UNIVERSIDADE FEDERAL DE SANTA MARIA
COLÉGIO POLITÉCNICO
CURSO SUPERIOR DE TECNOLOGIA EM SISTEMAS PARA INTERNET

Giancarlo Girardon Bolzan

USO DE IOT PARA AUTOMATIZAR A COLETA DE DADOS RELEVANTES À PRODUÇÃO DE MUDAS DE MORANGO

Santa Maria, RS
2023

Giancarlo Girardon Bolzan

USO DE IOT PARA AUTOMATIZAR A COLETA DE DADOS RELEVANTES À PRODUÇÃO DE MUDAS DE MORANGO

Trabalho de Conclusão apresentado ao Curso Superior de Tecnologia em Sistemas para Internet, da Universidade Federal de Santa Maria (UFSM, RS), como requisito parcial para a obtenção do título de Tecnólogo em Sistemas para Internet.

Orientador: Prof. Dr. Daniel Lichtnow

Santa Maria, RS
2023

Giancarlo Girardon Bolzan

USO DE IOT PARA AUTOMATIZAR A COLETA DE DADOS RELEVANTES À PRODUÇÃO DE MUDAS DE MORANGO

Trabalho de Conclusão apresentado ao Curso Superior de Tecnologia em Sistemas para Internet, da Universidade Federal de Santa Maria (UFSM, RS), como requisito parcial para a obtenção do título de Tecnólogo em Sistemas para Internet.

Aprovado em 3 de fevereiro de 2023:

Daniel Lichtnow, Dr. (UFSM)
(Presidente/Orientador)

Rafael Gressler Milbradt, Dr. (UFSM)

Valmir Aita, Dr. (UFSM)

Santa Maria, RS
2023

RESUMO

USO DE IOT PARA AUTOMATIZAR A COLETA DE DADOS RELEVANTES À PRODUÇÃO DE MUDAS DE MORANGO

AUTOR: Giancarlo Girardon Bolzan
ORIENTADOR: Daniel Lichtnow

O presente trabalho se propõe a analisar as possibilidades de utilização de recursos da Internet das Coisas (IoT) no contexto do processo produtivo de mudas de morango. É introduzido o tema com uma explanação do estado atual da utilização de IoT na agricultura brasileira, situando a diferença entre o cultivo do morango – normalmente realizado por pequenos produtores rurais – e outros tipos de cultivo mais voltados a produção em larga escala. É conceituada a IoT e são apresentadas as dificuldades quanto a sua adoção em zonas rurais, além de explicar como essa adoção pode beneficiar a produção das mudas de morango por meio da utilização de sensores para obter dados. É estruturado um protótipo de sistema de coleta, armazenamento e apresentação de dados relevantes para a produção de mudas de morango, bem como são selecionadas tecnologias de hardware e software que possibilitem sua implementação de modo viável. É descrito o desenvolvimento do protótipo e apresentado seu funcionamento e considerações sobre trabalhos futuros.

Palavras-chave: IoT. Morango. Sensor.

ABSTRACT

IOT USE TO AUTOMATE THE COLLECTION OF RELEVANT DATA FOR STRAWBERRY SEEDLINGS PRODUCTION

AUTHOR: Giancarlo Girardon Bolzan
ADVISOR: Daniel Lichtnow

This paper proposes an analysis of possible uses for Internet of Things (IoT) resources when contextualized for the production process of strawberry seedlings. This topic is introduced with an explanation of the current state of IoT use in Brazilian agriculture, situating the difference between strawberry cultivation – normally carried out by small farmers – and other types of cultivation more focused on large-scale production. IoT is conceptualized and the difficulties regarding its adoption in rural areas are presented, additionally explaining how this adoption can benefit the production of strawberry seedlings through the use of sensors to obtain data. A prototype system is structured consisting in collecting, locally storing and presenting relevant data for the production of strawberry seedlings, and hardware and software technologies that may allow its implementation in a viable way are selected. The development of the prototype is described, its operation is presented, and considerations about future works are given.

Keywords: IoT. Strawberry. Sensor.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE SIGLAS

AMQP	*Advanced Message Queuing Protocol*
API	*Application Programming Interface*
BNDES	Banco Nacional de Desenvolvimento Econômico e Social
CoAP	*Constrained Application Protocol*
COM	*Communication Port*
DIL	*Dual In-Line Package*
ESP-IDF	*Espressif IoT Development Framework*
HDMI	*High-Definition Multimedia Interface*
GPIO	*General Purpose Input/Output*
HTTP	*Hypertext Transfer Protocol*
IEC	*International Electrotechnical Commission*
IETF	*Internet Engineering Task Force*
IIC	*Industry IoT Consortium*
IoT	*Internet of Things*
IIoT	*Industrial Internet of Things*
IDE	*Integrated Development Environment*
ISO	*International Organization for Standardization*
LPDDR	*Low-Power Double Data Rate*
MPDG	Ministério do Planejamento, Desenvolvimento e Gestão
MCTIC	Ministério da Ciência, Tecnologia, Inovações e Comunicações
MQTT	*Message Queuing Telemetry Transport*
NTP	*Network Time Protocol*
OS	*Operating System*
RAM	*Random-Access Memory*
REPL	*Read-Eval-Print Loop*
ROM	*Read-Only Memory*
RTC	*Real-Time Clock*
PIB	Produto Interno Bruto
SCTP	*Stream Control Transmission Protocol*
SGBD	Sistema de Gerenciamento de Banco de Dados
SoC	*System on a Chip*

TCC	Trabalho de Conclusão de Curso
TCP	*Transmission Control Protocol*
UART	*Universal Asynchronous Receiver/Transmitter*
UDP	*User Datagram Protocol*
UFSM	Universidade Federal de Santa Maria
USB	*Universal Serial Bus*
VCP	*Virtual COM Port*
W3C	*World Wide Web Consortium*

LISTA DE ABREVIATURAS

GB	Gigabyte
GHz	Giga-hertz
MB	Megabyte
MHz	Mega-hertz
KB	Kilobyte
Nº	Número
p. ex.	por exemplo

SUMÁRIO

1 INTRODUÇÃO

A IoT (Internet das Coisas, do inglês *Internet of Things*) vem a cada ano ampliando seu alcance, principalmente a partir da recente popularização de microcontroladores conectáveis a redes sem fio por meio de módulos já integrados aos mesmos. Esses, não necessitando módulos adicionais para tal, deixam de ser um fator proibitivo para a viabilidade técnica e econômica do desenvolvimento e implementação de sistemas IoT em novas áreas e/ou contextos anteriormente impossibilitados.

1.1 PROBLEMA

A agricultura é uma das grandes bases econômicas do Brasil, onde “cerca de 80% da produção nacional atende o mercado interno, representando 21% do PIB em 2016” (BNDES; MPDG; MCTIC. 2017. p. 6).

Sabendo disso, pontua-se uma diferença entre os grandes e pequenos produtores que não se resume apenas a disparidade financeira, mas também aos próprios tipos de cultivo. Como explicitado em estudo liderado pelo BNDES, enquanto a produção em grandes propriedades volta-se principalmente à exportação de grãos para o mercado externo, outras culturas – dentre as quais a hortifruticultura – são supridas pela produção em propriedades de pequeno e médio porte (que dominam cerca de 78% de toda área cultivada do Brasil) e voltam-se para o mercado interno.

É a partir e para esse contexto, então, que o presente trabalho surge e pretende voltar sua atenção. Para tal o foco será voltado à cultura do morango – típica de pequenas propriedades – e especialmente no controle de produção de mudas de morango. Ao se considerar que essa cultura é “muito sensível à falta de água, baixa umidade relativa, alta temperatura e intensidade e duração da luz” (BORTOLOZZO et al., 2007) e percebendo que essas variáveis podem ser mensuradas por sensores – e que consequentemente os dados por eles coletados podem ser transmitidos, armazenados e analisados – tal característica torna a cultura de morango propícia para a implantação de soluções de IoT que possam oferecer auxílio relevante para melhoria dos seus processos.

1.2 OBJETIVOS

1.2.1 Objetivo geral

Compreender a utilização da IoT como auxiliadora do processo produtivo de mudas de morango por meio de um sistema que obtenha dados relevantes a mesma.

1.2.2 Objetivos específicos

- Conceituar o campo tecnológico da IoT;
- Apresentar o contexto produtivo no qual o sistema será inserido;
- Definir tecnologias adequadas para tal contexto;
- Desenvolver e implementar o sistema.

1.3 JUSTIFICATIVA

Pretendendo analisar a viabilidade da IoT na agricultura brasileira, um estudo divulgado pelo BNDES em 2017 verifica que a utilização da IoT ainda não encontra-se em uma fase de completa adoção. Apesar de ser inicialmente possível para a produção em larga escala, realizada pelos maiores produtores, a IoT ainda não alcançou os pequenos e médios produtores (BNDES; MPDG; MCTIC. 2017. p. 11).

Sabendo então que os fatores ligados à capacidade de investimento podem continuar sendo impeditivos para o uso da IoT na agricultura foi tomada a decisão de focar na produção de mudas, pois uma melhora de qualidade das mesmas e uma redução de custos para essa produção tendem a desencadear benefícios ao processo de modo geral. Com isso, e aliado à possibilidade de trabalhar em conjunto com a QP Mudas, empresa da Incubadora Pulsar da Universidade Federal de Santa Maria (UFSM) focada na produção de mudas de morango, viu-se possível especificar ainda mais o trabalho para a obtenção de dados relevantes à mesma.

Foram encontrados trabalhos voltados à integração de IoT para a produção de mudas de outras culturas, ou então para produção de morangos, mas não de suas mudas. O presente trabalho dá início a utilização da IoT dentro deste contexto, procurando identificar tecnologias e necessidades, mas não explora todas as possibilidades. Não é feita análise dos dados coletados, uma vez que o tempo

disponível para realização do projeto e o volume de dados coletados não permite isso. Além disso, durante a realização do trabalho o ambiente de produção de mudas da empresa incubada passou por modificações e aperfeiçoamentos.

1.4 ESTRUTURAÇÃO DO TRABALHO

Este trabalho está dividido em cinco capítulos. O presente capítulo trata de estabelecer o contexto geral do trabalho, apontando o problema percebido, os objetivos e a justificativa quanto a sua realização, bem como esta apresentação da sua estrutura.

O capítulo 2 caracteriza a Internet das Coisas, bem como o ambiente de produção de mudas de morango. O capítulo 3 apresenta as tecnologias utilizadas. Já o capítulo 4 descreve o sistema desenvolvido e o capítulo 5 seus testes no ambiente de produção de mudas.

Finalmente, o capítulo 6 servirá como conclusão ao trabalho, detalhando ainda possibilidades identificadas para prosseguimento por meio de futuros estudos.

2 CONTEXTUALIZAÇÃO

As seguintes subseções tratarão de apresentar a IoT, descrever o ambiente no qual se pretende utilizar de seus recursos e os desafios para essa integração.

2.1 A INTERNET DAS COISAS

Ao abordar termos largamente difundidos, há a possibilidade de que os mesmos tenham adquirido diferentes significados ao longo do tempo. Deste modo, torna-se interessante buscar as primeiras aparições dos mesmos para melhor compreendê-los.

Quanto à Internet das Coisas, sua provável primeira utilização vem de uma palestra ministrada por Kevin Ashton em 1999, sendo posteriormente comentada pelo mesmo em um artigo de 2009. Neste, Ashton esclarece que sua intenção era nomear um sistema computacional que independa de intervenção humana quanto à provisão de dados. Segundo ele, “se tivéssemos computadores que soubessem tudo o que há para saber sobre as coisas – usando dados que coletaram sem nossa ajuda – poderíamos rastrear e contar tudo, reduzindo muito o desperdício, a perda e o custo.” (ASHTON, 2009, tradução nossa).

Uma ressalva, então, ao conceito como apresentado por Ashton, pode aqui ser colocada, já que em grande parte o que hoje é conhecido como IoT pelo público geral não leva em consideração a obtenção automatizada de informação, sendo mais compreendida pelo conhecimento coletivo como objetos controláveis por meio de redes sem fio. Tal afirmação refere-se principalmente no que se tange a aplicações para automação residencial (tipicamente comercializadas como componentes das “casas inteligentes”) e no setor industrial, no que já se trata de modo especializado por Internet das Coisas Industrial (IIoT) e contando com um consórcio próprio (IIC, *Industry IoT Consortium*).

Ademais, e integrando-se agora ao contexto da agricultura, percebe-se que para a mesma o foco tem sido voltado também à automação, como por exemplo em sistemas de irrigação. O escopo deste trabalho, porém, está mais ligado ao conceito inicial, pois a principal característica será a obtenção de dados, de modo que a automação que aqui se apresenta é a da coleta de variáveis, e não do controle do processo produtivo em si.

2.2 O AMBIENTE PRODUTIVO

A partir de visitas e comunicação com a equipe da QP Mudas foi possível compreender as necessidades a serem supridas pela utilização de tecnologias de IoT.

O ambiente da QP Mudas apresenta espaços separados para as várias etapas do processo produtivo das mudas de morango. São eles o jardim clonal, o ambiente de enraizamento, a aclimatização, a geração de mudas comerciais e o fechamento de lotes de mudas comerciais.

- Jardim clonal: no jardim clonal inicia-se o processo de geração de mudas. Este jardim é formado por uma série de bancadas com plantas de morango, de onde é feita a coleta das pontas de estolão. Esta coleta é feita por bancada, sendo as pontas colocadas em bandejas.

- Enraizamento e aclimatização: tão logo coletadas, as bandejas com as pontas de estolão são colocadas para enraizamento em um ambiente distinto do jardim clonal. É feito o acompanhamento e manejo das bandejas com as pontas de estolão. O enraizamento dura cerca de 5 dias, sendo depois as mudas originadas das pontas de estolão transferidas para aclimatização.

- Geração de mudas comerciais: cerca de 45 dias após o início do enraizamento, as pontas de estolão que enraizaram são transferidas para outras bandejas e outro ambiente, formando os lotes comerciais.

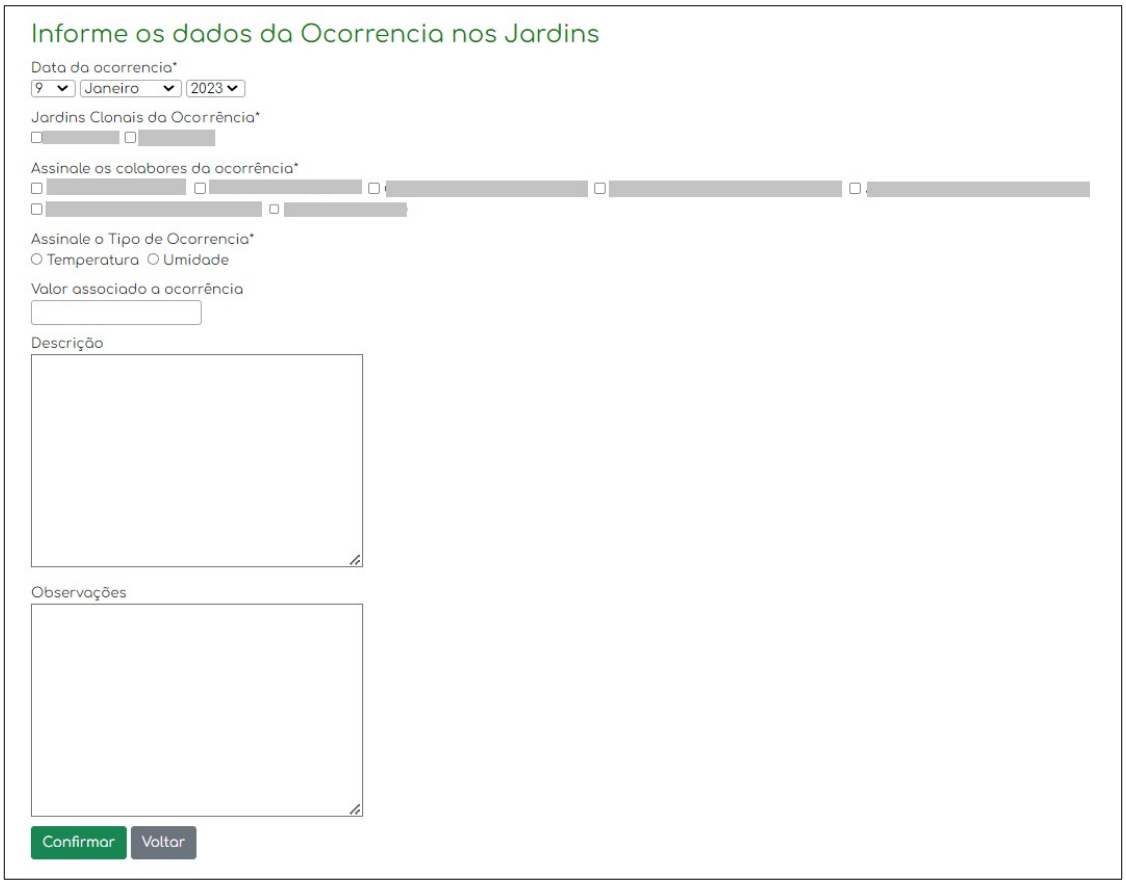
Importante salientar que nem todas pontas de estolão se tornam mudas. Existem perdas no processo, advindo disso a necessidade de manejo e controle do ambiente. Neste sentido, em todos os ambientes envolvidos no processo são considerados importantes os registros de condições relevantes, para que assim possam ser adotadas medidas de manejo necessárias de acordo com cada caso, bem como para posterior controle de perdas, possibilitando descobrir onde ocorreram as falhas no processo que as causaram. Essas condições incluem, por exemplo, o controle de pH da solução nutritiva utilizada no jardim clonal, e, de um modo geral, a temperatura e a umidade, sendo considerada a etapa do enraizamento a que mais requer o acompanhamento destas últimas.

Cabe aqui ressaltar que, em paralelo à realização do presente trabalho, ocorreu a inclusão de novas funcionalidades ao sistema Web da empresa, inicialmente desenvolvido em um projeto anterior. Estas funcionalidades não foram desenvolvidas no presente trabalho, mas pela coordenação do projeto SISTEMA

PARA APOIO À COMERCIALIZAÇÃO E PRODUÇÃO DE MUDAS DE MORANGO (registrado na UFSM com o número 056256) a partir da interação com os técnicos da QP Mudas.

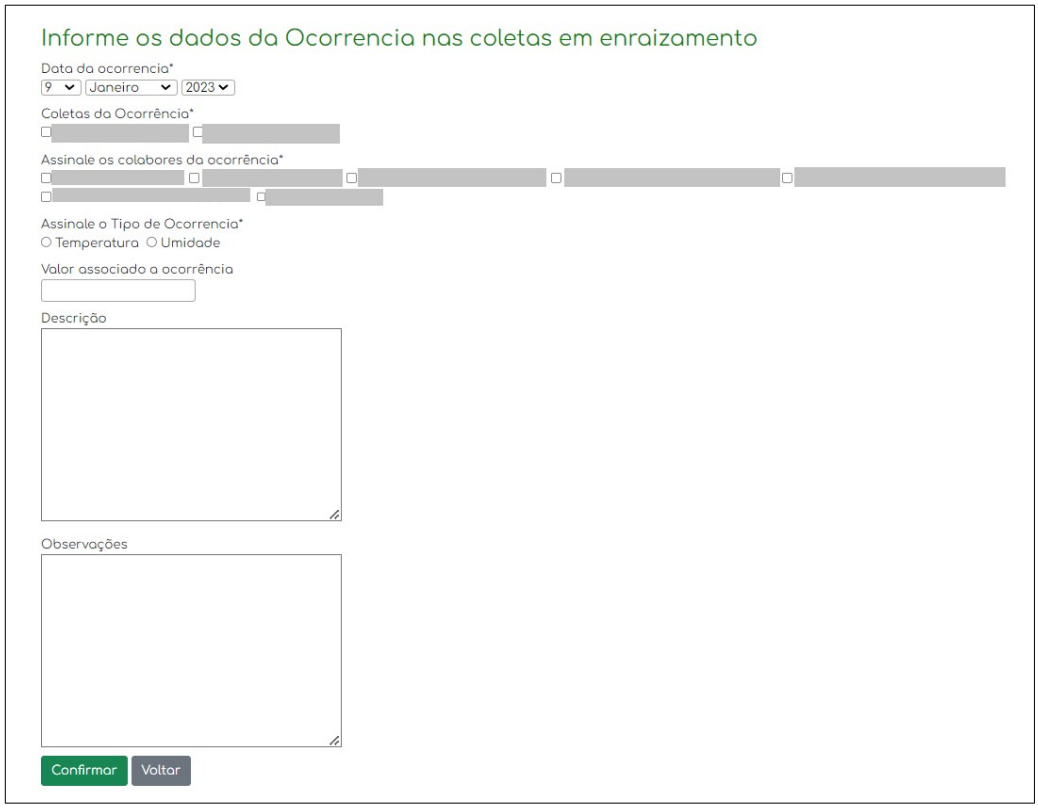
Dentre essas funcionalidades, que envolvem todo processo de geração de mudas, está a capacidade de cadastrar a ocorrência de condições em cada um dos ambientes/etapas desse processo. As Figuras 1, 2 e 3 mostram a interface para realização deste registro sem a automatização gerada pelo uso de sensores, havendo a exigência que algum colaborador realize o cadastro da ocorrência. Com os sensores, parte destas ocorrências podem passar a ser registradas no sistema de forma automática. Desse modo, tornam-se possíveis posteriores estudos dos dados coletados, a fim de detectar quais são as condições mais relevantes para a produção de mudas de alta qualidade, considerando sob que condições as mudas foram geradas.

Figura 1 – Cadastro de ocorrência nos jardins



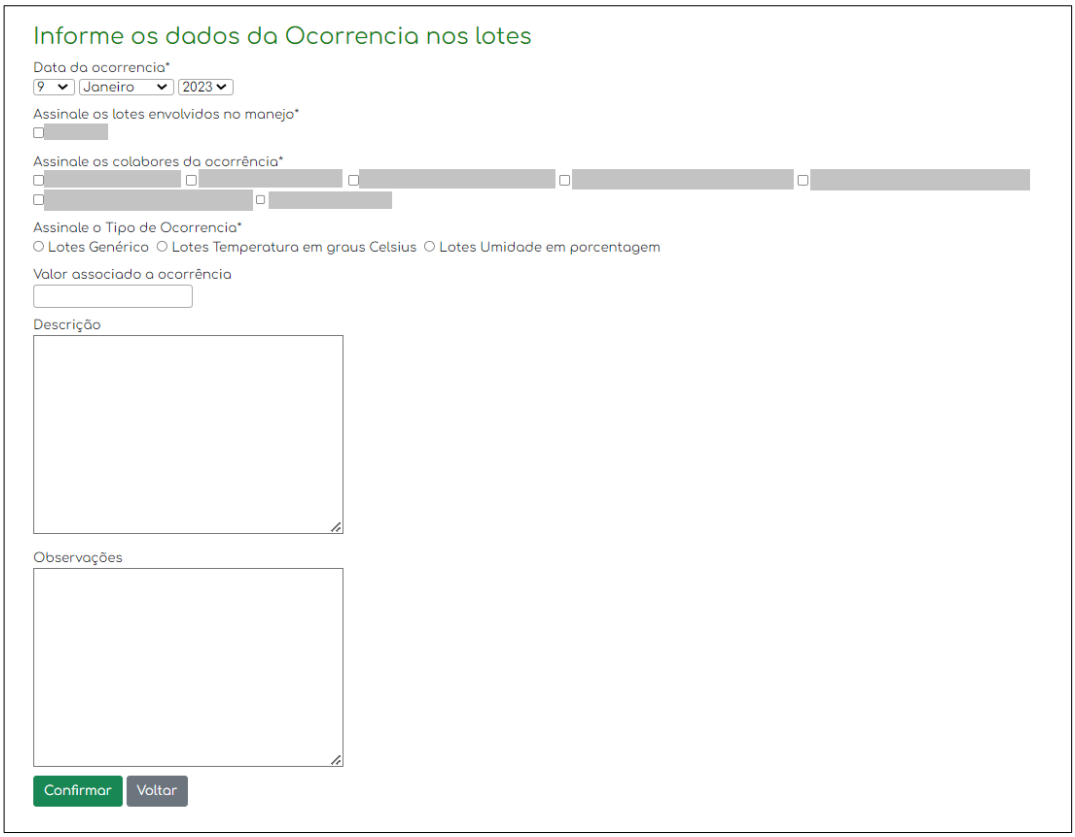
Fonte: (AUTOR, 2023)

Figura 2 – Cadastro de ocorrência no enraizamento



Fonte: (AUTOR, 2023)

Figura 3 – Cadastro de ocorrência nos lotes



Fonte: (AUTOR, 2023)

Assim, percebe-se aqui que a capacidade da IoT para obtenção de dados pode ser de grande valia, principalmente para o caso das variáveis de temperatura e umidade, das quais sensores podem manter constante monitoramento.

Houve a decisão tomada em conjunto com os responsáveis pela QP Mudas de realizar uma leitura por hora, pois seria suficiente e possibilitaria a execução dos sistema por um maior período. Cabe aqui mencionar ainda o trabalho de Martins (2017) voltado à monitoração ambiental em espaços fechados, que ratificou essa decisão por utilizar-se de execuções com o intervalo de meia-hora – também relativamente alto considerando-se a possibilidade de leitura quase ininterrupta quando não há impedimento para maior consumo de energia elétrica.

2.3 OS DESAFIOS DECORRENTES DO USO DE IOT

Como anteriormente comentado, a integração dos recursos de IoT ao ambiente produtivo para obter os dados dos sensores, transmiti-los via rede e armazená-los em banco de dados gera alguns desafios, que são aqui apresentados.

2.3.1 Transmissão

Ao analisar as necessidades particulares da transmissão de dados originários de sensores, percebe-se que, por conta de seu alto volume, o protocolo HTTP (*Hypertext Transfer Protocol*) não é uma boa opção. Apesar de estar estabelecido como o padrão global da *web*, quando comparado aos protocolos MQTT (*Message Queuing Telemetry Transport*), CoAP (*Constrained Application Protocol*) e AMQP (*Advanced Message Queuing Protocol*), o HTTP "envolve a maior largura de banda e latência [e] requer mais energia e recursos do que qualquer outro protocolo" (NAIK, 2017, tradução nossa).

Quanto aos outros três, o CoAP apresenta-se como a melhor opção quanto a essas variáveis (largura de banda, latência e consumo). Porém, tal capacidade traz consigo também um ponto negativo, pois ela é advinda principalmente da sua utilização do protocolo UDP (*User Datagram Protocol*), que não garante a entrega dos pacotes, ao contrário do protocolo TCP (*Transmission Control Protocol*) utilizado por AMQP e MQTT (além do HTTP).

Tabela 1: Comparação entre os protocolos MQTT, CoAP, AMQP e HTTP

Critério	**MQTT**	**CoAP**	**AMQP**	**HTTP**
Ano	1999	2010	2003	1997
Tamanho do cabeçalho	2 Bytes	4 Bytes	8 Bytes	Indefinido
Tamanho da mensagem	Pequena e indefinida	Pequena e indefinida	Negociável e indefinida	Grande e indefinida
Padrões	OASIS, Eclipse Foundation	IETF, Eclipse Foundation	OASIS, ISO/IEC	IETF e W3C
Protocolo de transporte	TCP	UDP, SCTP	TCP, SCTP	TCP
Formato	Binário	Binário	Binário	Texto
Suporte organizacional	IBM, Facebook, Cisco, Red Hat, Amazon Web Services, entre outros	Grande comunidade web, Cisco, entre outros	Microsoft, Bank of America, Goldman Sachs, Credit Suisse	Protocolo padrão da web global

Fonte: (AUTOR, 2022)

Ademais, quanto ao diferencial que possivelmente seja o grande definidor em benefício do MQTT, ele “tem sido empregado por um grande número de organizações [e] está emergindo para ser o protocolo *de facto* para IoT” (NAIK, 2017, tradução nossa). Desse modo, e considerando-se ainda sua boa integração junto a opção de armazenamento escolhida (como será visto na seção 2.3.2), o mesmo fora selecionado por já ser amplamente utilizado e consolidado.

Quanto ao seu funcionamento, o protocolo MQTT utiliza-se do padrão de mensagens publisher-subscriber. Para explicá-lo de maneira simples, pode-se comparar com o conceito do padrão de projeto Observer (também conhecido como Publish-Subscribe) apresentado por Gamma et al. (2007), no qual:

> Os objetos-chave nesse padrão são subject (assunto) e observer (observador). Um subject pode ter um número qualquer de observadores dependentes. Todos os observadores são notificados quando o subject sofre uma mudança de estado. (GAMMA et al., 2007, p. 275).

Note-se aqui que os conceitos de subject (assunto) e de observer (observador) relacionam-se, respectivamente, ao publisher e ao subscriber. Ademais, para organizar essa comunicação o protocolo MQTT conta com um broker. Um broker se trata de um mediador que cuida da correta comunicação das mensagens do publisher aos devidos subscribers, sendo por vezes designado como um servidor por conta disso.

2.3.2 Armazenamento

Partindo também da consideração sobre a natureza e o volume dos dados obtidos por sensores foram buscadas soluções capazes de suprir a demanda de armazenamento e consultas. Desse modo, as séries temporais surgiram como claras alternativas, pois, segundo Dunning e Friedman (2014), elas são a melhor escolha para grandes quantidades de dados e consultas baseadas em intervalos de tempo.

Uma ressalva quanto a isso apresentou-se por conta da falta de suporte para dados dessa natureza em bancos de dados mais tradicionais, como descrito por Elmasri e Navathe (2019). Desse modo, e apesar do recente surgimento de tecnologias que integram séries temporais ao modelo relacional – como o TimescaleDB (TIMESCALE, 2022), base de dados de séries temporais implementada como uma extensão para PostgreSQL – preferiu-se selecionar uma alternativa especializada (descrita na seção 3.2.2), pois, como também salientado por Elmasri e Navathe (2019), o uso de sistemas de gerenciamento de série de tempo especializados no lugar de SGBDs de uso geral tem se tornado normal para gerenciar tais informações.

2.3.3 Alimentação

Quanto a fonte de alimentação para o funcionamento do sistema também surge uma dificuldade. Por conta da inexistência de tomadas dentro do ambiente produtivo localizadas em proximidade às bancadas onde encontram-se as mudas viu-se necessária a utilização de fontes portáteis de energia elétrica. Esta foi uma das dificuldades encontradas na realização do trabalho posteriormente discutida.

3 DEFINIÇÃO DAS TECNOLOGIAS

Após tomadas essas primeiras decisões conceituais fora estruturada uma base do sistema quanto aos componentes necessários para sua implementação. Partindo das propriedades desses componentes, foram então posteriormente escolhidas tecnologias concretas que suprem os requisitos.

Desse modo, estarão aqui descritas essas tecnologias (separadas em hardware e software) utilizadas, bem como alternativas possíveis analisadas porém não selecionadas, aproveitando-se também para esclarecer os fatores que levaram a tais escolhas.

3.1 DISPOSITIVOS

Dentre as categorias de dispositivos selecionados estão os sensores, os microcontroladores aos quais esses sensores estarão conectados (preferencialmente já contando com módulo Wi-Fi integrado, pois a conexão à rede é imprescindível) e um computador a ser utilizado para executar um broker MQTT, além dos periféricos básicos.

Ademais, é também necessário um roteador para estabelecer a conexão entre microcontroladores e o computador utilizado como servidor. Tal roteador não será detalhado, sendo que a única particularidade indispensável para o mesmo é a possibilidade de definir um IP estático para o computador para que o mesmo seja identificável pelos microcontroladores.

Ainda sobre os microcontroladores – ou, de modo mais específico, dispositivos baseados em microcontroladores – será aqui destinado um espaço inicial para rapidamente elucidar quanto a sua definição, que, segundo Tollervey (2017):

> Microcontroladores são computadores encolhidos em um único chip muito pequeno. […] Esses dispositivos são muito diferentes de outros tipos de computador. A maioria dos computadores contém muitas partes: memória, armazenamento e processamento são componentes fisicamente separados contendo vários chips especializados. Eles também podem conter partes adicionais para recursos de som, gráficos e rede. Esses computadores são significativamente mais poderosos do que os dispositivos de recursos limitados baseados em microcontroladores. (TOLLERVEY, 2017, p. 1, tradução nossa)

3.1.1 Dispositivo ESP32

Partindo dessa ideia geral do que é um microcontrolador, pode-se entender melhor o dispositivo que será utilizado.

> O ESP8266 e o ESP32 são apenas chips microcontroladores puros, em vez de dispositivos completos e amigáveis ao desenvolvedor. No entanto, você pode comprar diferentes dispositivos e placas de desenvolvimento que contenham, digamos, o ESP8266. (TOLLERVEY, 2017, p. 41, tradução nossa)

O dispositivo ESP32 utilizado neste trabalho é um desses, porém não contendo um microcontrolador do tipo ESP8266, mas sim ESP32, outro integrante da família de microcontroladores ESP desenvolvidos pela Espressif Systems, que têm como grande diferencial a possibilidade integrada de conexão Wi-Fi, e, no caso dos ESP32, também Bluetooth.

Tais dispositivos baseados em ESP32 são normalmente vendidos sem maior especificação acerca do modelo exato, sendo conhecidos genericamente como microcontroladores ESP32. Segundo Tollervey (2017), “isso torna a discussão do hardware até certo ponto problemática, embora muitas vezes haja recursos comuns entre esses dispositivos.” (TOLLERVEY, 2017, p. 41, tradução nossa)

Desse modo, a identificação das características específicas de cada modelo pode ser um tanto confusa. O modelo aqui utilizado, por exemplo, traz apenas a informação de que contém o módulo ESP32-WROOM-32. Assim, cabe a recomendação quanto a ferramenta ESP Product Selector (ESPRESSIF SYSTEMS, 2022) que pode auxiliar nestes casos, tendo sido utilizada para descobrir que o módulo ESP32-WROOM-32 contém o microcontrolador ESP32-D0WDQ6.

Apesar disso, e ao menos de modo mais técnico, não pode-se ainda dizer que o dispositivo *é* um ESP32-WROOM-32, pois módulos não incluem os conectores DIL (*Dual In-Line*), sendo que estes estão presentes no dispositivo. Assim, um modo mais tecnicamente correto de defini-lo seria dizer que se trata de um “dispositivo baseado no módulo ESP32-WROOM-32, que contém o microcontrolador ESP32-D0WDQ6, integrante da família de microcontroladores ESP32”. Sendo, porém, tal nomenclatura inconveniente e desnecessária para o escopo deste trabalho, o mesmo será aqui nomeado como dispositivo, ESP32 ou dispositivo ESP32, utilizando os outros termos apenas se necessário.

Quanto a alternativas, foram analisados outros dois dispositivos que integram conexão Wi-Fi. Na Tabela 2 a seguir podemos perceber que, quando comparado a Arduino UNO WiFi Rev 2 e Raspberry Pi Pico W, o ESP32-WROOM-32 é superior em todos os aspectos. Ressalta-se aqui que os dados apresentados consideram o ESP32-WROOM-32 como base, pois algumas das características listadas variam entre diferentes dispositivos ESP32.

Além disso, os ESP32 são mais facilmente encontrados, sendo que tal fator fora também preponderante na escolha. Porém, é importante ressaltar que ambos os outros são boas alternativas para projetos que por eles possam ser supridos, principalmente caso encontrados a preços correspondentes às suas características.

Tabela 2 – Comparação entre ESP32-WROOM-32, UNO WiFi Rev 2 e Pico W

Critério	**Espressif ESP32-WROOM-32**	**Arduino UNO WiFi Rev 2**	**Raspberry Pi Pico W**
Núcleos	2	1	2
Ciclos por segundo	160 MHz	16 MHz	133 MHz
Wi-Fi integrado	Sim	Sim	Sim
Bluetooth integrado	Sim	Sim	Não
Memória RAM	520 KB	6 KB	264 KB
Memória flash	4 MB	48 KB	2 MB
GPIO	26	14	26

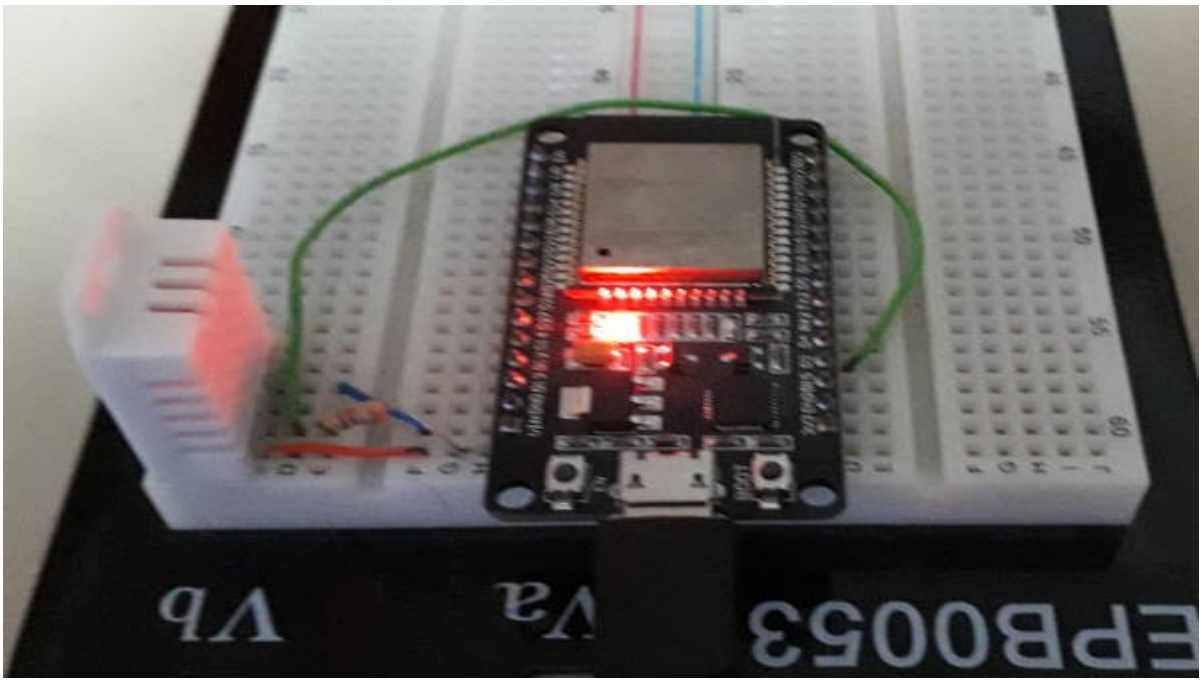
Fonte: (AUTOR, 2022)

Ademais, a Figura 1 ilustra-o ligado pelos conectores DIL a uma placa de prototipagem, também conhecida como protoboard. Por meio dela é dispensada a necessidade de soldagem para a utilização do sensor em conjunto com o dispositivo, sendo, em vez disso, ambos o dispositivo e o sensor a ela conectados.

3.1.2 Sensor DHT22

Como anteriormente citado na seção 2.2, um dos desafios quanto a produção de morango e suas mudas é o controle de temperatura e umidade, e, portanto, percebe-se que a utilização de um sensor capaz de medir ambas seria o ideal. Essa característica foi encontrada no sensor DHT22 (Figura 1, à esquerda), sendo por isso o selecionado para integrar o sistema.

Figura 4 – DHT e ESP32



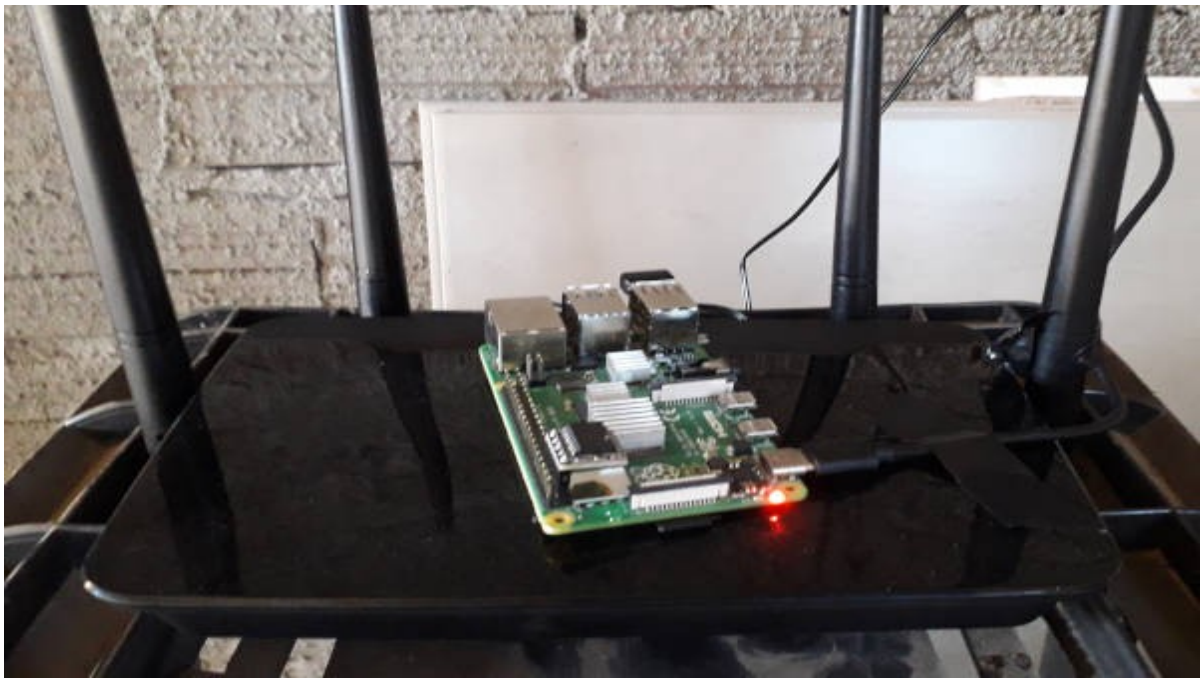
Fonte: (AUTOR, 2023)

Para a alimentação do microcontrolador e do sensor foram analisadas algumas alternativas. A primeira foi o uso de powerbank, mas foram encontrados problemas derivados disso que se tornaram impeditivos para a correta execução do sistema. Esses problemas, descritos no capítulo 5, levaram a utilização de baterias conectadas à protoboard através de um suporte, tendo sido esta a solução por fim utilizada, apesar de ocorridos problemas para o seu uso também explicitados no mesmo capítulo 5.

3.1.3 Raspberry Pi 4 Model B

O Raspberry Pi 4 Model B é um computador de placa única (SoC, *System on a Chip*) desenvolvida pela Raspberry Pi Foundation. Trata-se de uma solução de bom custo-benefício para ser utilizada como servidor em projetos de escala local. A ela foi ainda integrado um módulo de relógio de tempo real (RTC, *Real-Time Clock*) para a manutenção de data e hora mesmo em caso de quedas de energia. Tal inclusão foi necessária pois a placa não possui um RTC integrado, normalmente dependendo da sincronização de data e hora com um servidor de NTP (*Network Time Protocol*) ao iniciar, sendo isso impossibilitado por conta do sistema rodar em uma rede estritamente local.

Figura 5 – Raspberry Pi 4 Model B acima do roteador utilizado



Fonte: (AUTOR, 2022)

Não foram encontradas outras opções de fácil obtenção no país e de custo-benefício comparável, sendo a mais próxima encontrada a ASUS Tinker Board 2. Ademais, uma possibilidade seria o uso de um computador padrão, sendo apenas desejável um baixo consumo de energia para ser mantido ligado continuamente.

Tabela 3 – Comparação entre Raspberry Pi 4 Model B e ASUS Tinker Board 2

Critério	**Raspberry Pi 4 Model B**	**ASUS Tinker Board 2**
Núcleos (e ciclos por segundo)	4 (1.5 GHz)	2 (1.5 GHz), 4 (2.0 GHz)
Wi-Fi integrado	Sim (2.4 / 5.0 GHz)	Sim (2.4 / 5.0 GHz)
Bluetooth integrado	Sim (5.0)	Sim (5.0)
Memória RAM	4 GB (LPDDR4)	4 GB (LPDDR4)
Entradas USB	2 (2.0), 2 (3.0)	3 (3.2), 1 (3.2 Tipo C)

Fonte: (AUTOR, 2022)

3.2 SOFTWARES

Quanto aos softwares utilizados, uma característica comum a todos é estarem disponíveis gratuitamente por meio de licenças de código aberto. Ressaltada tal característica, estarão aqui descritos brevemente estes softwares, explicitando seu papel no sistema.

3.2.1 Mosquitto

Mosquitto (ECLIPSE FOUNDATION, 2022) é um broker de mensagens que implementa o protocolo MQTT, sendo um dos brokers MQTT mais conhecidos e utilizados.

3.2.2 InfluxDB e Telegraf

Estabeleceu-se a utilização do InfluxDB (INFLUXDATA, 2022) após ser definida a utilização de um banco de dados especializado em séries temporais, como anteriormente comentado na seção 2.3.2.

O InfluxDB apresentou-se como uma alternativa de ampla utilização para a IoT e que já está há mais tempo consolidada em relação a seus concorrentes, sendo isso evidenciado por sua colocação em 1º lugar no ranking de popularidade de bases de dados de séries temporais do site DB-Engines desde janeiro de 2016 até o presente momento. Desse modo, tal escolha fora balizada por também oferecer maior garantia de suporte de comunidade e facilidade de acesso à documentação quando necessário.

Já o Telegraf (INFLUXDATA, 2022) pode ser descrito como um facilitador do processo de coleta de métricas. Trabalhando em conjunto ao InfluxDB, o Telegraf trata de destinar os dados coletados a ele, garantindo uma integração fluida a dados transmitidos de diversos modos, incluindo por meio do protocolo MQTT.

3.2.3 MicroPython

O MicroPython (GEORGE ROBOTICS LIMITED, 2022) é uma implementação da linguagem de programação Python 3 destinada principalmente à execução de códigos em microcontroladores – nos quais seus firmware é instalado – incluindo apenas um subconjunto da biblioteca padrão do Python de modo a otimizar sua utilização em ambientes restritos.

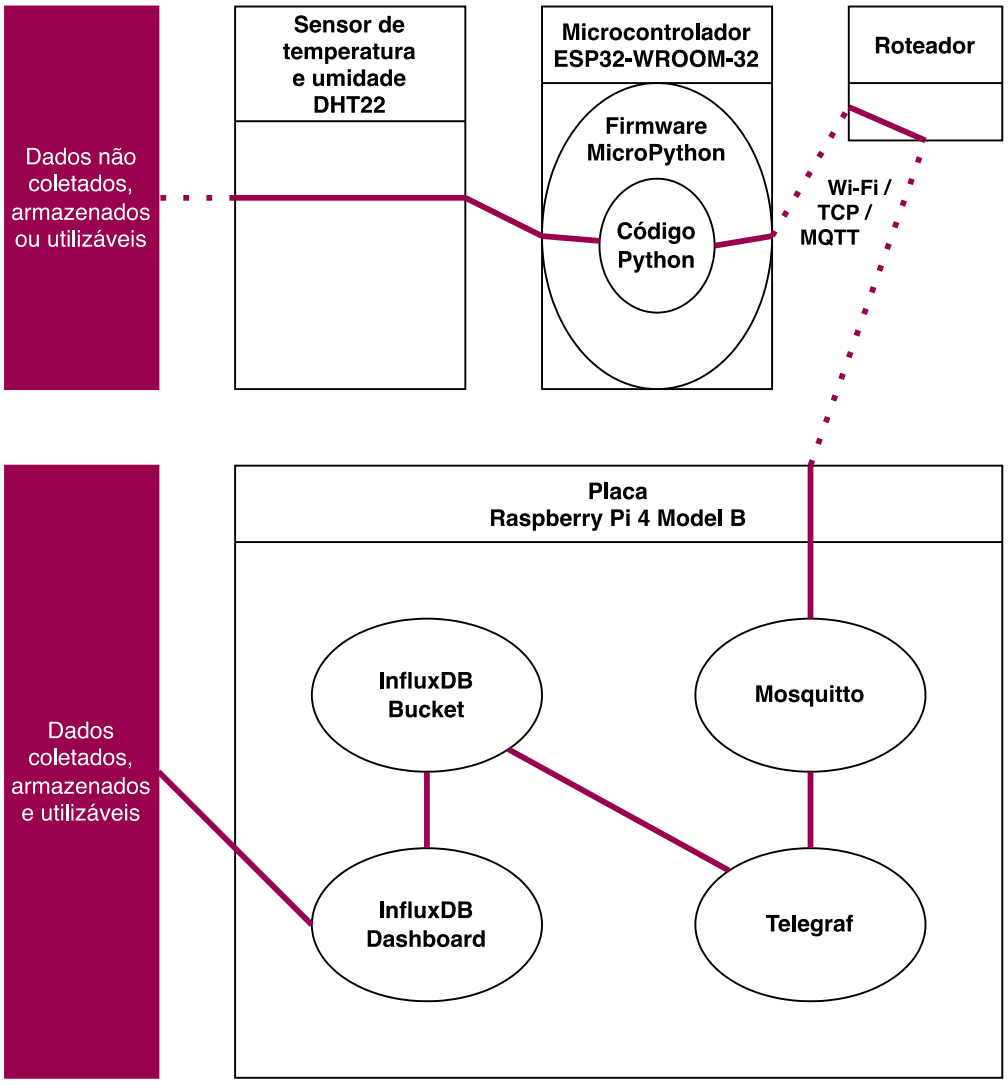
Além disso, para instalar o firmware é necessária a utilização do Esptool.py (ESPRESSIF SYSTEMS, 2022), sendo esse um utilitário escrito em Python que permite a comunicação com o bootloader (carregador de inicialização) da ROM (*Read-Only Memory*) nos microcontroladores da Espressif Systems.

4 DESENVOLVIMENTO DO SISTEMA

Após definidos os hardwares e softwares, pode ser iniciada a explicação quanto a implementação do sistema. Desse modo, estarão aqui registradas as atividades realizadas, não adentrando, porém, em minúcias da instalação e configuração, que tendem a mudar conforme novas versões e ferramentas surjam. Por conta disso, serão referenciadas quando julgadas necessárias as documentações utilizadas como base para tal, além de pontuais considerações.

A Figura 6 mostra a arquitetura simplificada do sistema destacando seus componentes e o fluxo dos dados.

Figura 6– Diagrama do sistema



Fonte: (AUTOR, 2022)

Na Figura 6 foram representados os hardwares como retângulos e os softwares como elipses, além das transmissões sem fio estarem identificadas por linhas pontilhadas, diferenciando-as de transmissões via cabo e/ou entre softwares e hardwares. Vale também ressaltar que a Figura 6 serve apenas para melhor exemplificação da estrutura, não detalhando alguns componentes ou mesmo omitindo outros – como os firmwares dos sensores e o Raspberry Pi OS – de modo a focar no entendimento geral.

4.1 MICROCONTROLADOR

Na arquitetura mostrada na Figura 6 constata-se que a viabilização da coleta dos dados ocorre a partir dos sensores conectados ao ESP32. Por meio de uma porta serial virtual (VCP, *Virtual COM Port*) que possibilita a conexão ao microcontrolador, ligando-se sua entrada micro-USB tipo B a entrada USB tipo A do computador, utilizou-se as ferramentas e documentações disponíveis para o ESP32 (ESPRESSIF, 2022), e do firmware MicroPython para realizar a sua instalação no dispositivo.

Com isso, pode-se então executar códigos Python na placa, facilitando o processo de desenvolvimento por disponibilizar um REPL (*Read-Eval-Print Loop*, “laço de leitura, execução e impressão”), o que permite a realização de testes mais rapidamente. Para a edição e execução dos códigos utilizou-se da IDE Thonny, que oferece de modo prático a possibilidade de selecionar o interpretador Python que está executando na ESP32 como destino de execução do código nela desenvolvido.

O código desenvolvido está disponível no Apêndice A. No corpo deste trabalho estarão comentadas apenas algumas linhas que contenham informações mais relevantes, iniciando-se por algumas bibliotecas utilizadas (1):

```
from machine import Pin, unique_id, deepsleep
from network import WLAN, STA_IF
from umqtt.simple import MQTTClient
import dht
```
(1)

Apesar de contar com apenas um subconjunto das bibliotecas padrão do Python 3 (PYTHON SOFTWARE FOUNDATION, 2022), fica evidente um grande benefício da utilização do MicroPython por incluir entre as suas bibliotecas algumas outras extremamente necessárias para projetos IoT, tendo como exemplos mais claros a biblioteca “network”, para possibilitar a conexão a redes, e a biblioteca “MQTTClient”, que possibilita a publicação e subscrição a tópicos MQTT, sendo aqui representado pela execução da função “publish” a partir do objeto “cliente”:

```
temperatura = str(dht_sensor.temperature()).encode()
cliente.publish(b"temperatura",
    b"bancada_b temperatura=" + temperatura)
```
(2)

Um detalhe presente em (2) que é interessante de ser lembrado é a conversão dos dados a serem transmitidos para binário, pois esse é o formato utilizado pelo protocolo MQTT, diferentemente do protocolo HTTP. Isso ocorre tanto por meio da inserção do caractere “b” ao identificar por uma *string* o tópico MQTT ao qual a mensagem se destina como pela utilização da função “encode” para conversão da temperatura obtida, após ser inicialmente transformada em uma *string* pelo construtor “str”.

Informa-se aqui que, de modo geral, quanto a implementação do Python “[…] a maioria dos recursos do MicroPython são idênticos […]” (GEORGE et al, 2022, tradução nossa). Desse modo, explica-se o motivo pelo qual detalhes mais técnicos aqui presentes referenciam a documentação do Python e não do MicroPython.

Retornando um pouco à seleção de bibliotecas importadas, cabe ressaltar a a utilidade da biblioteca “machine” que contém o método “deepsleep”. Com ele, o dispositivo entra em um estado de espera, diminuindo o consumo de energia, o que é necessário para aumentar autonomia do sistema. Ademais, ressalta-se também a presença da biblioteca “dht” que facilita a obtenção dos dados captados pelo sensor DHT22.

4.2 ROTEADOR

Passando para o contexto geral do sistema, percebe-se que apesar de ter o código sendo executado e ser possível enviar os dados para um broker público, a maior probabilidade de não ser possível conectar-se a internet em uma área rural remota ressalta a necessidade de um broker local. Desse modo, deve-se ao menos criar uma rede sem fio local, o que pode ser resolvido com o uso de um roteador comum. Com isso, fora necessário apenas configurá-lo para que a máquina na qual o broker executará receba um IP estático, sendo que os passos para tal variam de modelo para modelo, não fazendo parte do escopo geral desse trabalho.

4.3 BROKER

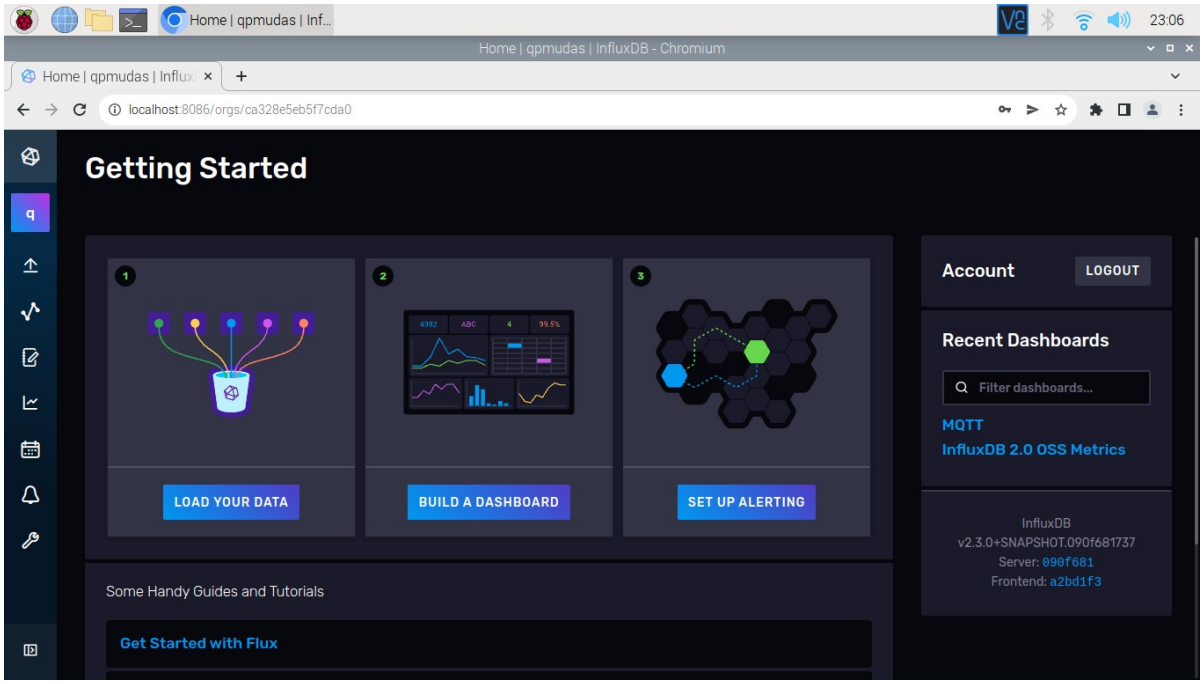
Quanto a Raspberry Pi 4, por meio do programa Raspberry Pi Imager é possível instalar o Raspberry Pi OS em um cartão microSD, inserindo-o então na entrada localizada na parte inferior da placa. Essa pode então ficar conectada a um monitor por meio de um cabo HDMI e um adaptador para micro HDMI (tipo D), além da fonte de alimentação conectada por USB tipo C. Após realizados os procedimentos pós primeira inicialização deve ser instalado o broker MQTT Mosquitto, disponibilizado no próprio repositório do Raspberry Pi OS, e configurado conforme sua documentação, que inclui a criação de um arquivo contendo as senhas para os clientes MQTT permitidos e outro de configurações gerais no local padrão para definição de opções personalizadas, sendo que nele também estará indicado o local do arquivo de senhas. Resolvidos esses detalhes, a linha de comando do Thonny já deve imprimir na tela as atualizações de estado.

4.4 RECEPÇÃO E ARMAZENAMENTO

Realizadas as instalações de InfluxDB e Telegraf, foram seguidos os passos iniciais para a utilização do InfluxDB, bem explicados pela interface gráfica do mesmo que fica disponível por padrão por meio da porta 8086 (acessível a partir de um navegador, na Raspberry Pi pelo endereço “localhost:8086” ou em outro computador da rede pelo IP estático definido para a Raspberry Pi, p. ex. “192.168.0.42:8086”).

Por conta da integração de dashboards tal como as presentes no Node-Red, essa interface assemelha-se a um misto do mesmo com interfaces gráficas para gerenciadores de bancos de dados tradicionais. Além disso, uma particularidade a mais é a utilização de algumas nomenclaturas diferentes, sendo usado, por exemplo, bucket (“balde”) em vez de um simples database (“base de dados”). Ademais, e apesar desses detalhes, o InfluxDB foi considerado, no geral, de fácil uso e compreensão.

Figura 7 – Página inicial da interface gráfica do InfluxDB



Fonte: (AUTOR, 2022)

Sobre a integração com o Telegraf, o arquivo de configuração padrão do plugin para entrada de mensagens MQTT (INFLUXDATA, 2022) conta com quase todas as informações necessárias para adicionar as fontes de entrada dos dados, bem como para direcioná-los para o InfluxDB pelo seu respectivo plugin de saída. Isso facilita o processo de configuração, e, ao final, removendo-se as linhas comentadas que explicavam seu uso, o arquivo fica bem reduzido, podendo ser apresentado aqui integralmente (3):

```
[[inputs.mqtt_consumer]]
  servers = ["tcp://localhost:1883"]
  topics = ["temperatura", "umidade"]
  topic_tag = "esp32"
  username = "NOME DE USUÁRIO MQTT"
  password = "SENHA MQTT"
  data_format = "influx"
[[outputs.influxdb_v2]]
  urls = ["http://localhost:8086"]
  token = "$INFLUX_TOKEN"
  organization = "NOME DA ORGANIZAÇÃO"
  bucket = "NOME DO BUCKET"
```

(3)

Trata-se, em boa parte, de algo autoexplicável. Na primeira linha está sendo informado que o plugin de entrada (“inputs”) sendo utilizado é o “mqtt_consumer”. Desse modo, abaixo dela estão as configurações para essa entrada, contando com listas para os endereços dos brokers de onde originam os dados (“servers”) e para os tópicos que devem ser escutados (“topics”). Ademais, é ainda digna de nota a utilização de “data_format” como “influx” para indicar que os dados que serão recebidos estão sendo enviados seguindo a sintaxe do protocolo Line (INFLUXDATA, 2023)

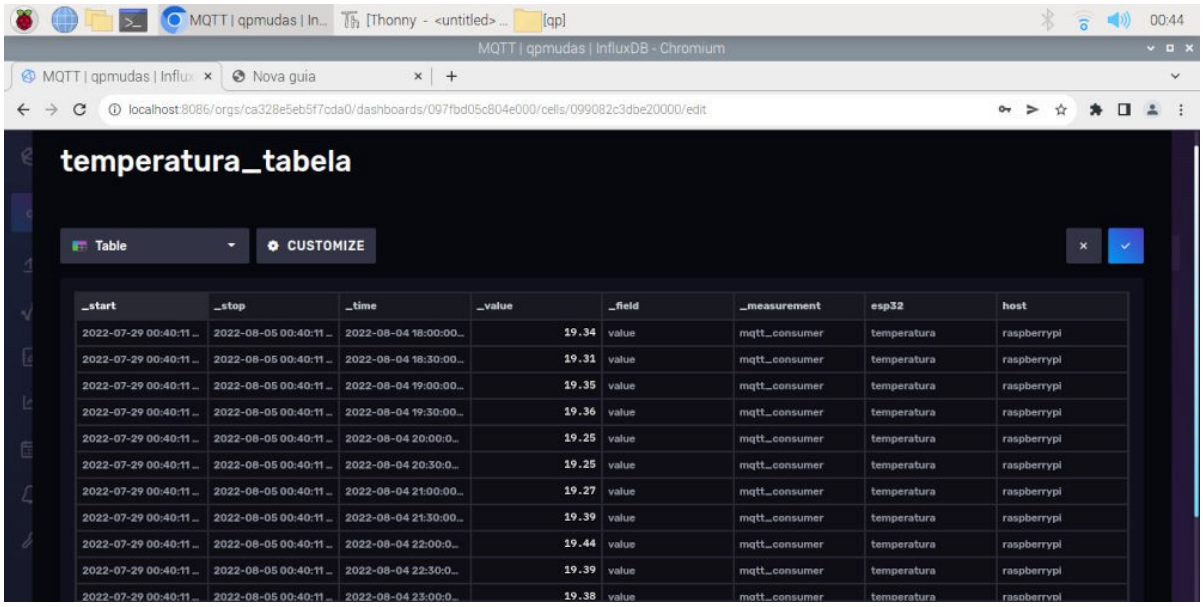
Já “[[outputs.influxdb_v2]]” define que as linhas subsequentes são relacionadas a saída dos dados para o InfluxDB. Nesse ponto ressalta-se apenas a necessidade de informar um token de acesso à API (*Application Programming Interface*) do InfluxDB para que o Telegraf possa acessá-la. Para utilizá-lo, então, pode-se simplesmente informá-lo, ou, como é recomendado pela documentação, guardar o token como uma váriavel do ambiente – por exemplo, “$INFLUX_TOKEN” – informando-a em vez do mesmo.

Chegando nesse ponto, os dados começam a ser armazenados. Um detalhe adicional que torna-se preponderante quando relacionado a volumes de dados como esses, que podem seguir aumentando indefinidamente, é a possibilidade de configurar o InfluxDB de forma a limitar a idade das entradas, limpando anteriores para economizar espaço. Acrescendo-se a isso o fator de que a capacidade de

armazenamento da Raspberry Pi depende, pelo menos no presente momento, do tamanho de um cartão microSD nela utilizado, pode-se começar a pensar em uma solução híbrida, como será comentado na próxima e última seção.

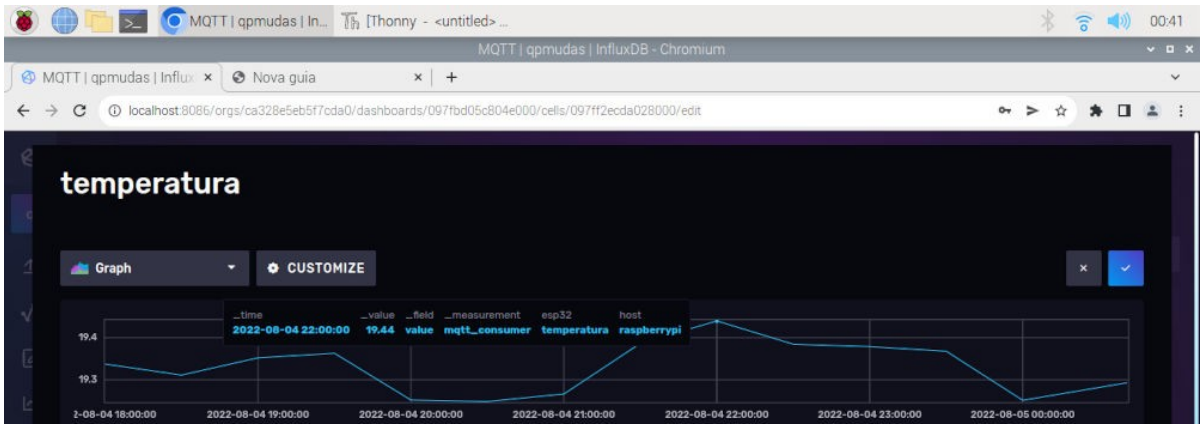
Finalmente, quanto a visualização dos dados, os dashboards do InfluxDB possibilitam diversos estilos de gráficos. Aqui, porém, estarão ilustrados um tabular (Figura 5), e outro em linha do tempo (Figura 6), que demonstra mais fidedignamente o aspecto temporal dos dados.

Figura 8 – Gráfico em tabela



Fonte: (AUTOR, 2022)

Figura 9 – Gráfico em linha do tempo



Fonte: (AUTOR, 2022)

5 IMPLANTAÇÃO NO AMBIENTE PRODUTIVO

Por fim, o sistema passou para a fase de implantação, sendo esse o ponto mais problemático do processo. Foi decidido que o ambiente a ser controlado seria o jardim clonal, até pelo fato de que a QP Mudas estava aprimorando os demais ambientes. Por conta do baixo alcance da conexão Wi-Fi dos microcontroladores foi necessário esperar algum tempo até que o roteador pudesse ser ligado em um local mais próximo aos sensores, em um pequeno prédio cuja construção estava sendo finalizada.

Após as primeiras visitas ao ambiente de produção de mudas foi percebido também que não haveria possibilidade de alimentação do microcontrolador e sensor por meio de um simples carregador, por conta da falta de tomadas próximas às bancadas do jardim clonal. Portanto, iniciou-se pela utilização um powerbank (Figura 10) com capacidade de armazenamento suficiente para a manutenção do sistema executando por longo período, além da integração de um painel solar para carregá-lo.

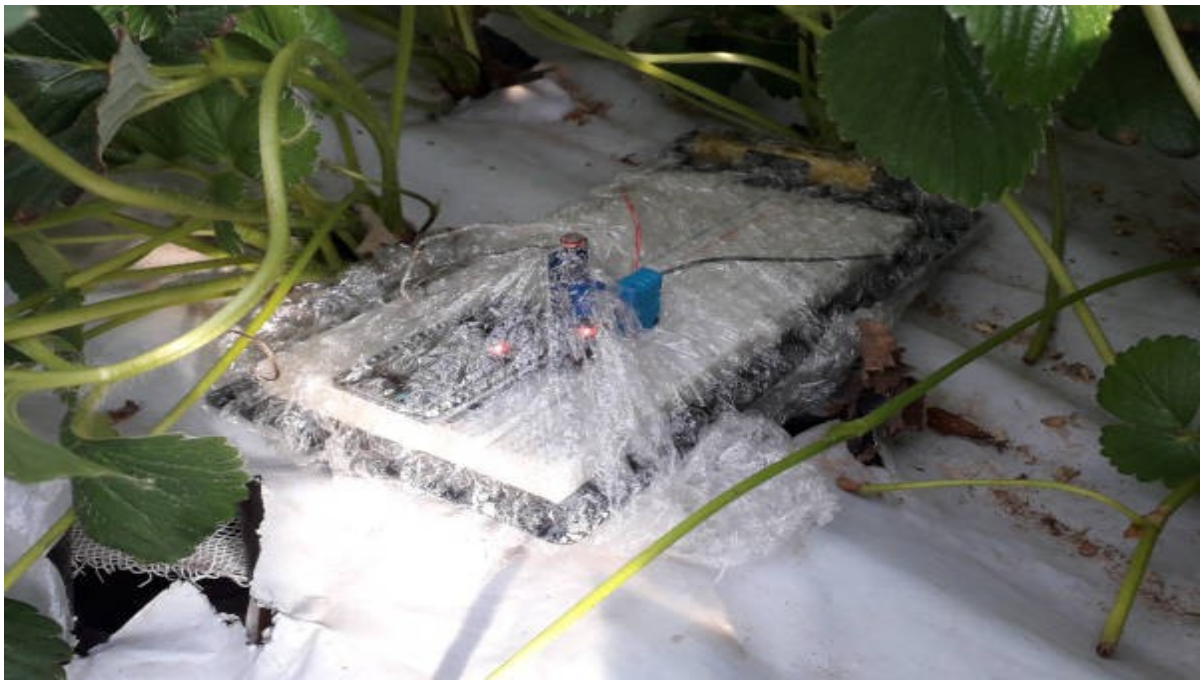
Porém, tal solução falhou por conta de outra característica útil para o uso comum do powerbank selecionado, a interrupção da alimentação no caso de não ser detectado um consumo mínimo de carga. Sendo esse exatamente o caso, já que o objetivo é consumir o mínimo possível, foi buscado como opção o uso de baterias.

Figura 10 – Powerbank com placa solar

Fonte: (AUTOR, 2023)

Novamente, porém, foram encontradas dificuldades, pois para a utilização da mesma a organização dos itens na protoboard teve de ser modificada. Ademais, viu-se necessário manter a alimentação do sensor por meio da saída de 5V do microcontrolador e não diretamente pelos 3.7V da bateria, após essa tensão mostrar-se próxima do mínimo para a alimentação do sensor, de tal modo afetando a confiabilidade de que o mesmo funcione sempre que requerido.

Figura 11 – Tentativa de implantação com bateria

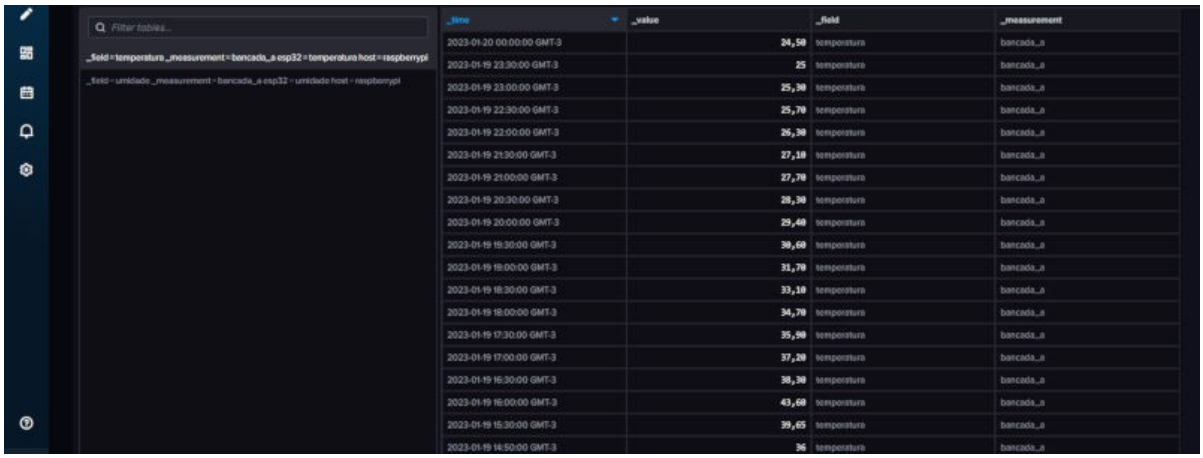


Fonte: (AUTOR, 2022)

Por conta dessas dificuldades, na última tentativa decidiu-se pelo teste de dois esquemas no ambiente produtivo, um utilizando bateria com a última configuração acima comentada e outro com um powerbank simples, sem painel solar, mas que não demonstrou possuir a mesma característica de interrompimento da alimentação.

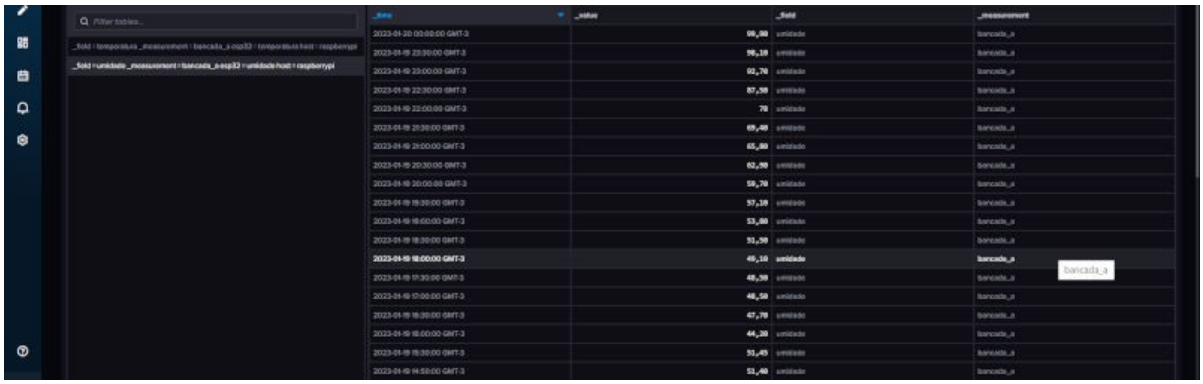
A Figura 12 e a Figura 13 mostram o dashboard com dados das últimas coletas realizadas (temperatura e umidade).

Figura 12 – Dados coletados (temperatura)



Fonte: (AUTOR, 2022)

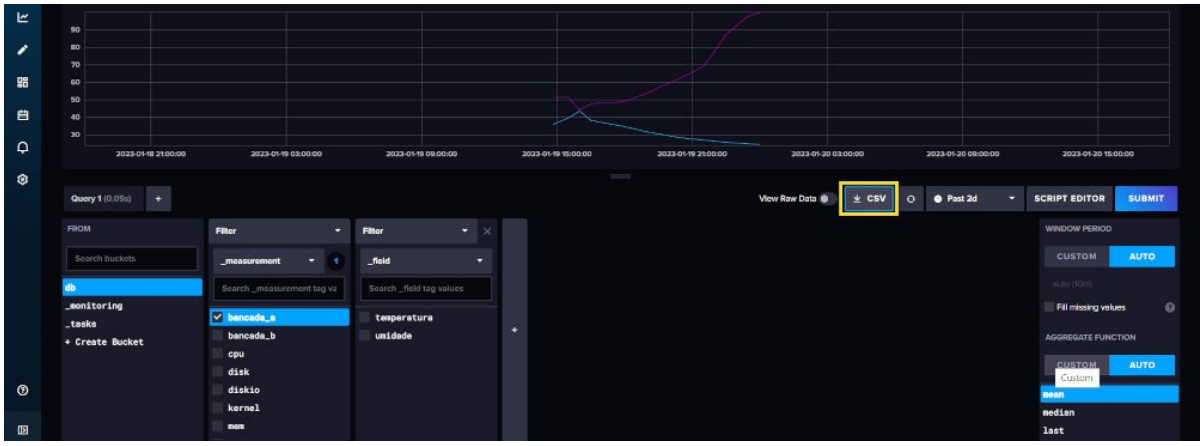
Figura 13 – Dados coletados (umidade)



Fonte: (AUTOR, 2022)

Estes dados podem ser extraídos do banco de dados gerando um arquivo CSV (Figura 14).

Figura 14 – Opção para geração de arquivo CSV



Fonte: (AUTOR, 2022)

6 CONSIDERAÇÕES FINAIS

A partir das análises realizadas quanto ao uso da IoT na produção rural, e, mais especificamente, para a produção de morangos e mudas de morango, pode-se perceber que há espaços ainda não devidamente explorados. Dentre esses está o uso de sensores para monitoramento do ambiente, capturando dados que são fatores determinantes para a cultura do morango, de modo que a IoT possa auxiliar no controle dessa cadeia produtiva.

Ademais, o estudo de tecnologias ainda emergentes, como a IoT, é sempre uma experiência que pode render bons frutos. Quando voltado, então, para um resultado mais prático, e, além disso, agregando conhecimentos de áreas diferentes – sejam não tanto, como o desenvolvimento de sistemas para internet e o uso de microcontroladores e sensores, ou mais, como entre essas e a produção de mudas de morango – a possibilidade de tal resultado ser positivo aumenta.

Desse modo, e pensando em futuros trabalhos a serem realizados, aparece como boa alternativa a inclusão de outros sensores além de temperatura e umidade, como por exemplo a intensidade da luz solar. Na outra ponta, quanto ao armazenamento, a utilização de solução híbrida, na qual os dados presentes em bases de série temporal possam ser periodicamente transferidos para bancos de dados mais tradicionais – quiçá na nuvem, caso se conte com possibilidade de acesso relativamente estável a internet – também parece ser uma opção, enquanto a visualização dos dados pode beneficiar-se da integração de sistemas mais específicos para a análise dos mesmos.

O fato do ambiente de produção da empresa estar sendo aperfeiçoado durante a realização do projeto limitou o seu alcence. Estas melhorias, especialmente as feitas no ambiente de enraizamento, poderão ser exploradas em trabalhos futuros, pois estão sendo colocados sensores para automação visando manutenção da umidade e temperatura que geram dados que poderão ser utilizados.

Salienta-se como maior dificuldade percebida a escolha e utilização de fontes de alimentação adequadas, principalmente no caso de prévio distanciamento deste tipo de prática, sendo que, caso tal adversidade fosse totalmente superada, traria como benefício a obtenção dos dados durante mais tempo ou em maior frequência, bem como possivelmente teria permitido a integração ao banco de dados existente

da QP Mudas a tempo da conclusão deste trabalho, o que não foi possível. Assim, uma alternativa que se apresenta seria a realização de trabalhos conjuntos nos quais seja possível focar de modo individual – porém ainda integrado – os diferentes componentes do sistema, que, como pode ser percebido, abarcam uma ampla área de conhecimento, tornando o tema propício para estudos envolvendo abordagem de maior âmbito interdisciplinar.

REFERÊNCIAS

ARDUINO. **Arduino UNO WiFi Rev2 Documentation**. Disponível em: https://docs.arduino.cc/hardware/uno-wifi-rev2. Acesso em: 27 jul. 2022.

ASHTON, Kevin. That ‘internet of things’ thing. **RFID journal**, v. 22, n. 7, p. 97–114, 2009.

ASUS. **ASUS Tinker Board 2 Series specifications summary**. Dez. 2020. Disponível em: https://tinker-board.asus.com/doc_tb2.html. Acesso em: 27 jul. 2022.

BERTOLETI, Pedro H. F. Controle e Monitoramento IoT com NodeMCU e MQTT. **Blog da FilipeFlop**: tutoriais, projetos e conteúdo exclusivo. 30 maio 2016. Disponível em: https://www.filipeflop.com/blog/controle-monitoramento-iot-nodemcu-e-mqtt. Acesso em: 27 jul. 2022.

BORTOLOZZO, Adriane R. *et al*. **Produção de morangos no sistema semi-hidropônico**. Bento Gonçalves: Embrapa Uva e Vinho, 2007.

BRASIL. Banco Nacional de Desenvolvimento Econômico e Social; Ministério do Planejamento, Desenvolvimento e Gestão; Ministério da Ciência, Tecnologia, Inovações e Comunicações. Produto 7C: aprofundamento de verticais – Rural. **Internet das coisas**: um plano de ação para o Brasil. Dez. 2017. Disponível em: https://www.gov.br/mcti/pt-br/acompanhe-o-mcti/transformacaodigital/arquivosinternetdascoisas/fase3_7c_relatorio-de-aprofundamento-das-verticais-rural.pdf. Acesso em: 12 jul. 2022.

DUNNING, Ted; FRIEDMAN, Ellen. **Time series databases**: new ways to store and access data. Sebastopol: O’Reilly Media Inc., 2014.

ECLIPSE FOUNDATION. Mosquitto. Disponível em: https://mosquitto.org/download. Acesso em: 27 jul. 2022.

ECLIPSE FOUNDATION. Publicly available Eclipse Mosquitto MQTT server/broker. Disponível em: https://test.mosquitto.org/. Acesso em: 27 jul. 2022.

ELMASRI, Ramez; NAVATHE, Shamkant B. **Sistemas de bancos de dados**. 7. ed. São Paulo: Pearson, 2019.

ESPRESSIF SYSTEMS. ESP Product Selector. Disponível em: https://products.espressif.com/#/product-selector. Acesso em: 27 jul. 2022.

ESPRESSIF SYSTEMS. ESP-IDF: Official IoT Development Framework. Disponível em: https://espressif.com/en/products/sdks/esp-idf. Acesso em: 27 jul. 2022.

ESPRESSIF SYSTEMS. **ESP32-WROOM-32 Datasheet**. Disponível em: https://www.espressif.com/sites/default/files/documentation/esp32-wroom-32_datasheet_en.pdf. Acesso em: 27 jul. 2022.

ESPRESSIF SYSTEMS. Esptool.py. Disponível em: https://github.com/espressif/esptool/releases/latest. Acesso em 05 ago. 2022.

ESPRESSIF SYSTEMS. Flashing firmware. **Esptool.py documentation**. Disponível em: https://docs.espressif.com/projects/esptool/en/latest/esp32/esptool/flashing-firmware.html. Acesso em 05 ago. 2022.

GAMMA, Erich; HELM, Richard; JOHNSON, Ralph; VLISSIDES, John. **Padrões de projeto**: soluções reutilizáveis de software orientado a objetos. Porto Alegre: Bookman, 2007.

GEORGE, Damien P. *et al*. Quick reference for the ESP32: OneWire driver. **MicroPython documentation**. Disponível em: https://docs.micropython.org/en/latest/esp32/quickref.html#onewire-driver. Acesso em: 06 ago. 2022.

GEORGE, Damien P. *et al*. MicroPython language and implementation. **MicroPython documentation**. Disponível em: https://docs.micropython.org/en/latest/reference/index.html. Acesso em: 06 ago. 2022

GEORGE ROBOTICS LIMITED. ESP32 Firmware. Disponível em: https://micropython.org/download/esp32. Acesso em: 27 jul. 2022.

INDUSTRY IOT CONSORTIUM. Disponível em: https://www.iiconsortium.org/about-us. Acessado em: 30 jul. 2022.

INFLUXDATA. InfluxDB. Disponível em: https://portal.influxdata.com/downloads/#influxdb. Acesso em: 27 jul. 2022.

INFLUXDATA. Line protocol. Disponível em: https://docs.influxdata.com/influxdb/v2.6/reference/syntax/line-protocol/. Acesso em: 10 jan. 2022.

INFLUXDATA. MQTT consumer input plugin: configuration. **Github**: telegraf repository. Disponível em: https://github.com/influxdata/telegraf/tree/master/plugins/inputs/mqtt_consumer. Acesso em: 06 ago. 2022.

INFLUXDATA. Telegraf. Disponível em: https://portal.influxdata.com/downloads/#telegraf. Acesso em: 27 jul. 2022.

MARTINS Pedro M. dos S. **Monitoração ambiental em espaços fechados**. 2017. Dissertação (Mestrado em Engenharia Eletrotécnica e de Computadores) – Faculdade de Ciências e Tecnologia, Universidade NOVA de Lisboa, Portugal, 2017. Disponível em: https://run.unl.pt/handle/10362/31880. Acesso em: 10 dez. 2023.

NAIK, Nitin. Choice of effective messaging protocols for IoT systems: MQTT, CoAP, AMQP and HTTP. *In* 2017 IEEE INTERNATIONAL SYSTEMS ENGINEERING SYMPOSIUM (ISSE). **Anais** […]. IEEE, 2017.

PYTHON SOFTWARE FOUNDATION. Built-in types: bytes. **Python documentation**. Disponível em: https://docs.python.org/3/library/stdtypes.html#bytes. Acesso em: 06 ago. 2022.

PYTHON SOFTWARE FOUNDATION. Built-in types: str. **Python documentation**. Disponível em: https://docs.python.org/3/library/stdtypes.html#str. Acesso em: 06 ago. 2022.

PYTHON SOFTWARE FOUNDATION. Built-in types: str.encode. **Python documentation**. Disponível em: https://docs.python.org/3/library/stdtypes.html#str.encode. Acesso em: 06 ago. 2022.

QP MUDAS. **QP Mudas – Mudas de morango**. Disponível em: https://qpmudas.com.br. Acesso em: 06 ago. 2022.

RASPBERRY PI LTD. **Raspberry Pi 4 Model B Product Brief**. Jan. 2021. Disponível em: https://datasheets.raspberrypi.com/rpi4/raspberry-pi-4-product-brief.pdf. Acesso em: 27 jul. 2022.

RASPBERRY PI LTD. Raspberry Pi OS & Raspberry Pi Imager. Disponível em: https://www.raspberrypi.com/software/. Acessado em: 05 ago. 2022.

RASPBERRY PI LTD. **Raspberry Pi Pico W Product Brief**. Jun. 2022. Disponível em: https://datasheets.raspberrypi.com/picow/pico-w-product-brief.pdf. Acesso em: 27 jul. 2022.

SILLICON LABS. CP210x USB to UART Bridge VCP Drivers. Disponível em: https://www.silabs.com/developers/usb-to-uart-bridge-vcp-drivers. Acesso em: 27 jul. 2022.

SOLID IT GMBH. Trend of time series DBMS popularity. **DB-Engines Ranking**. Disponível em: https://db-engines.com/en/ranking_trend/time+series+dbms. Acesso em: 29 jul. 2022.

THONNY. Thonny. Disponível em: https://thonny.org/. Acesso em: 27 jul. 2022.

TIMESCALE. TimescaleDB. Disponível em: https://docs.timescale.com/install/latest/. Acessado em: 04 ago. 2022.

TOLLERVEY, Nicholas H. **Programming with MicroPython**: embedded programming with microcontrollers and Python. Sebastopol: O'Reilly Media Inc., 2017

APÊNDICE A – CÓDIGO PYTHON

```
from machine import Pin, unique_id, deepsleep
from time import sleep
from network import WLAN, STA_IF
from ubinascii import hexlify
from umqtt.simple import MQTTClient
from micropython import const
import dht

PIN_DHT = const(4)
WIFI_SSID = "SSID"
WIFI_PASSWORD = "PASSWORD"
MQTT_BROKER = "000.000.0.000"
MQTT_PORT = const(1883)
MQTT_ID = hexlify(unique_id()).decode()
MQTT_USER = "USER"
MQTT_PASSWORD = "PASSWORD"

def get_connected_mqtt_client():
    wifi = WLAN(STA_IF)
    wifi.active(False)
    wifi.active(True)
    wifi.disconnect()
    wifi.connect(WIFI_SSID, WIFI_PASSWORD)
    while not wifi.isconnected():
        pass
    cliente = MQTTClient(MQTT_ID, MQTT_BROKER, MQTT_PORT,
        MQTT_USER, MQTT_PASSWORD, keepalive=65535)
    cliente.connect()
    return cliente
```

```
def main():
    try:
        cliente = get_connected_mqtt_client()
        dht_sensor = dht.DHT22(Pin(PIN_DHT))
        dht_sensor.measure()
        temperatura = str(dht_sensor.temperature()).encode()
        cliente.publish(b"temperatura",
            b"bancada_a temperatura=" + temperatura)
        umidade = str(dht_sensor.humidity()).encode()
        cliente.publish(b"umidade",
            b"bancada_a umidade=" + umidade)
        sleep(5)
    finally:
        deepsleep(3600000)

if __name__ == "__main__":
    main()
```