quando o cabo Ethernet foi recolocado e a conexão foi restabelecida, os clientes MQTT puderam restabelecer a comunicação com o *broker* MQTT. Como a conexão entre o módulo coletor e o *broker* acontece com o QoS 1 com *keepalive* de 60 segundos, houve a extrapolação do tempo de *keepalive*, consequentemente, não há o enfileiramento dos dados coletados durante o período de desconexão. A Figura 37 apresenta esta descontinuidade nos

Figura 36: Registros do *broker* MQTT informando a desconexão por tempo excedido.



Fonte: Autoria própria.

dados em decorrência da perda de conexão, que ocorreu entre o ID do dado 1.029.912 e o 1.029.913. Nas referidas linhas observou-se 74 segundos de diferença entre um dado e outro, se estivesse enfileirado com o QoS 1 teria sido mantido o tempo de 20 segundos entre as duas inserções de cada sensor.

Figura 37: Dados inseridos no banco de dados após a desconexão e reconexão de 70 segundos.

id_dado	id_sensor	id_unidade	timestamp_dado	valor_dado
1.029.909	2	1	2023-05-15 20:21:48.000	28,700000763
1.029.910	2	2	2023-05-15 20:21:48.000	57,900001526
1.029.911	1	1	2023-05-15 20:21:51.000	22,600000381
1.029.912	1	2	2023-05-15 20:21:51.000	44,599998474
1.029.913	3	1	2023-05-15 20:23:05.000	25,600000381
1.029.914	3	2	2023-05-15 20:23:05.000	56
1.029.915	2	1	2023-05-15 20:23:10.000	28,600000381
1.029.916	2	2	2023-05-15 20:23:10.000	57,900001526
1.029.917	1	1	2023-05-15 20:23:16.000	22,600000381
1.029.918	1	2	2023-05-15 20:23:16.000	44,5

Fonte: Autoria própria.

Durante o experimento, pode ser verificado que a conexão via Wi-Fi aliada ao protocolo MQTT para a comunicação dos módulos coletores funciona corretamente durante interrupções momentâneas da conexão. Além da comprovação que ao desconectar o cliente MQTT por um período maior que o valor definido para o *keepalive*, ao iniciar a conexão com o *broker*, as leituras dos sensores não foram registradas. Foram realizados testes com o *keepalive* configurado em 60 segundos, com desconexões de 70 segundos e 35 segundos. Esse experimento evidenciou a

importância de ajustar o intervalo de *keepalive* conforme o período de coleta, visando garantir que o QoS 1 minimize a perda de dados coletados em caso de falha na conexão. No entanto, é necessário conduzir estudos sobre o custo computacional do uso de intervalos de *keepalive* superiores a 60 segundos no protocolo MQTT, levando em consideração o número de conexões simultâneas.

4.2 INSERÇÃO DOS DADOS DOS SENSORES NO BANCO DE DADOS

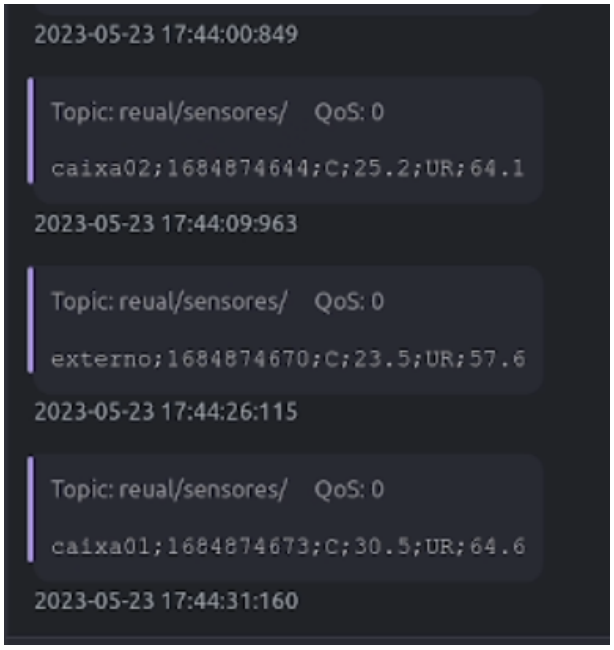
O experimento de inserção dos dados no banco de dados foi projetado para verificar se os dados recebidos através do protocolo MQTT são corretamente transcritos e armazenados no banco de dados. Os módulos de coleta estavam com a configuração de período de leituras a cada 20 segundos. Esse experimento foi acompanhado remotamente a partir de um microcomputador rodando o sistema operacional GNU/Linux, com a Distribuição Pop OS 22.04, executando a aplicação cliente MQTT chamada MQTTX, utilizada para estabelecer a conexão com *broker* MQTT executado no subsistema de névoa onde o módulo coletor publica suas coletas. Além disso, foi utilizado o software cliente de banco de dados chamado DBeaver para visualizar de forma amigável os resultados da consulta ao SGBD PostgreSQL executado no subsistema em nuvem. O experimento foi documentado em vídeo, disponibilizado em `https://youtu.be/C7blHWC8kKg`, fornecendo uma compreensão mais detalhada de todo o processo.

Esse experimento foi executado para confirmar o funcionamento correto da aplicação que assina os tópicos MQTT “reual/sensores”, onde os sensores publicam suas leituras, no formato “Nome do sensor;timestamp;unidade de medida 1; valor 1; unidade de medida 2, valor 2, unidade de medida N, valor N”. Posteriormente a aplicação divide os dados do sensor por unidade de medida aferida, realizando a inserção dos dados no banco de dados, sendo cada linha composta pelo ID do sensor, ID da unidade de medida lida, *timestamp* da leitura e o valor lido. Por serem 2 unidades de medida lidas em 3 sensores, totalizam-se 6 inserções no banco de dados a cada 20 segundos.

A aplicação MQTTx se inscreve no tópico MQTT “reual/sensores”, para que

seja possível observar os dados. No cliente DBeaver, uma consulta foi configurada para exibir as últimas 6 inserções, sendo deixada pronta para execução. Quando ocorrem três publicações consecutivas no tópico MQTT, conforme ilustrado na Figura 38, a consulta é manualmente executada e o MQTTx é desconectado para se observar os últimos dados apresentados na tela.

Figura 38: Três publicações MQTT registradas na aplicação MQTTx para o experimento de inserção dos dados no banco de dados.

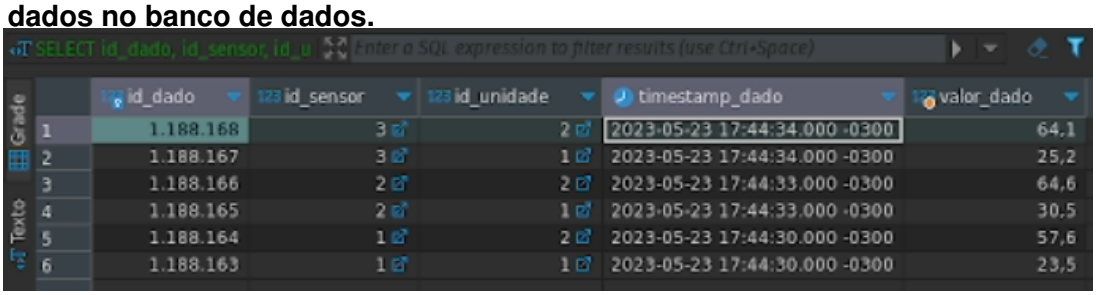


Fonte: Autoria própria.

Para compreender os dados é necessário associar o ID do sensor aos seus respectivos nomes, onde o sensor com ID 1 representa o sensor externo, o sensor com ID 2 representa a caixa01 e o sensor com ID 3 representa a caixa02. Além disso, o ID da unidade lida permite identificar as grandezas medidas, sendo o ID 1 correspondente à temperatura em graus Celsius e o ID 2 à umidade relativa do ar em porcentagem. A consulta realizada no banco de dados, como mostrado na Figura 39, apresenta os resultados dessas associações, fornecendo uma visão clara e compreensível dos dados coletados. Essa validação é fundamental para garantir a precisão e a consistência das informações registradas no banco de dados durante o experimento.

O experimento de inserção dos dados no banco de dados por meio do protocolo MQTT foi realizado com sucesso. Foi possível estabelecer a conexão com o

Figura 39: Resultado da consulta de 6 linhas no DBeaver para o experimento de inserção dos dados no banco de dados.



Fonte: Autoria própria.

broker MQTT e visualizar os resultados da consulta ao banco de dados PostgreSQL. A aplicação desenvolvida demonstrou sua capacidade de capturar, dividir e inserir corretamente os dados dos sensores no banco de dados, o que validou a eficiência do sistema. Esse experimento evidenciou a viabilidade e a eficácia do uso do protocolo MQTT na coleta e armazenamento de dados provenientes de sensores em um ambiente de névoa e nuvem.

4.3 COMUNICAÇÃO ENTRE NÉVOA E NUVEM

Este experimento foi elaborado com o propósito de comprovar o funcionamento das funções de comunicação e sincronismo dos dados entre névoa e nuvem. Com o intuito de permitir uma visualização clara das informações trocadas entre os subsistemas. Para simular a queda na conexão, foram interrompidas as aplicações responsáveis pelo sincronismo durante um período. Utilizando aplicações em névoa e nuvem que exibem os IDs dos últimos dados inseridos em cada banco de dados, foi possível comparar a eficiência das aplicações de sincronismo. O experimento foi acompanhado por meio de terminais SSH remotos, foi gravado e disponibilizado em `https://youtu.be/lRYGtbTWFWU`.

A visualização do experimento foi realizada a partir de um microcomputador com sistema operacional GNU/Linux, especificamente a Distribuição Pop OS 22.04. O microcomputador estava conectado utilizando conexões de terminais SSH, aos subsistemas de névoa e nuvem, para permitir a interação e o monitoramento dos dados. Durante o experimento, foram estabelecidas conexões de terminais de

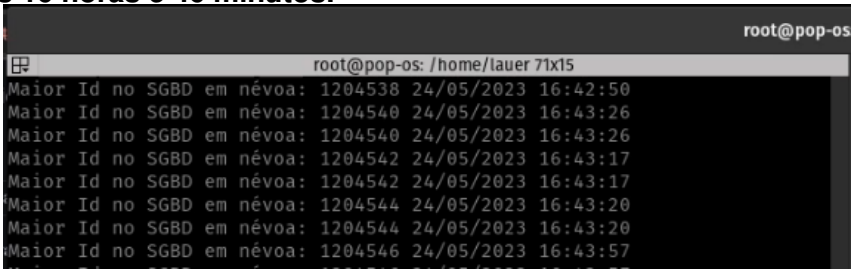
acesso remoto utilizando o protocolo *Secure Socket Shell* (SSH) para acesso remoto e também foram realizadas conexões aos bancos de dados por meio do cliente DBeaver. Essas conexões desempenharam um papel importante na configuração, monitoramento e análise dos dados coletados durante o experimento.

Foi criada uma aplicação na linguagem Python para informar a cada 5 segundos no terminal remoto o ID e *timestamp* do último dado inserido no banco de dados. No caso específico para a aplicação executada no servidor na nuvem, também apresenta em tela o seu tempo de execução e o número de índices inseridos durante a execução. As informações do *timestamp* foram convertidos para GMT -3, hora de Brasília, para facilitar sua compreensão.

No início do experimento, as aplicações de sincronismo que funcionam em névoa e em nuvem foram interrompidas em 22 de maio de 2023, às 7 horas e 10 minutos. Durante esse período, os módulos coletores continuaram enviando os dados para o subsistema de névoa, mantendo a atualização do banco de dados nesse ambiente. No entanto, os novos valores deixaram de ser recebidos e registrados no banco de dados da nuvem.

O experimento foi retomado no dia 24/05/2023 às 16 horas e 46 minutos, quando foram reiniciadas as aplicações de sincronismo implementadas em contêineres Docker. É possível observar na Figura 40 os valores dos últimos IDs armazenados em névoa e na Figura 41 os armazenados em nuvem. Nesse momento, o último ID inserido no banco de dados em névoa era o 1.204.568 e o último ID inserido no banco de dados na nuvem era o 1.157.816, totalizando uma diferença de 46.752.

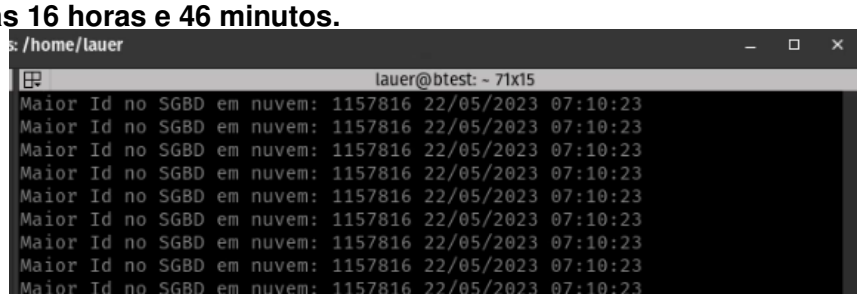
Figura 40: Maiores IDs dos dados inseridos no banco de dados em névoa no dia 24/05/2023 às 16 horas e 46 minutos.



Fonte: Autoria própria.

Durante o experimento, a aplicação que apresentava as informações da

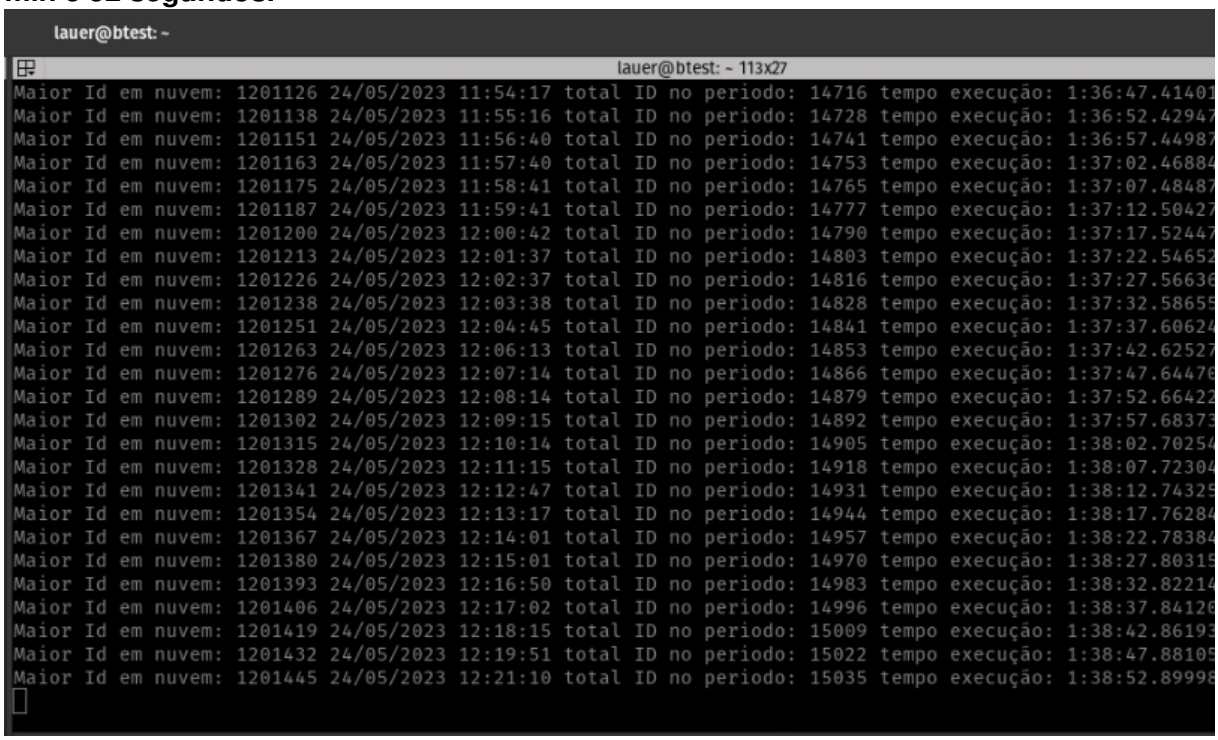
Figura 41: Maiores IDs dos dados inseridos no banco de dados em nuvem no dia 24/05/2023 às 16 horas e 46 minutos.



Fonte: Autoria própria.

nuvem, foi editada, incluindo o número de IDs inseridos desde o início da aplicação e o tempo que já estava em execução. Com essa informação em mãos, pôde-se calcular o número de linhas de informações transferidas em média a cada minuto. Com os dados presentes na Figura 42. Nessa figura, na última linha está a informação que em 01:38:52 h, foram inseridos 15035 IDs, portanto, em média foram inseridos 172,3 IDs por minuto, ou 10340,9 por hora.

Figura 42: Número de IDs inseridos no banco de dados em nuvem durante 1 hora, 37 min e 52 segundos.

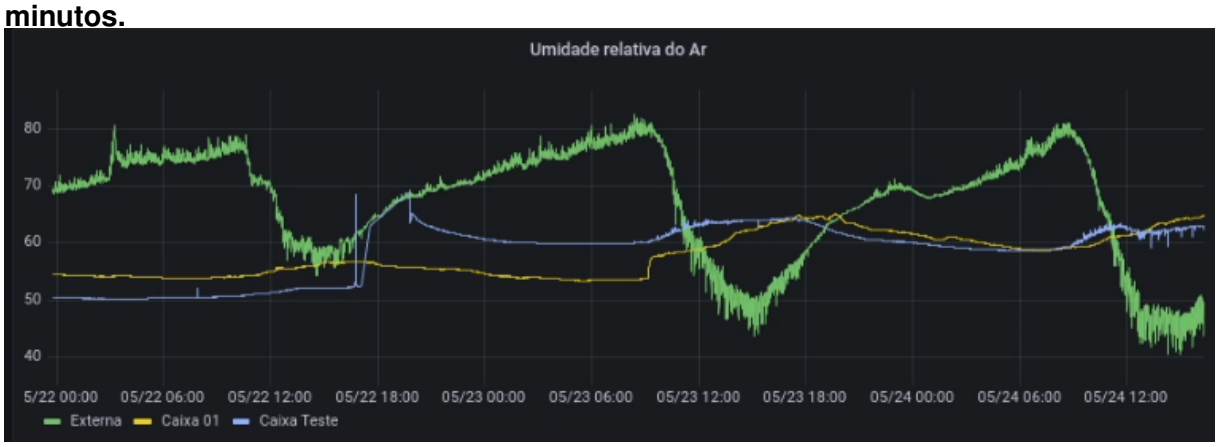


Fonte: Autoria própria.

Outra forma de observar a diferença dos dados presentes no banco de dados

de cada sistema é pelos próprios gráficos da aplicação Web. A Figura 43 mostra o gráfico da aplicação Web que representa a umidade relativa do armazenados no banco de dados em névoa. Já a Figura 44 apresenta o gráfico ao mesmo momento, mas exibindo os dados armazenados em nuvem, com funções de sincronismo ainda desativadas. Pode-se comparar a diferença entre os dados presentes em cada local de armazenamento, pois a visualização desses dados foram configuradas para um período específico, iniciando à meia-noite do dia 22/05/2023 e finalizando às 16 horas e 46 minutos do dia 24/04/2023. À medida que o processo de sincronismo avançava, o gráfico exibido como névoa mostrava as informações mais atualizadas, ou seja, as linhas correspondentes aos dados mais recentes. Por isso, é possível observar que ele possui dados atualizados, enquanto o gráfico da nuvem continuou exibindo dados apenas até o dia 22/05/2023.

Figura 43: Gráficos de umidade relativa do ar em névoa 24/05/2023 às 16 horas e 46 minutos.



Fonte: Autoria própria.

Foi verificado na manhã do dia 25, que os dados estavam sincronizados, e os painéis de temperatura e umidade relativa do ar na aplicação Web já possuíam os índices correspondentes. Esses painéis são apresentados na Figura 45, sendo exibidos os gráficos de temperatura e umidade relativa dos dados armazenados em névoa, e na Figura 46, com gráficos de temperatura e umidade relativa dos dados em nuvem.

Além das imagens que apresentam os dados nos painéis, é possível verificar os IDs das linhas de dados correspondentes entre a névoa e a nuvem nas Figuras

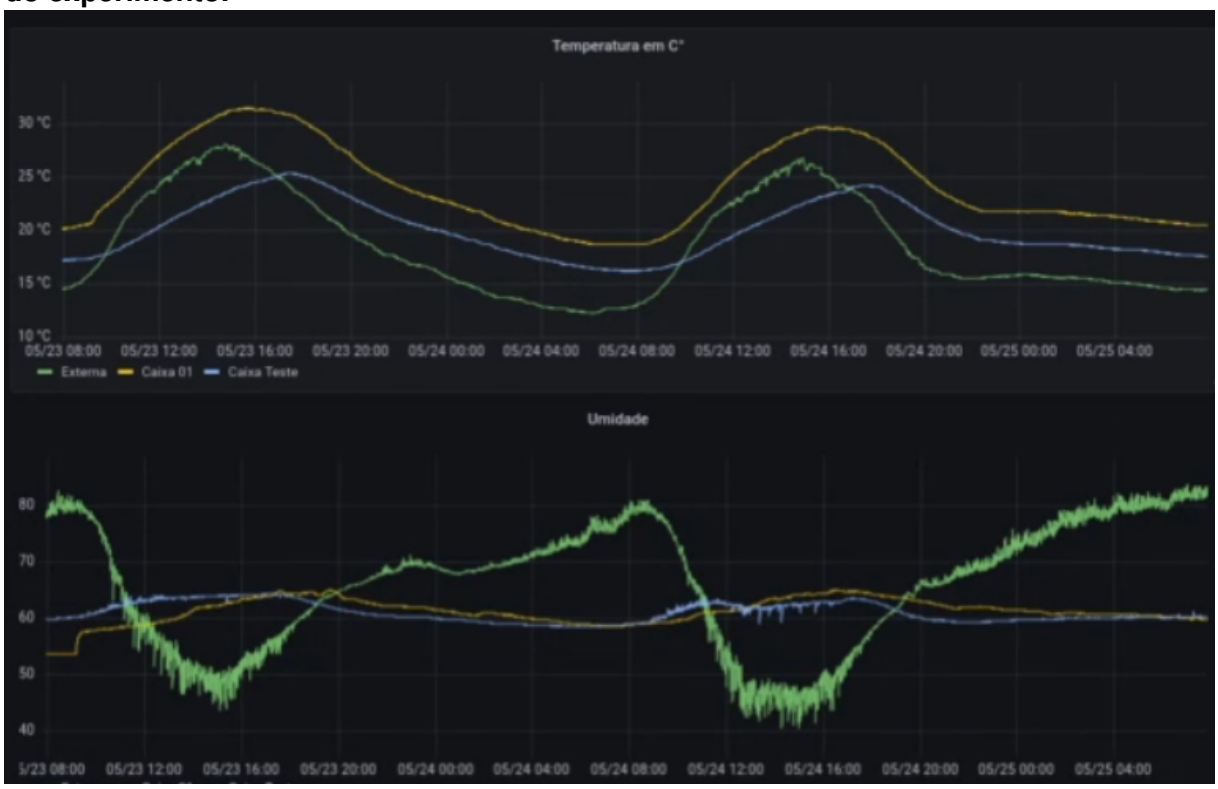
Figura 44: Gráficos de umidade relativa do ar em nuvem 24/05/2023 às 16 horas e 46 minutos.

Fonte: Autoria própria.

Figura 45: Gráficos de umidade relativa do ar e temperatura exibidos em névoa ao final do experimento.

Fonte: Autoria própria.

Figura 46: Gráficos de umidade relativa do ar e temperatura exibidos em nuvem ao final do experimento.



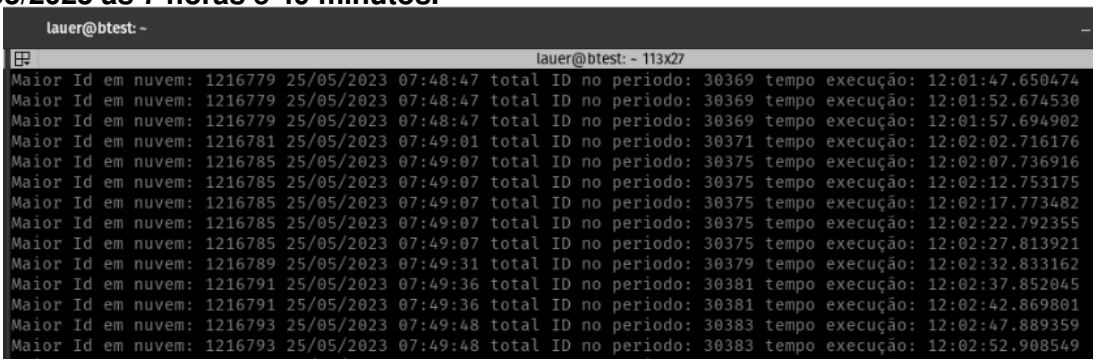
Fonte: Autoria própria.

47 e 48. Essas figuras ilustram os terminais remotos executando as aplicações nos ambientes de névoa e nuvem, permitindo visualizar o processo de sincronização dos dados. Através dessas figuras, é possível obter uma visão clara das informações comparadas entre os dois ambientes, exibindo a consistência e a integridade dos dados entre a névoa e a nuvem.

Figura 47: Maiores IDs dos dados inseridos no banco de dados em névoa no dia 25/05/2023 às 7 horas e 49 minutos.

Fonte: Autoria própria.

Figura 48: Maiores IDs dos dados inseridos no banco de dados em nuvem no dia 25/05/2023 às 7 horas e 49 minutos.



Fonte: Autoria própria.

Essas comparações destacam o bom funcionamento do sistema responsável pelo sincronismo entre o banco de dados armazenado na névoa e o banco de dados armazenado na nuvem. A interrupção temporária das aplicações de sincronismo permitiu observar que os dados continuaram sendo atualizados e registrados corretamente no banco de dados da névoa durante esse período. Ao retomar o sincronismo, as comparações entre os IDs das linhas de dados nas imagens demonstraram a consistência entre os dois ambientes. Isso reforça a confiabilidade

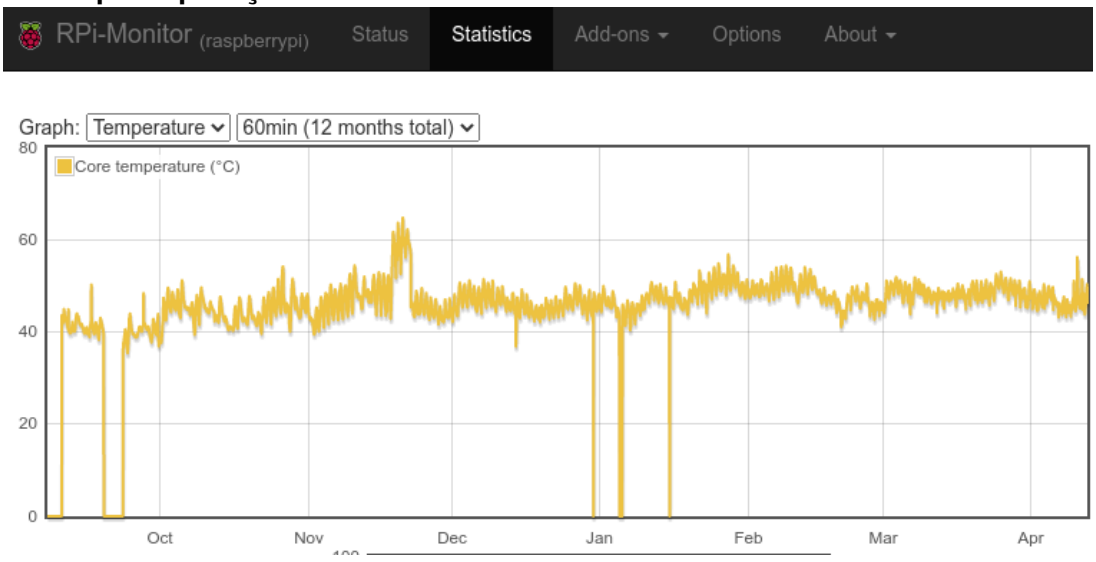
do sistema de sincronismo, garantindo que as informações sejam devidamente transmitidas e mantidas atualizadas tanto na névoa quanto na nuvem.

4.4 MONITORAMENTO DE RECURSOS DA RASPBERRY PI 3B+

Esse experimento foi elaborado visando testar a capacidade do modelo Raspberry Pi 3B+ em executar as tarefas designadas para o subsistema de névoa no projeto, descritas na Seção 3.2. Com o intuito de verificar se o referido modelo possuiu requisitos de *hardware* suficiente para executar de maneira satisfatória as tarefas requeridas, foi instalada na Raspberry Pi a aplicação RPi-Monitor (BERGER, 2014).

Utilizou-se a aplicação no período de setembro de 2022 a abril de 2023 para monitorar a utilização de recursos da Raspberry Pi 3B+. A Figura 49 apresenta a variação da temperatura do núcleo do processador durante esse período, que oscilou entre 35 ℃ e 65 ℃, chegando a zero quando a Raspberry Pi esteve desligada. Segundo Gamess e Hernandez (2022), a temperatura do núcleo é tolerada ao estar abaixo de 80 ℃, logo, os valores observados estão nos limites aceitáveis para o referido modelo. É importante ressaltar que ao longo do experimento não foi utilizada ventilação forçada para resfriar o processador.

Figura 49: Gráfico de temperatura do núcleo do processador da Raspberry Pi 3B+ produzido pela aplicação RPi-monitor.



Fonte: Autoria própria.

Além disso, foram observadas as variações no uso de memória. A Raspberry Pi 3B+ possui 922 MB de memória RAM. Durante o experimento, foram registrados picos de uso de memória de até 900 MB, porém, a média se manteve entre 600 e 850 MB. Essa variação se manteve mesmo após a adição da coleta dos dados de um sensor adicional. Logo, essa adição não resultou no aumento do consumo de memória.

Com o sistema de coleta e apresentação de dados em funcionamento, a Raspberry Pi utiliza entre 2% e 75% do seu poder de processamento. Essas informações fornecem uma visão abrangente do desempenho da Raspberry Pi 3B+ ao longo do período de monitoramento. É importante ressaltar que o experimento foi conduzido sem o uso de ventilação forçada, destacando a capacidade da Raspberry Pi 3B+ de lidar com as demandas do sistema sem problemas de superaquecimento ou estouro de memória.

4.5 APRESENTAÇÃO VISUAL DOS GRÁFICOS NA APLICAÇÃO WEB

Foi conduzido um experimento para avaliar a capacidade da aplicação Web desenvolvida utilizando o Grafana em apresentar os dados atualizados e com horário correto do sistema de forma visual, conforme sua finalidade específica. O objetivo era verificar se a aplicação desempenha adequadamente a função para a qual foi desenvolvida.

A aplicação Web teve seu *dashboard* configurado de maneira que apresente as últimas leituras dos sensores nos painéis superiores. Os dois painéis seguintes possuem gráficos de linhas que apresentam os valores lidos em cada sensor. É possível observar na Figura 50 a apresentação da aplicação Web do sistema onde o *zoom* estava configurado para os últimos 2 dias e foi realizada no dia 10/07/2023 as 18:22. Se os dados estiverem sendo apresentados visualmente com horários incorretos ou desatualizados, a visualização não será considerada correta. A precisão e a atualização dos dados são elementos essenciais para garantir que a informação seja representada de maneira confiável. Caso contrário a confiabilidade da visualização pode ser comprometida.

Figura 50: Dashboard da aplicação Web utilizando *zoom* de 2 dias em 10/07/2023, 18:22.

Fonte: Autoria própria.

Com o objetivo de validar os dados apresentados pela aplicação no subsistema de névoa, instalado na Raspberry Pi, foi realizado o seguinte experimento. O software DBeaver, um cliente de banco de dados, foi utilizado para executar a consulta ao banco de dados para a comparação com os dados apresentados nos gráficos. A consulta em questão tinha como objetivo mostrar os quatro conjuntos de dados mais recentes registrados por cada sensor, ordenados pelo ID dos dados. Na aplicação Web foi configurado um painel, adicionando um gráfico do tipo tabela para exibir os dados. Além disso, um recurso de *zoom* foi configurado para apresentar somente os últimos cinco minutos de dados lidos, a Figura 51 apresenta essas configurações e aparência.

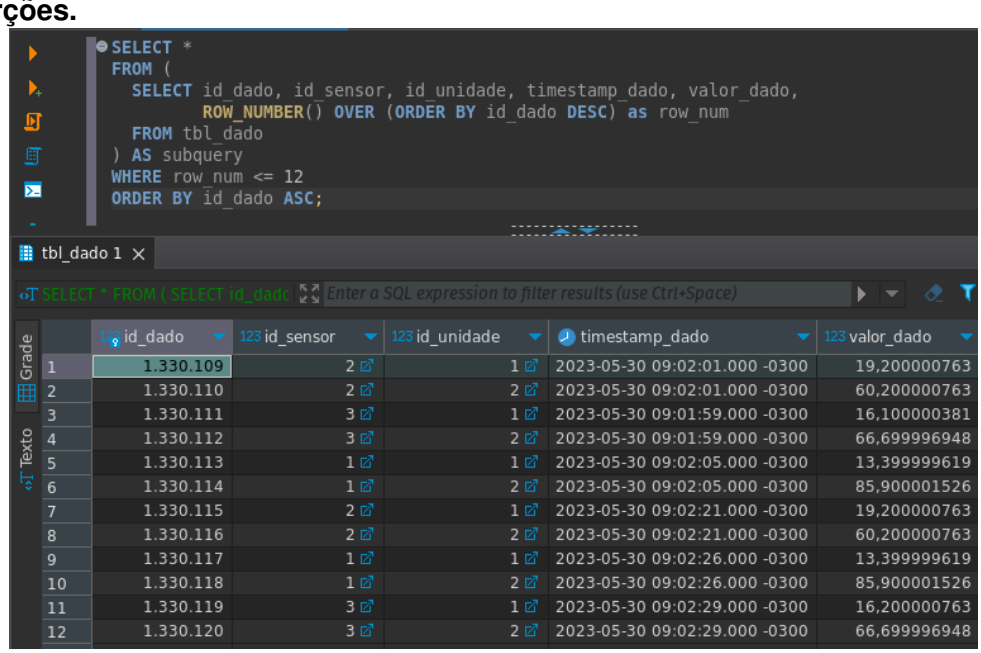
Figura 51: Aparência da Dashboard do Grafana configurado para a realização do experimento.

Fonte: Autoria própria.

A consulta realizada a partir do DBeaver no banco de dados armazenado no subsistema de névoa deve apresentar os mesmos dados que o gráfico de tabela da aplicação Web. Na Figura 52 está a saída da execução da consulta com 12 linhas, apresentando os 2 últimos valores de temperatura e os 2 últimos valores de umidade relativa do ar, para cada um dos 3 sensores. Os últimos 12 IDs, de 1.330.109 a 1.330.120, correspondem aos últimos dados retornados pela consulta em 30/05/2023

às 09:02:40.

Figura 52: Resultado da consulta em 30/05/2023 às 09:02:40 no Dbeaver com as últimas 12 inserções.



	id_dado	id_sensor	id_unidade	timestamp_dado	valor_dado
1	1.330.109	2	1	2023-05-30 09:02:01.000 -0300	19,200000763
2	1.330.110	2	2	2023-05-30 09:02:01.000 -0300	60,200000763
3	1.330.111	3	1	2023-05-30 09:01:59.000 -0300	16,100000381
4	1.330.112	3	2	2023-05-30 09:01:59.000 -0300	66,699996948
5	1.330.113	1	1	2023-05-30 09:02:05.000 -0300	13,399999619
6	1.330.114	1	2	2023-05-30 09:02:05.000 -0300	85,900001526
7	1.330.115	2	1	2023-05-30 09:02:21.000 -0300	19,200000763
8	1.330.116	2	2	2023-05-30 09:02:21.000 -0300	60,200000763
9	1.330.117	1	1	2023-05-30 09:02:26.000 -0300	13,399999619
10	1.330.118	1	2	2023-05-30 09:02:26.000 -0300	85,900001526
11	1.330.119	3	1	2023-05-30 09:02:29.000 -0300	16,200000763
12	1.330.120	3	2	2023-05-30 09:02:29.000 -0300	66,699996948

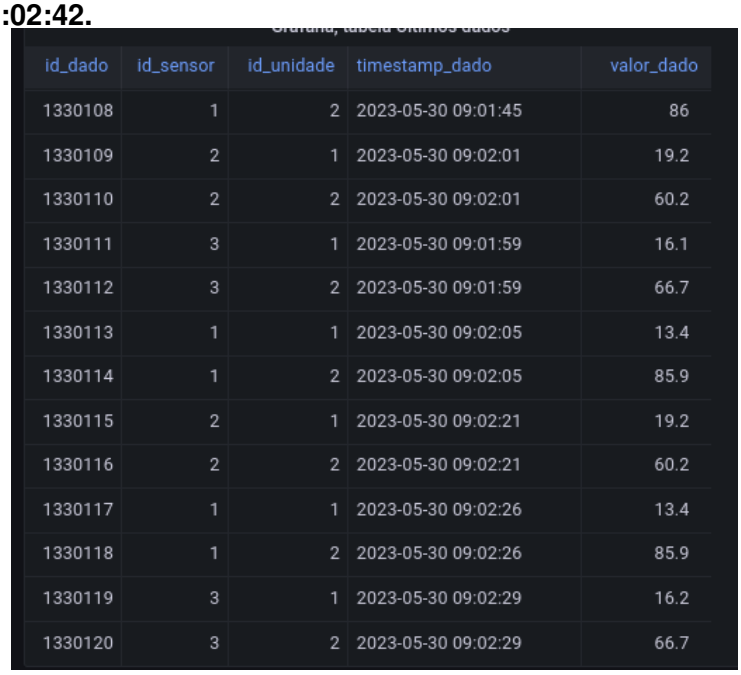
Fonte: Autoria própria.

A Figura 53 apresenta a tabela que permite a visualização dos dados. A barra de navegação da tabela foi arrastada até o fim, possibilitando a visualização dos últimos dados consultados pela aplicação Web após atualizar os painéis em 30/05/2023 às 09:02:42.

Na Figura 53 é possível verificar a apresentação dos dados com IDs entre 1.330.108 e 1.330.120. Após a realização da comparação entre as saídas exibidas na aplicação Web e da consulta pelo Dbeaver, constatou-se que o maior identificador de dado lido em ambas as consultas foi o 1.330.120. Esse identificador foi utilizado como base para apresentar a comparação, tendo em mente que as demais linhas consultadas também exibiram os dados corretamente.

Na linha correspondente ao identificador 1.330.120, encontra-se a leitura proveniente do sensor com ID 3, denominado caixa teste. Essa leitura refere-se à unidade de medida 2, que representa a umidade relativa do ar. O registro foi realizado no dia 30/05/2023, às 09:02:29. O valor registrado para essa leitura foi de 66,7%. Caso o valor na coluna timestamp_dado estiver sendo apresentado com o final +00, para obter o horário de Brasília da coleta, é necessário subtrair 3 unidades do valor da

Figura 53: Últimos dados que a aplicação Web retorna em sua consulta realizada 30/05/2023 às 09:02:42.



id_dado	id_sensor	id_unidade	timestamp_dado	valor_dado
1330108	1	2	2023-05-30 09:01:45	86
1330109	2	1	2023-05-30 09:02:01	19.2
1330110	2	2	2023-05-30 09:02:01	60.2
1330111	3	1	2023-05-30 09:01:59	16.1
1330112	3	2	2023-05-30 09:01:59	66.7
1330113	1	1	2023-05-30 09:02:05	13.4
1330114	1	2	2023-05-30 09:02:05	85.9
1330115	2	1	2023-05-30 09:02:21	19.2
1330116	2	2	2023-05-30 09:02:21	60.2
1330117	1	1	2023-05-30 09:02:26	13.4
1330118	1	2	2023-05-30 09:02:26	85.9
1330119	3	1	2023-05-30 09:02:29	16.2
1330120	3	2	2023-05-30 09:02:29	66.7

Fonte: Autoria própria.

hora.

É importante ressaltar que as duas consultas realizadas, tanto no Grafana quanto no SGBD PostgreSQL, apresentaram os mesmos valores para essa leitura específica. Esse resultado demonstra que tanto a coleta dos dados quanto a sua apresentação estão corretamente configuradas e em conformidade, demonstrando a precisão da apresentação dos valores na aplicação Web.

4.6 SISTEMA DE ALARMES PARA TEMPERATURA E UMIDADE RELATIVA DO AR

O sistema foi projetado com o propósito de detectar e sinalizar ao criador de abelhas a ocorrência de anomalias. O experimento foi conduzido com o intuito de demonstrar o funcionamento adequado dos alarmes, os quais desempenharão um papel fundamental na proteção das colônias. Este experimento foi registrado em vídeo, editado para aprimorar a qualidade do áudio e remover períodos de espera, e está disponível para consulta em `https://youtu.be/t1odGqu7W3w`.

Este experimento foca no uso dos alarmes, que geram modificações no painel

do tipo grau da aplicação Web, que exibe as informações de temperatura e umidade registradas na última leitura feita em cada sensor. Essas características relacionadas às temperaturas e umidade externas e interna da caixa01 podem ser observadas na Figura 54 em uma situação normal em que não há o acionamento dos alarmes. Na parte externa do mostrador pode ser observada a cor que o mostrador irá aparecer ao atingir cada faixa definida.

Figura 54: Painel da aplicação Web com medidores de temperatura e umidade em uma situação normal sem o acionamento dos alarmes.

Fonte: Autoria própria.

Para o registro do experimento, foram apontadas uma câmera com a visão superior e outra com a visão frontal da bancada, conforme pode ser observado na Figura 55. Ao lado esquerdo é exibida a visão superior e ao lado direito a visão frontal dos itens do experimento. Foram dispostos na bancada os seguintes itens:

- Caixa modelo INPA com o módulo coletor e sem abelhas, chamada de caixa02, além de ser apresentada como Caixa Teste;
- Visor do Meliponário;
- Secador de cabelos;
- Embalagem de gelo rígido reutilizável;

- Fonte de alimentação 5v para os dispositivos;
- Computador apresentando as leituras da aplicação Web.

Figura 55: Visões superior e frontal da bancada preparada para o experimento.

Fonte: Autoria própria.

A configuração das faixas de temperatura e umidade normais foram baseadas nas publicações de Loli (2008) e AIDAR (2010). Também foi efetuada a leitura dos valores presentes no *dataset*, coletado entre 15 de fevereiro de 2022 até 10 de março de 2023, para definir os valores de acionamento do alarme. Estes valores na caixa01 apresentaram a umidade relativa do ar variando entre 32 e 88% e a temperatura interna entre 7,5 ℃ e 34,6 ℃. Nessa caixa não havia pedra de aquecimento. Logo, foram definidos os valores normais de temperatura entre 15 ℃ e 35 ℃, e de umidade relativa do ar entre 35% e 89%.

No início do experimento, foi aberta a caixa teste, expondo o ESP D1 mini e seu respectivo sensor. Foi colocado o gelo rígido sobre o sensor, dessa forma forçando a temperatura a cair. Foi aguardada a ativação do alarme no Visor do Meliponário, que ocorreu assim que o sensor efetuou a publicação da temperatura abaixo de 15 ℃. A Figura 56 ilustra esse procedimento.

A temperatura caiu até 13,1 ℃, ativando o alarme no Visor do Meliponário. O Visor do Meliponário alterou a cor do LED de verde para vermelho e terminou de apresentar as informações lidas antes do alarme ser ativado. Posteriormente, passou a piscar o LED em vermelho e apresentou a mensagem: “Caixa02-ALARME, tmp 12,4”, demonstrando que na leitura seguinte, a temperatura foi reduzida a 12,4 ℃.

Figura 56: Gelo rígido disposto sobre o sensor.

Fonte: Autoria própria.

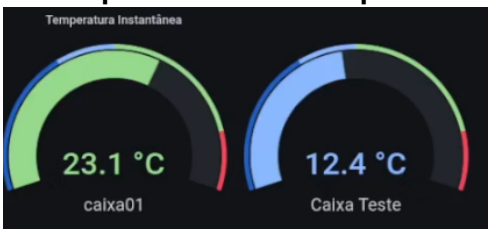
A mensagem de alerta no Visor do Meliponário e o LED aceso em vermelho são observados na Figura 57.

Figura 57: Visor do Meliponário com alarme de temperatura baixa.

Fonte: Autoria própria.

No painel do tipo grau na aplicação Web, demonstrado na Figura 58, a temperatura mostrou-se em um tom de azul claro. Essas duas funcionalidades do sistema, a informação na aplicação Web, junto do Visor do Meliponário, informam ao meliponicultor que uma ação deve ser tomada na referida caixa de abelhas.

Figura 58: Aplicação Web apresentando a temperatura da Caixa Teste baixa.



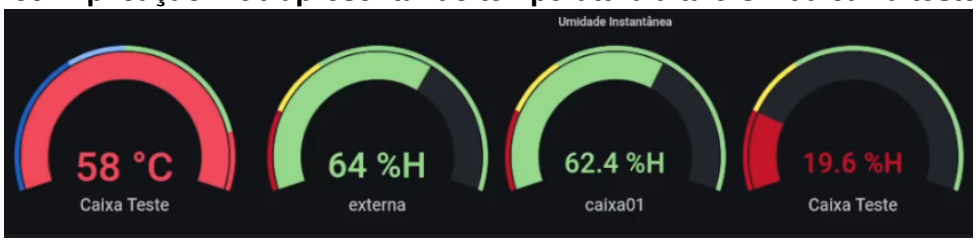
Fonte: Autoria própria.

Em seguida, o gelo rígido foi retirado e, após a temperatura ter atingido 15 ℃, o painel da aplicação Web voltou a exibir a temperatura da Caixa Teste em verde. O

Visor do Meliponário deixou de mostrar a situação de alarme, retornando a cor do seu LED para verde e exibindo as informações de temperatura e umidade relativa para todos os sensores: caixa01, caixa02 e externo à caixa01.

O experimento subsequente foi realizado utilizando o fluxo de ar quente do secador de cabelos, direcionado para o sensor caixa02. Isso resultou na redução da umidade relativa do ar e no aumento da temperatura. Na Figura 59, é possível observar o comportamento do indicador em forma de grau, exibindo uma temperatura de 58 ℃ e uma umidade relativa do ar de 19,6%.

Figura 59: Aplicação Web apresentando temperatura alta e UR da caixa teste baixa.



Fonte: Autoria própria.

A Figura60 mostra o Visor do Meliponário no momento do experimento. Na imagem o LED estava piscando em vermelho, também é exibida no *display* LCD a mensagem: “caixa02-alarme, UR 19 tmp 58.0”.

Figura 60: Visor do Meliponário apresentando alarme de temperatura e umidade relativa na caixa02.

Fonte: Autoria própria.

Após poucos minutos após desligar o secador de cabelos, a temperatura e umidade relativa do ar na caixa teste estabilizaram dentro da faixa definida como

estável. A aplicação Web voltou a apresentar os valores na cor verde e o Visor do Meliponário alterou a cor do LED para verde aceso constante, exibindo as informações de leituras dos sensores.

Com base nesse experimento, foi possível validar os acionamentos dos gatilhos, demonstrando o comportamento esperado conforme o propósito do sistema. O sistema foi capaz de acionar corretamente os gatilhos em situações em que a temperatura ficou abaixo ou acima da faixa definida como normal, assim como para variações na umidade relativa. Isso comprova a eficácia do sistema em identificar e responder adequadamente às condições anormais detectadas. Essa validação é de extrema importância, pois garante que o sistema é capaz de detectar e alertar o criador de abelhas sobre possíveis problemas ou situações críticas nas colmeias, permitindo uma intervenção rápida e eficaz. Com essa capacidade de monitoramento e acionamento de gatilhos, é possível detectar anomalias na colmeia de forma mais rápida e eficiente.

4.7 GERAÇÃO E DISPONIBILIZAÇÃO DO *DATASET*

Durante o período de 15 de fevereiro de 2022 a 10 de março de 2023, foram coletados dados de sensores de temperatura e umidade, instalados em uma caixa de ASF da espécie MQQ. Essas medições foram realizadas tanto internamente quanto externamente à caixa. Inicialmente, os dados eram armazenados no SGBD InfluxDB, no entanto, um problema ocorreu com o cartão de memória da Raspberry Pi onde os dados estavam armazenados, o que dificultou a recuperação dessas informações. Como solução, optou-se por substituir o InfluxDB para o PostgreSQL como SGBD. Posteriormente, foi obtido êxito na recuperação dos dados armazenados no InfluxDB, e, estes dados foram inseridos no PostgreSQL, abrangendo assim o período completo de coletas mencionado no início do parágrafo.

Foi implementada uma aplicação em Python para ler os dados armazenados no PostgreSQL e salvá-los em uma tabela no formato CSV, com os campos:

- ID, contendo o índice;

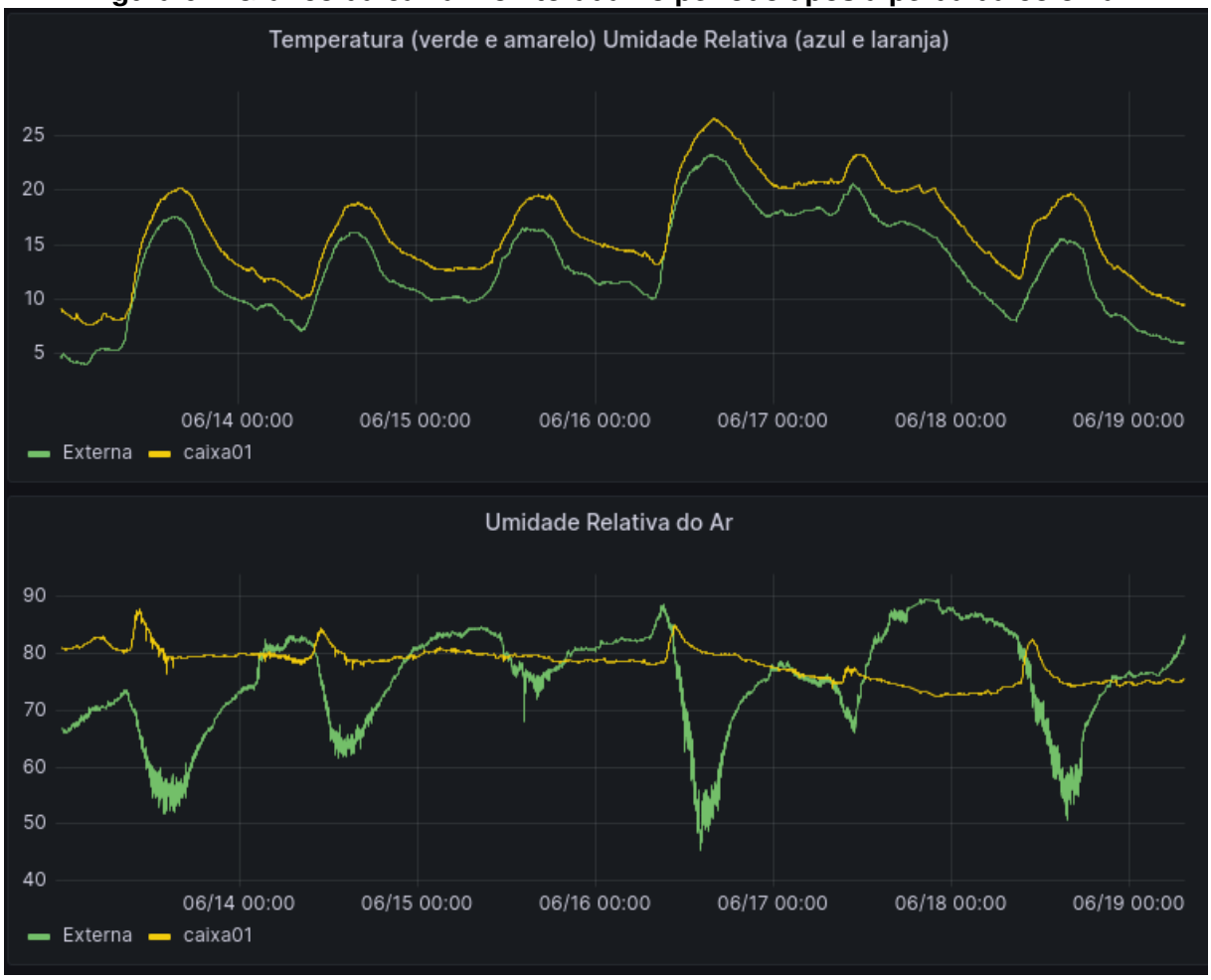
- *Datetime*, contendo um carimbo de data e hora no momento da aferição dos sensores;
- *Sensor*, com o nome do módulo coletor que o dado foi recebido;
- *Measure*, informando a unidade de medida do dado coletado;
- *Valuemeasure*, que mostra o valor coletado pelo sensor.

Durante o período de coleta de dados nos sensores chamados external e caixa01, uma colônia de abelhas da espécie MQQ sofreu um declínio e foi perdida. Infelizmente, os dados desses sensores não estavam sendo monitorados ativamente, e somente após algum tempo foi observado que não havia mais abelhas na caixa. Para comparação, foi gerado um gráfico de 6 dias do período sem abelhas, apresentado na Figura 61, e da colônia saudável em um período de temperaturas baixas, representado na Figura 62. Foi observado que a variação entre a temperatura externa e a interna da caixa01, na colônia vazia, foi de, no máximo, 3,7 ℃. No dia mais frio, aferiu-se que a temperatura da caixa01 era de 7,7 ℃, enquanto a temperatura externa era de 4,1 ℃. Ao analisar o gráfico de temperatura da caixa saudável, constatou-se que a menor temperatura externa registrada durante o período foi de 6,8 ℃, enquanto a temperatura da caixa01 nesse momento era de 14 ℃. A menor temperatura registrada no período foi de 13,5 ℃ e a menor diferença de temperatura externa e externa nos períodos frios foi de 8 ℃.

Ainda utilizando as Figuras 61 e 62, foi observada a variação da umidade relativa da caixa após a perda da colônia e quando estava saudável. Após a perda da colônia, o gráfico de umidade relativa apresentou várias flutuações ao longo de curtos períodos. Por outro lado, a caixa quando saudável manteve-se estável, sem grandes variações ao longo dos dias. Essa diferença no comportamento da umidade relativa entre as duas situações sugere uma possível relação entre a saúde da colônia e a estabilidade da umidade dentro da caixa.

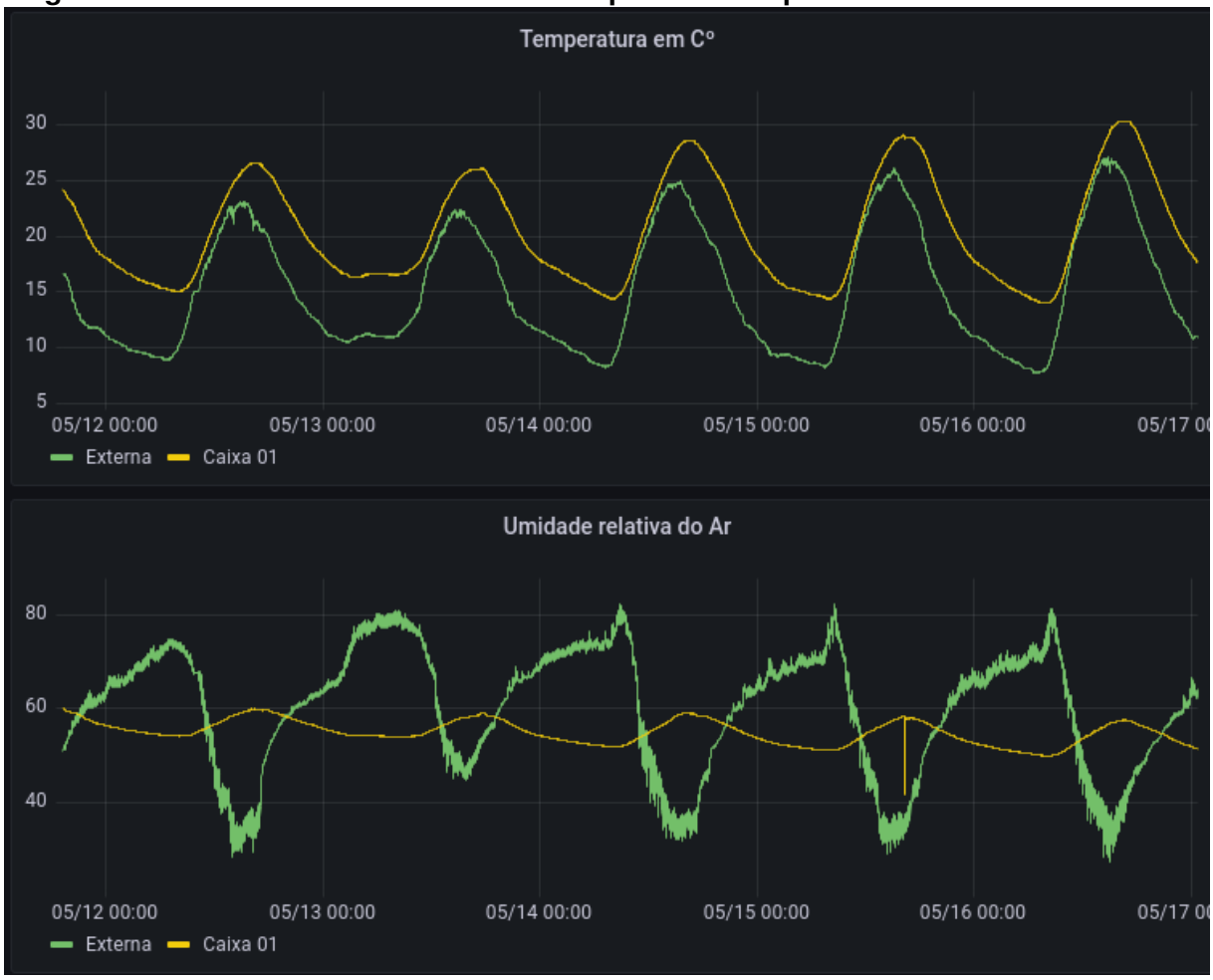
Além dos dados desses sensores, também foram inseridas no *dataset* informações provenientes de equipamentos usados para testes de programação, que foram criados durante o desenvolvimento, mas não realizaram coletas efetivas,

Figura 61: Gráfico da caixa monitorada no período após a perda da colônia.



Fonte: Autoria própria.

Figura 62: Gráfico da caixa monitorada no período em que a colônia estava saudável.



Fonte: Autoria própria.

denominados caixa03 e testePy. No total, o *dataset* possui 1335098 linhas com as coletas dos sensores teste, caixa03 e testePy.

Após a execução de uma aplicação em Python que removeu os dados dos sensores de teste denominados caixa03 e testePy, restaram um total de 1333962 linhas de dados. Em seguida, os dados foram agrupados pelo nome do sensor e divididos com base nas unidades de medida aferidas em cada sensor. A Figura 63 apresenta as estatísticas do *dataset* ao terminar a execução. Observa-se que no sensor caixa01, a umidade variou de 32,3% a 87,8%, com a média e a mediana em torno de 64%. A temperatura variou de 7,6 ℃ a 34,6 ℃, com a média e a mediana próximas a 21,7 ℃. Da mesma forma, foi observado que no sensor localizado externamente na tampa da caixa, denominado external na figura, a umidade variou entre 26,4% e 94,6% com a média de 75,4% e a mediana de 78,2%. A temperatura variou entre 2,6 ℃ e 31,8 ℃, a média foi de 18,6 ℃ e a mediana de 18,9 ℃. O *dataset* completo e o código da aplicação para gerar as estatísticas estão disponíveis em `https://github.com/agl77/coletasASF/`.

Figura 63: Valores de máximo, mínimo, média e mediana para os valores coletados pelos sensores disponíveis no *dataset*.

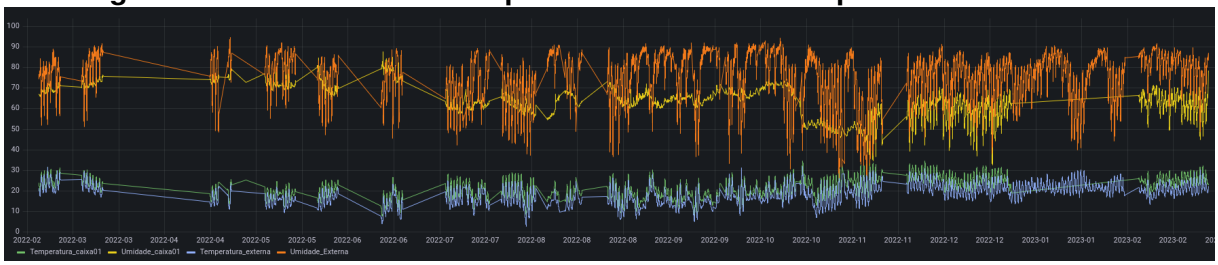
```
Estatísticas do dataset:
                      min    max  median       mean
sensor   measure
caixa01  humidity     32.30  87.83   64.23  63.498294
         temperature   7.58  34.60   21.70  21.650135
external humidity     26.40  94.63   78.20  75.454835
         temperature   2.64  31.80   18.90  18.591159

Data inicial dos dados em dataset: 2022-02-15 11:27:29.010101+00
Data final dos dados em dataset  : 2023-03-10 13:02:25.120331+00
Número de linhas do dataset com os sensores teste: 1.335.098
Número de linhas restantes após a retirada dos sensores de teste: 1.333.962
Tempo total de execução: 1.42 segundos
```

Fonte: Autoria própria.

Na Figura 64 é apresentado o gráfico gerado na aplicação Web, que ilustra a variação das coletas no período compreendido entre fevereiro de 2022 e março de 2023. Para uma melhor observação desses valores, os gráficos também foram gerados dividindo-o em 3 períodos contínuos, observados na Figura 65. Nesses gráficos, a linha verde representa a temperatura registrada pelo sensor caixa01, a linha amarela representa a umidade registrada pelo mesmo sensor, a linha azul representa a temperatura externa e a linha alaranjada representa a umidade externa. É possível

observar que em certos momentos, como no final de março e início de maio de 2022, as linhas apresentam-se retas, indicando que a comunicação falhou. Essas falhas no sistema foram posteriormente corrigidas, resultando na estabilização da coleta e armazenamento dos dados.

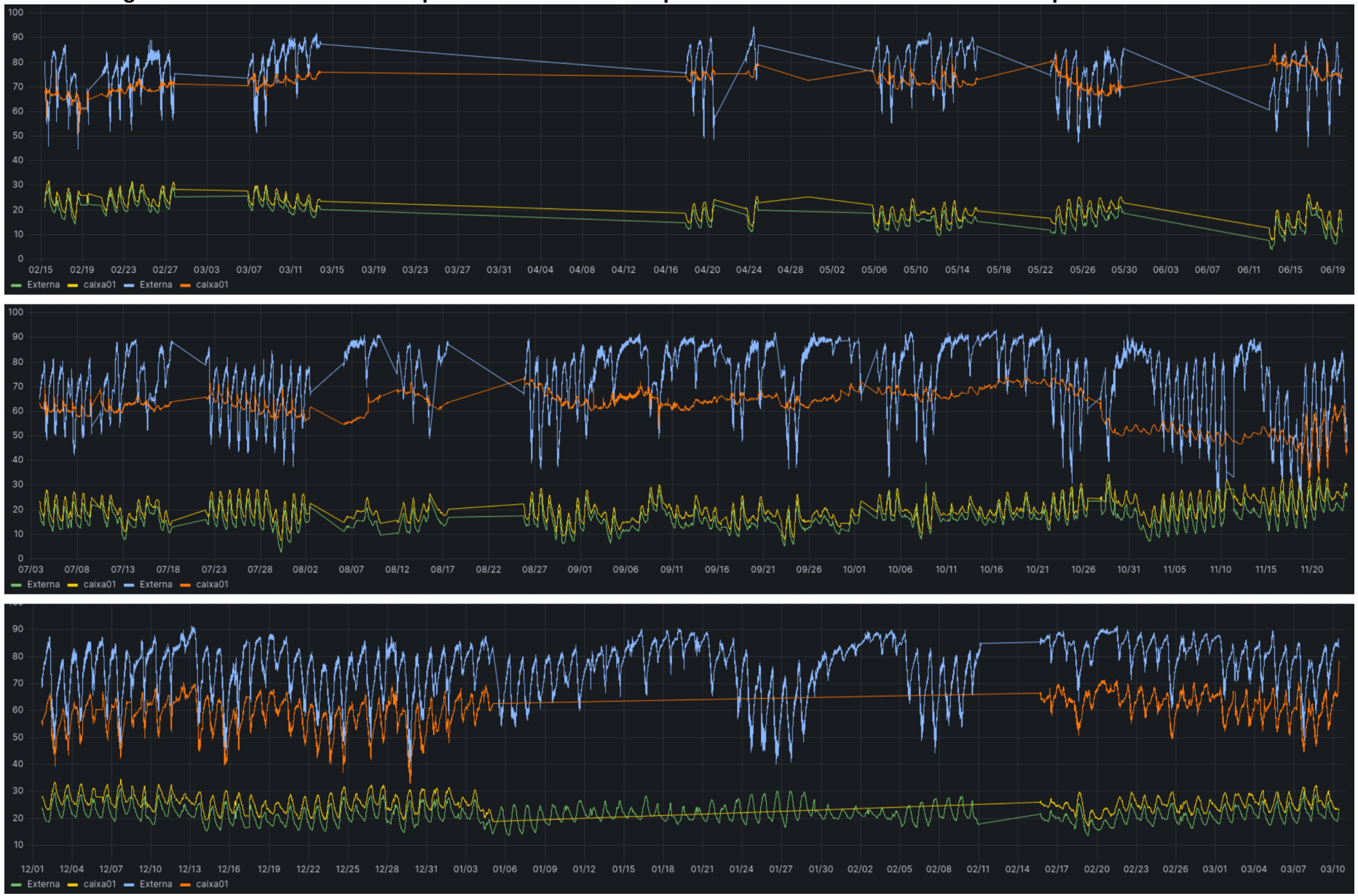
Figura 64: Gráfico de linhas representando as coletas presentes no *dataset*.



Fonte: Autoria própria.

Os resultados obtidos após a realização dos experimentos são promissores, indicando que o sistema desenvolvido pode ser empregado em contextos que requerem o monitoramento e a análise de dados coletados em sensores instalados nas caixas de abelhas. Os testes realizados comprovaram a capacidade do sistema em coletar, armazenar e sincronizar os dados entre os subsistemas de névoa e nuvem, garantindo a atualização das informações. No próximo capítulo, serão apresentadas as conclusões sobre o estudo realizado, bem como as possibilidades de trabalhos futuros.

Figura 65: Gráficos de linhas representando as coletas presentes no *dataset* dividido em três períodos contínuos.



Fonte: Autoria própria.

5 CONCLUSÕES

O principal objetivo desse trabalho foi desenvolver um sistema que possibilita a coleta e armazenamento de dados de sensores para o monitoramento de abelhas, aplicado na espécie *Melípona quadrifasciata quadrifasciata* (MQQ). O desenvolvimento do sistema envolveu a integração de *hardware* e *software*, com o intuito de viabilizar a coleta, armazenamento e visualização de dados, bem como o acionamento de gatilhos em situações específicas.

Com o módulo de coleta pronto, tornou-se evidente que ele pode ser adaptado para a instalação em outros modelos de caixas de abelhas, bem como para outras espécies de abelhas. Isso significa que não há limitação em relação à espécie de abelha criada em caixa racional. A flexibilidade do sistema permite sua aplicação em diferentes cenários, atendendo às necessidades específicas de criadores de abelhas de diferentes espécies e usando diferentes tipos de caixas. Além disso, o sistema pode ser utilizado em estudos e pesquisas relacionados às abelhas, permitindo a coleta de dados valiosos que contribuem para a compreensão do comportamento, saúde e desenvolvimento dessas polinizadoras.

Para alcançar esse objetivo, foi preciso superar desafios, tais como: a seleção de sensores e dispositivo que possibilitassem a coleta de dados; desenvolvimento de um módulo de coleta para a integração dos sensores em uma caixa de abelhas; desenvolvimento de aplicações que integram as coletas com o armazenamento em um banco de dados, que verificam valores para acionar gatilhos e que sincronizam os dados entre serviços na névoa e na nuvem. Também foram realizadas integrações com aplicações que permitiram a visualização intuitiva das coletas efetuadas.

Para a construção dos módulos de coleta foram utilizadas placas ESP 8266 do modelo D1 mini, com o sensor SHTC3, que juntos coletam dados de temperatura

e umidade e os enviam para o subsistema de névoa utilizando conexão Wi-Fi e o protocolo de comunicação MQTT. Foi utilizada uma Raspberry Pi 3B+ e um cartão de memória de 32 GB como dispositivo de névoa. Nele foram configurados o SGBD PostgreSQL e a aplicação Web baseada no Grafana, além de executar aplicações desenvolvidas em Python. Também foi adicionado nesse subsistema o Visor do Meliponário, um dispositivo composto por um ESP 8266 modelo D1 mini, um LED RGB e um visor LCD de 16 colunas e duas linhas, que permite ao usuário visualizar as leituras dos módulos coletores e os alarmes, caso sejam ativados.

Com o sistema em funcionamento, foram executados experimentos para comprovar a eficácia de cada um dos seus elementos, cujos resultados são apresentados no Capítulo 4. Estes experimentos demonstram a comunicação dos sensores em caixa de abelha com a aplicação da névoa, que trata e insere os dados recebidos através do protocolo MQTT no banco de dados PostgreSQL. Além disso, foram testadas as aplicações de sincronismo dos SGBD armazenados em névoa e nuvem. Do mesmo modo, houve testagem da aplicação de acionamento de gatilhos caso a temperatura interna das caixas ou a umidade relativa do ar não estejam na faixa definida como normal, exibindo estes dados no Visor do Meliponário.

Além disso, na aplicação Web é possível visualizar os dados coletados em gráficos, gerados a partir dos dados armazenados no SGBD, tanto para o subsistema em névoa quanto para o serviço em nuvem, permitindo seu acesso de forma remota. No caso do acesso à aplicação Web na névoa, o usuário pode observar os dados, sendo disponibilizados mais rapidamente do que no caso do acesso à aplicação Web na nuvem. Porém, os dados também podem ser observados na aplicação Web na nuvem, desde que a conexão com a Internet na névoa esteja estável. Em caso de perda de conexão, os dados continuam a ser coletados e armazenados na névoa, e a sincronização é iniciada assim que a conexão for restabelecida.

O Visor do Meliponário tornou-se uma parte importante, para informar a identificação de problemas nas caixas de ASF monitoradas, mesmo sem a necessidade do acesso do usuário ao painel da aplicação Web. Essa observação foi pelo uso de um LED que fica acesso em verde se as coletas apresentarem as informações dentro das faixas especificadas, e, piscando em vermelho se a coleta

apresentar dados fora da faixa especificada, além de apresentar em seu *display* LCD os dados das coletas ou alarmes. Dessa forma, o usuário tem a informação sobre o alarme facilmente acessível já próxima à colmeia sendo monitorada. Já a aplicação Web permite que a colmeia seja monitorada mesmo à distância e permite também uma visualização do histórico dos valores coletados para uma análise a longo prazo.

Durante a implementação do sistema foi percebido que a utilização de placas modelo D1 mini com o controlador ESP 8266 requereu mais tempo para a implementação do MQTT com QoS 1 do que se opção da placa tivesse sido por uma com controlador ESP 32. Esse *upgrade* na placa poderia ter adiantado o desenvolvimento em aproximadamente um mês. Posteriormente, foram verificados os valores entre a placa ESP D1 Mini com controlador ESP 8266 e uma placa com o controlador ESP 32 em lojas *on-line* brasileiras, como a CurtoCircuito (CURTOCIRCUITO, 2023), Eletrogate (ELETROGATE, 2023) e Piscaled (PISCALED, 2023), acessadas em 15 de junho de 2023. A diferença observada é que uma placa ESP 32 custa a partir de R$ 16,21 a mais que uma placa D1 mini com controlador ESP8266. Essa diferença é ganha no tempo de implementação, sem a necessidade de compilações das bibliotecas do MicroPython externas ao módulo nem de adaptações para permitir o uso do QoS 1.

Outra limitação do sistema diz respeito à comunicação entre os módulos coletores e o subsistema de névoa. Ao executar o teste de comunicação, percebeu-se que é necessário um aprofundamento no estudo da conexão MQTT entre o sensor executando MicroPython e o *broker* MQTT, assim como das ferramentas utilizadas. Com estas informações, pode-se aprimorar o código em MicroPython para manter a conexão do sensor por um período maior que o configurado durante os experimentos.

No geral, esta dissertação apresentou um sistema eficiente para a coleta, armazenamento e visualização de dados da colônia de ASF da espécie *Melipona quadrifasciata quadrifasciata* (MQQ), cumprindo o objetivo principal estabelecido. A integração de *hardware* e *software* permitiu a coleta precisa dos dados dos sensores, enquanto as funcionalidades de armazenamento em SGBD e visualização por meio da aplicação Web forneceram uma interface intuitiva para o usuário. Esta dissertação contribuiu para o avanço na área de monitoramento de colônias de abelhas sem ferrão,

fornecendo um modelo eficiente para coleta, armazenamento e visualização de dados. Espera-se que os resultados obtidos sirvam como base para futuras pesquisas e desenvolvimentos nessa área, promovendo a preservação e o estudo dessas espécies fundamentais para o ecossistema.

5.1 TRABALHOS FUTUROS

Esta dissertação apresentou o desenvolvimento e implementação de um sistema funcional de coleta, armazenamento e visualização de dados de uma colônia de abelha MQQ. No entanto, foram percebidas oportunidades para trabalhos futuros nessa área, relacionadas a seguir.

Cabe inferir, inicialmente, que a aplicação do sistema poderia ser aprimorada ao instalar módulos coletores em mais caixas de abelha MQQ, preferencialmente em parceria com criadores que já realizem um acompanhamento humano das colônias. Supostamente, tal abordagem permitiria uma identificação mais precisa e preditiva de anomalias pela associação dos dados coletados aos registros de intervenções e ocorrências feitas pelos criadores. Ao instalar sensores em mais caixas de abelhas MQQ, deduz-se que seria possível coletar dados de um maior número de colônias, assim proporcionando a ampliação da observação e comparação dos dados coletados com base no comportamento das abelhas e do ambiente em que estão inseridas, viabilizando o estudo de diferentes fatores que possam influenciar a saúde e o desenvolvimento das colônias, como variações de temperatura, umidade, pressão atmosférica entre outros dados que possam ser captados pelos sensores dos módulos.

Sob a perspectiva da ampliação do número de caixas de MQQ monitoradas, seria possível gerar um *dataset* mais representativo, o que pode contribuir para uma melhor compreensão das variações entre as caixas monitoradas. Também poderiam ser incorporados novos sensores ao módulo coletor, exigindo uma adaptação maior do sistema, com a criação de novos tópicos MQTT e incorporação de novos gráficos à aplicação Web. Tal *dataset* poderia ser utilizado para aplicar técnicas de aprendizado de máquina visando aprimorar o acionamento de gatilhos e a detecção de anomalias.

Uma sugestão de trabalho futuro seria aprimorar o Visor do Meliponário, substituindo o dispositivo de exibição LCD de 16 colunas x 2 linhas por um com capacidade de exibir informações de maneira mais abrangente. Como exemplo, um visor com maior capacidade, como um display gráfico ou um visor LCD maior, permitiria a exibição da hora atual e informações de vários módulos coletores com suas respectivas horas de última coleta simultaneamente. Dessa forma, forneceria aos criadores acesso às informações atualizadas e em tempo real, o que permitiria intervenções e tomadas de decisão mais ágeis.

É possível considerar também a investigação dos fatores que podem sobrecarregar o *broker* MQTT, como o número de dispositivos conectados, a frequência do envio das mensagens e a carga do processamento do servidor, o que possivelmente otimizaria o desempenho do sistema. Com bases nos estudos propostos, também poderão ser desenvolvidas estratégias para ajustar os valores de *keepalive* entre os dispositivos de coleta e o *broker*, estabelecendo um tempo adequado de conexão ativa. Ação que contribuirá para minimizar a perda de dados durante interrupções temporárias na conexão.

Uma área que pode ser explorada para otimizar o desempenho e a eficiência do sistema é a melhoria das aplicações de sincronismo de dados entre a névoa e a nuvem. Dentre as possibilidades está a compilação e o envio de mais de uma linha de dados por vez, o que pode resultar em um processo mais eficiente de transferência de informações. Essa otimização do sincronismo de dados contribuiria para um melhor aproveitamento dos recursos e uma redução do tempo necessário para transmitir os dados da névoa para a nuvem.

Uma sugestão relevante para aprimorar o sistema é a criação de uma interface intuitiva para a configuração dos sensores. Atualmente, a configuração dos sensores é realizada manualmente, o que demanda conhecimentos técnicos específicos. No entanto, ao desenvolver uma interface amigável e de fácil utilização, os usuários poderiam configurar os sensores de maneira mais simples e rápida, eliminando a necessidade de conhecimentos avançados em programação e conexão de dispositivos. Essa abordagem proporcionaria maior flexibilidade e autonomia aos usuários, facilitando a adaptação do sistema às suas necessidades específicas. Além

disso, uma interface intuitiva tornaria o processo de configuração mais acessível, permitindo que um maior número de pessoas possa utilizar o sistema com facilidade e eficientemente.

Outra proposta interessante é o desenvolvimento de um aplicativo móvel exclusivo para o Visor do Meliponário. O Visor do Meliponário desempenha um papel fundamental no sistema, fornecendo informações sobre o estado das caixas de abelhas sem ferrão, como temperatura, umidade, alarmes e valores que requerem atenção. Com a criação de um aplicativo móvel dedicado ao Visor do Meliponário, os usuários teriam a conveniência de acessar facilmente essas informações em seus dispositivos móveis, como *smartphones* ou *tablets*. Essa solução permitiria que os criadores de abelhas consultassem os dados de maneira rápida, intuitiva e conveniente, mesmo quando estivessem longe das colmeias. Além disso, um aplicativo móvel poderia oferecer recursos adicionais, como notificações em tempo real e histórico de dados, possibilitando uma análise mais abrangente do desempenho das colônias e auxiliando os criadores na tomada de decisões com base nos dados coletados.

Como última sugestão neste trabalho, seria benéfico conduzir uma investigação mais aprofundada sobre as perdas de pacotes na Raspberry Pi quando conectada ao Wi-Fi. Durante o desenvolvimento do sistema, notou-se a ocorrência de perdas de dados ao longo do tempo ao utilizar a conexão Wi-Fi, ao contrário do que foi observado quando a Raspberry Pi foi conectada via Ethernet. Para compreender melhor esse comportamento e identificar as possíveis causas das perdas de pacotes, seria recomendável realizar testes mais detalhados. Essa análise mais aprofundada permitiria obter informações precisas sobre as razões por trás das perdas de pacotes na conexão Wi-Fi da Raspberry Pi, possibilitando a implementação de medidas corretivas para mitigar essas perdas e garantir uma comunicação estável e confiável entre os dispositivos.

REFERÊNCIAS

AIDAR, D. S. **A mandaçaia: biologia, manejo e multiplicação artificial de colônias de abelhas, com especial referência à Melipona quadrifasciata Lep**. 2. ed. Ribeirão Preto: FUNPEC, 2010. 161 p.

AL-FUQAHA, A.; GUIZANI, M.; MOHAMMADI, M.; ALEDHARI, M.; AYYASH, M. Internet of things: A survey on enabling technologies, protocols, and applications. **IEEE Communications Surveys Tutorials**, v. 17, n. 4, p. 2347–2376, 2015.

ANUAR, N. H. K.; YUNUS, M. A. M.; BAHARUDDIN, M. A.; SAHLAN, S.; ABID, A.; RAMLI, M. M.; AMIN, M. R. A.; LOTPI, Z. F. M. IoT platform for precision stingless bee farming. In: **2019 IEEE International Conference on Automatic Control and Intelligent Systems (I2CACIS)**. Selangor, Malaysia: IEEE, 2019. p. 225–229.

APUROOP, K. G. S.; XIAN, L. J.; ELARA, M. R.; POVENDHAN, A. P.; SRINIKETH, K.; SAIRAM, B. C. S. C.; TERNTZER, D. N. Towards a cloud-based system architecture for drain inspection robots. In: **2021 4th International Conference on Information and Communications Technology (ICOIACT)**. Yogyakarta, Indonesia: IEEE, 2021. p. 209–214.

ARENDST. **Open source firmware for ESP devices**. 2022. Disponível em: `https://tasmota.github.io/`. Acesso em: 15 de outubro de 2022.

ATLAM, H. F.; WALTERS, R. J.; WILLS, G. B. Fog computing and the internet of things: A review. **big data and cognitive computing**, Multidisciplinary Digital Publishing Institute, v. 2, n. 2, p. 10, 2018.

BATISTA, H. W. d. M. **Uma ferramenta para simulação e monitoramento de comportamentos anômalos em contêineres Docker**. Dissertação (B.S. thesis) — Universidade Federal do Rio Grande do Norte, 2022.

BELLOS, C. V.; FYRARIDIS, A.; STERGIOS, G. S.; STEFANOU, K. A.; KONTOGIANNIS, S. A quality and disease control system for beekeeping. In: **2021 6th South-East Europe Design Automation, Computer Engineering, Computer Networks and Social Media Conference (SEEDA-CECNSM)**. Preveza, Greece: IEEE, 2021. p. 1–4.

BERGER, X. **RPi-Monitor Real Time Monitor For Embedded Devices**. 2014. Disponível em: `https://github.com/XavierBerger/RPi-Monitor`. Acesso em: 11 de fevereiro de 2022.

BONOMI, F.; MILITO, R.; ZHU, J.; ADDEPALLI, S. Fog computing and its role in the internet of things. In: **Proceedings of the first edition of the MCC workshop on Mobile cloud computing**. New York, NY, USA: Association for Computing Machinery, 2012. p. 13–16.

BRAGA, A. R.; FURTADO, L. S.; BEZERRA, A. D. M.; FREITAS, B. M.; CAZIER, J. A.; GOMES, D. G. Applying the long-term memory algorithm to forecast loss of thermoregulation capacity in honeybee colonies. **10 Workshop de Computação Aplicada à Gestão do Meio Ambiente e Recursos Naturais (WCAMA)**, p. 1–14, 2019.

BRAGA, A. R.; GOMES, D. G.; RICHARD, R.; HASSLERE, E. E.; FREITAS, B. M.; CAZIERE, J. A. A method for mining combined data from in-hive sensors, weather and apiary inspections to forecast the health status of honey bee colonies. **Computers and Electronics in Agriculture Volume 169**, p. 0168–1699, 2020.

CAESAR, L. **Síndrome anual da abelha mandaçaia (Melipona quadrifasciata) - o papel de simbiontes, sistema imune e ambiente**. Tese (Doutorado) — Universidade Federal do Rio Grande do Sul, 2020.

CHETTRI, L.; BERA, R. A comprehensive survey on internet of things (IoT) toward 5g wireless systems. **IEEE Internet of Things Journal**, v. 7, n. 1, p. 16–32, 2020.

COLORADO, L. A. M.; MARTÍINEZ-SANTOS, J. C. Leveraging 1-wire communication bus system for secure home automation. In: **Colombian Conference on Computing**. Cham: Springer International Publishing, 2017. p. 759–771.

CONTRERA, F.; VENTURIERI, G. Revisão das interações entre forídeos (diptera: Phoridae) e abelhas indígenas sem ferrão (apidae: Meliponini), e técnicas de controle. In: **VIII Encontro sobre abelhas.** Ribeirão Preto: Embrapa Amazônia Oriental, 2008.

CORDEIRO, C.; CHALLAPALI, K.; BIRRU, D.; SHANKAR, S. Ieee 802.22: the first worldwide wireless standard based on cognitive radios. In: **First IEEE International Symposium on New Frontiers in Dynamic Spectrum Access Networks, 2005. DySPAN 2005.** Baltimore, MD, USA: IEEE, 2005. p. 328–337.

COUSIN, P.; CAUIA, E.; SICEANU, A.; CLEDAT, J. de. The development of an efficient system to monitor the honeybee colonies depopulations. In: **2019 Global IoT Summit (GIoTS)**. Aarhus, Denmark: IEEE, 2019. p. 1–5.

COUTINHO, A. A.; CARNEIRO, E.; GREVE, F. Computação em névoa: Conceitos, aplicações e desafios. In: **Minicursos / XXXIV Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos**. Salvador: Sociedade Brasileira de Computação, 2016. p. 266–315. ISBN 2177-4978.

CRISP, J. **Introduction to microprocessors and microcontrollers**. 2nd. ed. Oxford: Newnes, 2004. ISBN 978-0750659895.

CURTOCIRCUITO. **Componentes Eletrônicos e Arduino**. 2023. Disponível em: `https://curtocircuito.com.br`. Acesso em: 15 de junho de 2023.

DATASHEET, E. Esp8266ex datasheet. **Espressif Systems Datasheet**, p. 1–31, 2015.

DECHMUNEE, P.; DAWAN, P.; TITIROONGRUANG, W.; ATIWONGSANGTHONG, N. Leakage current measurement with the bench test in watchdog timer power down mode for microcontroller device. In: **2017 14th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON)**. Phuket, Thailand: IEEE, 2017. p. 246–249.

DINCULEANĂ, D.; CHENG, X. Vulnerabilities and limitations of MQTT protocol used between IoT devices. **Applied Sciences**, Multidisciplinary Digital Publishing Institute, v. 9, n. 5, p. 848, 2019.

ELETROGATE. **Arduino, Robótica, IoT, Apostilas e Kits**. 2023. Disponível em: https://www.eletrogate.com/. Acesso em: 15 de junho de 2023.

FAN, K.; ZHANG, C.; YANG, K.; LI, H.; YANG, Y. Lightweight nfc protocol for privacy protection in mobile IoT. **Applied Sciences**, Multidisciplinary Digital Publishing Institute, v. 8, n. 12, p. 2506, 2018.

FERREIRA, R. d. C. **Elaboração de formulação semissólida com potencial farmacêutico à base de extrato da Geoprópolis da Melipona Quadrifasciata Anthidioides (Mandaçaia)**. Tese (Doutorado) — Universidade Federal da Bahia, 2021.

FITZGERALD, D. W.; MURPHY, F. E.; WRIGHT, W. M. D.; WHELAN, P. M.; POPOVICI, E. M. Design and development of a smart weighing scale for beehive monitoring. In: **2015 26th Irish Signals and Systems Conference (ISSC)**. Carlow, Ireland: IEEE, 2015. p. 1–6.

FRESINGHELI, K. M. L.; ABREU, S. M.; MAURER, P. F. A.; MOTTA, F. S.; CARABAJAL, C. L. I.; ETHUR, L. Z. A importância das abelhas: Uma proposta de atividade de educação ambiental. **Anais do Salão Internacional de Ensino, Pesquisa e Extensão**, v. 11, n. 3, 2019.

FUJITA, K.; KANAYAMA, Y.; KIM, J.; NAKAJIMA, K. A preliminary study on a voided volume measuring method using noncontact temperature sensors under the toilet seat. **Advanced Biomedical Engineering**, Japanese Society for Medical and Biological Engineering, v. 8, p. 1–6, 2019.

GALEGALE, G. P.; SIQUEIRA, É.; SILVA, C. B. H.; SOUZA, C. A. d. Internet das coisas aplicada a negócios-um estudo bibliométrico. **JISTEM-Journal of Information Systems and Technology Management**, SciELO Brasil, v. 13, p. 423–438, 2016.

GAMESS, E.; HERNANDEZ, S. Performance evaluation of different raspberry pi models for a broad spectrum of interests. **International Journal of Advanced Computer Science and Applications**, Science and Information (SAI) Organization Limited, v. 13, n. 2, 2022.

GEMIM, B. S.; SILVA, F. A. de M.; SCHAFFRATH, V. R. Aspectos socioambientais da meliponicultura na região do vale do ribeira, são paulo, brasil. **Guaju**, Revista Brasileira de Desenvolvimento Territorial Sustentável, v. 8, 2022.

GEMIRTER, C. B.; SENTURCA, C.; BAYDERE, S. A comparative evaluation of AMQP, MQTT and HTTP protocols using real-time public smart city data. In: **2021 6th International Conference on Computer Science and Engineering (UBMK)**. Ankara, Turkey: IEEE, 2021. p. 542–547.

GEORGE, D. **MicroPython**. 2022. Disponível em: https://micropython.org/. Acesso em: 09 de fevereiro de 2022.

GOMES, B. B.; FAITA, M. R.; SEZERINO, A. A.; POLTRONIERI, A. S. Perfil dos meliponicultores e aspectos da criação de abelhas sem ferrão em santa catarina. **Agropecuária Catarinense**, v. 35, n. 3, p. 76–81, dez. 2022.

GRAFANA LABS. **Operational dashboards for your data here, there, or anywhere**. 2022. Disponível em: https://grafana.com. Acesso em: 16 de agosto de 2022.

GRANDO, G. C. **Efeitos da combinação de agrotóxicos glifosato e imidacloprido no desenvolvimento, sistema imunológico e digestório de abelhas Scaptotrigona postica (Latreille, 1807)**. Dissertação (Mestrado) — Universidade Federal de São Paulo, 2022.

GREEN, J. The internet of things reference model. In: **Internet of Things World Forum**. San Jose, CA, USA: CISCO, 2014. p. 1–12.

GROUP, T. P. G. D. **What is PostgreSQL?** 2023. Disponível em: https://www.postgresql.org/about/. Acesso em: 4 de julho de 2023.

HARIPRIYA, A.; KULOTHUNGAN, K. Secure-MQTT: an efficient fuzzy logic-based approach to detect dos attack in MQTT protocol for internet of things. **EURASIP Journal on Wireless Communications and Networking**, SpringerOpen, v. 2019, n. 1, p. 1–15, 2019.

HARUN, A.; ZAABA, S.; KAMARUDIN, L.; ZAKARIA, A.; FAROOK, R. S. M.; NDZI, D. L.; SHAKAFF, A. Stingless bee colony health sensing through integrated wireless system. **Jurnal Teknologi**, v. 77, n. 28, 2015.

HINCH, P. **MicroPython AsynchronousMQTT**. 2019. Disponível em: https://github.com/peterhinch/micropython-mqtt/blob/master/mqtt_as/mqtt_as.py. Acesso em: 12 de fevereiro de 2022.

JESUS, F. T. d. et al. **Sistema de calefação para ninhos de abelhas-sem-ferrão com controle e leitura de temperatura interna por sistema remoto**. Dissertação (Mestrado) — Universidade Federal de Santa Catarina, 2017.

JIANGYI, Z.; DANHONG, C.; YU, Y. t. Design of intelligent hive and intelligent bee farm based on internet of things technology. In: **2019 Chinese Control And Decision Conference (CCDC)**. Nanchang, China: IEEE, 2019. p. 2432–2435.

KAFLE, V. P.; FUKUSHIMA, Y.; HARAI, H. Internet of things standardization in itu and prospective networking technologies. **IEEE Communications Magazine**, IEEE, v. 54, n. 9, p. 43–49, 2016.

KOCK, K. **Micropython SmartHome Node**. 2018. Disponível em: https://github.com/kevinkk525/pysmartnode/tree/master/pysmartnode. Acesso em: 26 de março de 2023.

KODALI, R. K. An implementation of MQTT using cc3200. In: **2016 International Conference on Control, Instrumentation, Communication and Computational Technologies (ICCICCT)**. Kumaracoil, India: IEEE, 2016. p. 582–587.

KRčO, S.; POKRIć, B.; CARREZ, F. Designing IoT architecture(s): A european perspective. In: **2014 IEEE World Forum on Internet of Things (WF-IoT)**. Seoul, Korea (South): IEEE, 2014. p. 79–84.

KULYUKIN, V. Audio, image, video, and weather datasets for continuous electronic beehive monitoring. **Applied Sciences**, Multidisciplinary Digital Publishing Institute, v. 11, n. 10, p. 4632, 2021.

LEE, S.; KIM, H.; HONG, D.-k.; JU, H. Correlation analysis of MQTT loss and delay according to QoS level. In: **The International Conference on Information Networking 2013 (ICOIN)**. Bangkok, Thailand: IEEE, 2013. p. 714–717.

LEIGHTON, B.; COX, S. J. D.; CAR, N. J.; STENSON, M. P.; VLEESHOUWER, J.; HODGE, J. A best of both worlds approach to complex, efficient, time series data delivery. In: DENZER, R.; ARGENT, R. M.; SCHIMAK, G.; HŘEBÍČEK, J. (Ed.). **Environmental Software Systems. Infrastructures, Services and Applications**. Cham: Springer International Publishing, 2015. p. 371–379. ISBN 978-3-319-15994-2.

LEITE, R. V. V.; VICENTE, J. P. C.; OLIVEIRA, T.; BARROS, P. O despertar para as abelhas: educação ambiental e contexto escolar. In: **Congresso Nacional de Educação. Natal**. NATAL, Brasil: CONEDU, 2016. p. 1–12.

LIMA, M. C. d.; ROCHA, S. d. A. Efeitos dos agrotóxicos sobre as abelhas silvestres no brasil. **Brasília: Ibama**, 2012.

LOLI, D. **Termorregulação colonial e energética individual em abelhas sem ferrão Melipona quadrifasciata Lepeletier (Hymenoptera, Apidae, Meliponini)**. Tese (Doutorado) — Universidade de São Paulo, 2008.

MACHHAMER, R.; ALTENHOFER, J.; UEDING, K.; CZENKUSCH, L.; STOLZ, F.; HARTH, M.; MATTERN, M.; LATIF, A.; HAAB, S.; HERRMANN, J.; SCHMEINK, A.; GOLLMER, K.-U.; DARTMANN, G. Visual programmed IoT beehive monitoring for decision aid by machine learning based anomaly detection. In: **2020 9th Mediterranean Conference on Embedded Computing (MECO)**. Budva, Montenegro: IEEE, 2020. p. 1–5.

MADAKAM, S.; LAKE, V.; LAKE, V.; LAKE, V. et al. Internet of things (IoT): A literature review. **Journal of Computer and Communications**, Scientific Research Publishing, v. 3, n. 05, p. 164, 2015.

MAHAMUD, M. S.; RAKIB, M. A. A.; FARUQI, T. M.; HAQUE, M.; RUKAIA, S. A.; NAZMI, S. Mouchak - an IoT basted smart beekeeping system using MQTT. In: **2019 4th International Conference on Robotics and Automation Engineering (ICRAE)**. Singapore: IEEE, 2019. p. 84–88.

MALINOWSKI, A.; YU, H. Comparison of embedded system design for industrial applications. **IEEE Transactions on Industrial Informatics**, v. 7, n. 2, p. 244–254, 2011.

MAXIM INTEGRATED PRODUCTS. **GUIDE TO 1-WIRE COMMUNICATION**. 2016. Disponível em: https://www.maximintegrated.com/en/design/

technical-documents/tutorials/1/1796.html. Acesso em: 14 de agosto de 2022.

MOURA, L. M. D.; GRADELA, A.; COSTA, M. M. da; SILVA, R. de F.; PEIXOTO, R. de M. Mel de mandaçaia e proópolis vermelha em lesões traumáticas de equídeos - literature review. In: **Estudos sobre as engenharias**. Triunfo, PE: Omnis Scientia, 2021. cap. 6, p. 75–87.

MUCKENFUHS, F. **Desenvolvimento de um sistema de monitoramento e controle de processos produtivos a partir do conceito de IoT**. Monografia (Trabalho de Conclusão de Curso) — Universidade Regional do Noroeste do Estado do Rio Grande do Sul, 2020.

MURPHY, F. E.; MAGNO, M.; O'LEARY, L.; TROY, K.; WHELAN, P.; POPOVICI, E. M. Big brother for bees (3b) — energy neutral platform for remote monitoring of beehive imagery and sound. In: **2015 6th International Workshop on Advances in Sensors and Interfaces (IWASI)**. Gallipoli, Italy: IEEE, 2015. p. 106–111.

MURPHY, F. E.; POPOVICI, E.; WHELAN, P.; MAGNO, M. Development of an heterogeneous wireless sensor network for instrumentation and analysis of beehives. In: **2015 IEEE International Instrumentation and Measurement Technology Conference (I2MTC) Proceedings**. Pisa, Italy: IEEE, 2015. p. 346–351.

MURPHY, F. E.; SRBINOVSKI, B.; MAGNO, M.; POPOVICI, E. M.; WHELAN, P. M. An automatic, wireless audio recording node for analysis of beehives. In: **2015 26th Irish Signals and Systems Conference (ISSC)**. Carlow, Ireland: IEEE, 2015. p. 1–6.

NAITO, K. A survey on the internet-of-things: Standards, challenges and future prospects. **Journal of Information Processing**, v. 25, p. 23–31, 01 2017.

NASCIMENTO, G. A. M. do; NETO, M. M. L.; SILVA, W. B. da. Uma aplicação didática do protocolo I2C em sistemas de comunicação. **Brazilian Journal of Development**, v. 7, n. 10, p. 94837–94853, 2021.

NOGUEIRA-NETO, P.; IMPERATRIZ-FONSECA, V. L.; KLEINERT-GIOVANNINI, A.; VIANA, B. F.; CASTRO, M. d. Biologia e manejo das abelhas sem ferrão. **São Paulo: Editora Tecnapis**, 1986.

NXP SEMICONDUCTORS. **I2C-bus specification and user manual**. Rev. 7.0. Holanda, 2021. Disponível em: https://www.nxp.com/docs/en/user-guide/UM10204.pdf. Acesso em: 14 de agosto de 2022.

OASIS MQTT TECHNICAL COMMITTEE. **MQTT Version 5.0**. USA, 2019. v. 22. Disponível em: https://docs.oasis-open.org/mqtt/mqtt/v5.0/os/mqtt-v5.0-os.html. Acesso em: 14 de agosto de 2022.

OCHOA, I. Z.; GUTIERREZ, S.; RODRíGUEZ, F. Internet of things: Low cost monitoring beehive system using wireless sensor network. In: **2019 IEEE International Conference on Engineering Veracruz (ICEV)**. Boca del Rio, Mexico: IEEE, 2019. I, p. 1–7.

OLIVEIRA, F. **Divisão de uma colônia de jupará (Melipona compressipes manaosessis) usando-se uma colmeia e o método de Fernando Oliveira**. Manaus: INPA, 2000.

OPENWRT. **About the OpenWrt/LEDE project**. 2022. Disponível em: `https://openwrt.org/about`. Acesso em: 14 de agosto de 2022.

PASSOS, G. B. **Avaliação morfométrica da identidade das espécies de abelhas mandaçaia (Melipona spp) da região da foz do rio São Francisco**. Monografia (Trabalho de Conclusão de Curso) — Universidade Federal de Sergipe, 2010.

PASTÓRIO, A.; RODRIGUES, L.; CAMARGO, E. de. Uma revisão sistemática da literatura sobre tolerância a falhas em internet das coisas. **Anais Estendidos do X Simpósio Brasileiro de Engenharia de Sistemas Computacionais**, SBC, p. 57–64, 2020.

PENIDO, E. d. C. C.; TRINDADE, R. S. **Microcontroladores**. Ouro Preto - MG: Rede e-Tec Brasil,, 2013.

PINTO, C. L.; BAMPI, A. C.; GALBIATI, C. Importância das abelhas para a biodiversidade na percepção de educandos de cáceres, mt. **Revista Ibero-Americana de Ciências Ambientais**, v. 9, n. 1, p. 152–163, 2018.

PISCALED. **Componentes Eletrônicos**. 2023. Disponível em: `https://www.piscaled.com.br/`. Acesso em: 15 de junho de 2023.

REGO, M. M. d. S. et al. **Toxicidade de agrotóxicos em abelhas africanizadas**. Dissertação (Mestrado) — Universidade Federal de Alagoas, 2022.

SANTANA, C.; ANDRADE, L.; MELLO, B.; SAMPAIO, J.; BATISTA, E.; PRAZERES, C. Teoria e prática de microserviços reativos: Um estudo de caso na internet das coisas. **Sociedade Brasileira de Computação**, 2019.

SANTOS, C. F. dos; RAGUSE-QUADROS, M.; RAMOS, J. D.; SILVA, N. L. G. da; CARVALHO, F. G. de; BARROS, C. A. de; BLOCHTEIN, B. Diversidade de abelhas sem ferrão e seu uso como recurso natural no brasil: permissões e restrições legais consorciadas a políticas públicas. **Revista Brasileira de Meio Ambiente**, v. 9, n. 2, 2021.

SANTOS, H. L.; ENDO, W.; SCALASSARA, P. R. Estudo de arquitetura em microcontroladores e definição de conceito ótimo para aplicação de beamformers em FPGA. In: **XXI Congresso Brasileiro de Automática**. Vitória: Sociedade Brasileira de Automática, 2016. ISSN 2525-8311.

SARTO, M. C. L. D. **Avaliação de Melipona quadrifasciata Lepeletier (Hymenoptera: Apidae) como polinizador da cultura do tomateiro em cultivo protegido**. Dissertação (Mestrado) — Universidade Federal de Viçosa, 2005.

SCHUWART, R. N. G.; SIMIONI, R. L.; SCODELER, A.; EMBOABA, B. C. F.; MACHADO, F. C. L.; MARTINS, H. M.; FARIA, J. de P.; ELIAS, K. E. de M. Sem abelha, sem alimento: A morte dos polinizadores por contato com os agrotóxicos. **Ratio Juris. Revista Eletrônica da Graduação da Faculdade de Direito do Sul de Minas**, v. 2, n. 2, p. 127–131, 2019.

SENSIRION SMART SENSORS SOLUTION. **Datasheet SHTC3 - Humidity and Temperature Sensor IC**. Estados Unidos, 2021. Disponível em: https://sensirion.com/media/documents/643F9C8E/6164081E/Sensirion_Humidity_Sensors_SHTC3_Datasheet.pdf. Acesso em: 14 de agosto de 2022.

SHAGHAGHI, N.; LIANG, L.; YABE, Y.; LAMA, S.; MAYER, J.; FERGUSON, P. Identifying beehive frames ready for harvesting. In: **2019 IEEE Global Humanitarian Technology Conference (GHTC)**. Seattle, WA, USA: IEEE, 2019. p. 1–4.

SHENG, Z.; MAHAPATRA, C.; ZHU, C.; LEUNG, V. C. M. Recent advances in industrial wireless sensor networks toward efficient management in IoT. **IEEE Access**, v. 3, p. 622–637, 2015.

SIKIMIC, M.; AMOVIC, M.; VUJOVIC, V.; SUKNOVIC, B.; MANJAK, D. An overview of wireless technologies for IoT network. In: **2020 19th International Symposium INFOTEH-JAHORINA (INFOTEH)**. East Sarajevo, Bosnia and Herzegovina: IEEE, 2020. p. 1–6.

STALLINGS, W. **Cryptography and Network Security: Principles and Practice**. 8th. ed. United Kingdom: Pearson, 2020. ISBN 9780135764213.

STOKSTAD, E. The case of the empty hives. **Science**, American Association for the Advancement of Science, v. 316, n. 5827, p. 970–972, 2007.

TANI, G. **PeaZip Free Archiver**. 2014. Disponível em: https://peazip.github.io/. Acesso em: 5 de fevereiro de 2023.

TARGA, M. S.; SILVA, M. C.; CEZAR, V. R. S. Uso de microcontrolador arduino para a determinação da permeabilidade do solo. **Revista Técnica Ciências Ambientais**, v. 1, n. 1, p. 1–14, 2019.

THOMAZINI, D.; ALBUQUERQUE, P. U. B. de. **Sensores industriais: fundamentos e aplicações**. 9. ed. São Paulo: Érica, 2020.

TOLDINAS, J.; LOZINSKIS, B.; BARANAUSKAS, E.; DOBROVOLSKIS, A. MQTT quality of service versus energy consumption. In: **2019 23rd International Conference Electronics**. Palanga, Lithuania: IEEE, 2019. p. 1–4.

VENTURIERI, G. Meliponicultura i: caixa racional de criação. **Embrapa Amazônia Oriental-Comunicado Técnico (INFOTECA-E)**, Embrapa Amazônia Oriental, Belém, PA, 2004. ISSN 1517-2244.

VENTURIERI, G. C. **Criação de abelhas indígenas sem ferrão**. 2. ed. Belém, PA: Embrapa Amazônia Oriental, 2004. 60 p. ISBN 978-85-87690-76-0.

VERMESAN, O.; FRIESS, P. **Internet of things: converging technologies for smart environments and integrated ecosystems**. Denmark: River publishers, 2013. ISBN 978-87-92982-96-4.

VILLAS-BÔAS, J. **Mel de Abelhas sem Ferrão**. Brasília - DF: Instituto Sociedade, População e Natureza (ISPN), 2012. ISBN 978-85-63288-08-0.

WOLFF, L. F.; LOPES, M. d. R.; PEREIRA, F. d. M.; CAMARGO, R. C. R. de; NETO, J. V. Localização do apiário e instalação das colméias. **Embrapa Meio-Norte-Documentos (INFOTECA-E)**, Teresina: Embrapa Meio-Norte, 2006.

WORTMANN, F.; FLÜCHTER, K. Internet of things. **Business & Information Systems Engineering**, Springer, v. 57, n. 3, p. 221–224, 2015.

YOUSEFPOUR, A.; FUNG, C.; NGUYEN, T.; KADIYALA, K.; JALALI, F.; NIAKANLAHIJI, A.; KONG, J.; JUE, J. P. All one needs to know about fog computing and related edge computing paradigms: A complete survey. **Journal of Systems Architecture**, Elsevier, v. 98, p. 289–330, 2019.

YUSOF, Z. M.; BILLAH, M. M.; KADIR, K.; ALI, A. M. M.; AHMAD, I. Improvement of honey production: A smart honey bee health monitoring system. In: **2019 IEEE International Conference on Smart Instrumentation, Measurement and Application (ICSIMA)**. Kuala Lumpur, Malaysia: IEEE, 2019. p. 1–5.

ZABASTA, A.; KUNICINA, N.; KONDRATJEVS, K.; RIBICKIS, L. IoT approach application for development of autonomous beekeeping system. In: **2019 International Conference in Engineering Applications (ICEA)**. Sao Miguel, Portugal: IEEE, 2019. p. 1–6.

ZACEPINS, A. et al. Applications of bee hive temperature measurements for recognition of bee colony state. In: [LLU]. **International Scientific Conference: Applied Information and Communication Technologies, 5, Jelgava (Latvia), 26-27 Apr 2012**. Jelgava, Latvia: Latvia University of Agriculture, 2012.

ZGANK, A. Bee swarm activity acoustic classification for an IoT-based farm service. **Sensors**, Multidisciplinary Digital Publishing Institute, v. 20, n. 1, p. 21, 2020.

ZUBEN, L. G. V. **Determinantes bionômicos e eco-químicos do cleptoparasitismo de Lestrimelitta limao Smith 1863 (Hymenoptera: Apidae, Meliponini)**. Tese (Doutorado) — Universidade de São Paulo, 2012.

UTFPR
UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ

PPGI
programa de pós-graduação
em informática

@ppgicp

www ppgi-cp@utfpr.edu.br
portal.utfpr.edu.br/cp/ppgi