INSTITUTO FEDERAL DE SANTA CATARINA
CURSO DE ENGENHARIA ELÉTRICA

GUSTAVO AVELAR CABRAL

DESENVOLVIMENTO DE UM PROTÓTIPO PARA GERENCIAMENTO E CONTROLE DE UMA ESTUFA BASEADA EM INTERNET DAS COISAS

ITAJAÍ
2025

Gustavo Avelar Cabral

Desenvolvimento de um protótipo para gerenciamento e controle de uma estufa baseada em internet das coisas

Monografia apresentada ao curso de Engenharia Elétrica do Instituto Federal de Santa Catarina, para obtenção do título de Bacharel em Engenharia Elétrica.

Orientadora: Profª. Dra. Fernanda Isabel Marques Argoud da Silva

Itajaí

2025

Ficha de identificação da obra elaborada pelo autor,
através do Programa de Geração Automática da Biblioteca do IFSC.

Cabral, Gustavo Avelar
DESENVOLVIMENTO DE UM PROTÓTIPO PARA GERENCIAMENTO E CONTROLE DE UMA ESTUFA BASEADA EM INTERNET DAS COISAS / Gustavo Avelar Cabral ; orientador, Fernanda Isabel Marques Argoud da Silva, 2025.
137 p.

Trabalho de Conclusão de Curso (graduação) - Instituto Federal de Santa Catarina, Campus Itajaí, Graduação em Engenharia eletrica, Itajaí, 2025.

Inclui referências.

1. Engenharia eletrica. 2. Agricultura indoor. 3. Internet das Coisas. 4. Raspberry Pi. 5. Aplicativos móveis. I. da Silva, Fernanda Isabel Marques Argoud. II. Instituto Federal de Santa Catarina. Graduação em Engenharia eletrica. III. Título.

DESENVOLVIMENTO DE UM PROTÓTIPO PARA GERENCIAMENTO E CONTROLE DE UMA ESTUFA BASEADA EM INTERNET DAS COISAS

Gustavo Avelar Cabral

Este trabalho foi julgado adequado para obtenção do Título de Engenheiro Eletricista e aprovado na sua forma final pela banca examinadora do curso de engenharia elétrica do Instituto Federal de Educação, Ciência e Tecnologia de Santa Catarina.

Itajaí, 02 de dezembro de 2025.

BANCA EXAMINADORA:

gov.br Documento assinado digitalmente
FERNANDA ISABEL MARQUES ARGOUD DA SILV
Data: 15/12/2025 17:34:41-0300
Verifique em https://validar.iti.gov.br

Dra. Fernanda Isabel Marques Argoud da Silva
Instituto Federal de Santa Catarina – IFSC

gov.br Documento assinado digitalmente
DOUGLAS ALEXANDRE RODRIGUES DE SOUZA
Data: 11/12/2025 18:31:36-0300
Verifique em https://validar.iti.gov.br

Me. Douglas Alexandre Rodrigues de Souza
Instituto Federal de Santa Catarina – IFSC

gov.br Documento assinado digitalmente
SERGIO AUGUSTO BITENCOURT PETROVCIC
Data: 11/12/2025 17:12:37-0300
Verifique em https://validar.iti.gov.br

Dr. Sérgio Augusto Bitencourt Petrovcic
Instituto Federal de Santa Catarina - IFSC

AGRADECIMENTOS

Agradeço primeiramente à minha mãe, Sueli Avelar Mello, por todo o suporte que me foi dado durante todo o meu percurso até a minha formação. Agradeço por cada momento e esforço que dedicou a mim sem pensar duas vezes. Este trabalho é resultado do seu amor e sacrifício.

A Deus, por ter me dado direcionamento nos momentos em que mais precisei, e à minha família, que em diversas vezes me deu apoio fundamental nessa jornada.

À minha namorada, Geovana Spack, por dividir todos os momentos que passei durante essa jornada, me apoiando em cada decisão e me fazendo uma pessoa melhor a cada dia. Obrigado por tudo.

Aos meus amigos Vinícius Hercilio Correa e Leonir José da Costa Júnior. Durante nossa formação passamos por muita coisa juntos: muitas horas de estudo, diversas madrugadas viradas, muito café e diversos momentos incríveis. Foi uma honra finalizar este curso com vocês.

Ao Instituto Federal de Santa Catarina - Campus Itajaí, por oferecer este curso e toda a infraestrutura de qualidade que foi fundamental para minha formação acadêmica e profissional.

"Os discípulos não são maiores que seu mestre. Mas o aluno bem instruído será como o mestre." (Lucas 6:40)

RESUMO

O presente trabalho apresenta o desenvolvimento de um protótipo de estufa para ambientes internos baseada em Internet das Coisas (IoT), projetado para monitorar e controlar variáveis ambientais críticas ao cultivo. O sistema foi estruturado com sensores de luminosidade, temperatura e umidade do ar e do solo, interligados a um microcontrolador Raspberry Pi 4 responsável pela centralização dos dados e acionamento dos atuadores, compostos por ventoinhas, luminária de cultivo, resistor de aquecimento e bomba peristáltica para irrigação. A arquitetura de *software*, implementada com o Firebase, plataforma de desenvolvimento de aplicativos móveis e *web* do Google, assegurou o armazenamento de dados e a comunicação em tempo real, possibilitando a integração entre o protótipo físico e o aplicativo móvel. Esse aplicativo, desenvolvido em React Native, *framework* para criação de aplicativos móveis com JavaScript, utilizando a plataforma Expo, viabilizou a visualização e o gerenciamento da estufa em tempo real, com configuração de variáveis desejadas, calibração de atuadores e acompanhamento por meio de gráficos de tendências. Os testes contemplaram a validação das principais funcionalidades do sistema, incluindo o controle do fotoperíodo da luminária (tempo de exposição à luz artificial em um ciclo de 24 horas), a regulação térmica por meio do aquecedor e das ventoinhas, a verificação do controle de umidade do ar e a avaliação da irrigação automatizada com bomba peristáltica. Além desses ensaios individuais, foi realizado um ciclo experimental de três dias, correspondendo às diferentes fases do cultivo, a fim de avaliar o funcionamento integrado da estufa em operação contínua. Durante todos os ensaios, o sistema efetuou a coleta contínua de dados de luminosidade, temperatura e umidade do ar e do solo, registrando-os em tempo real e possibilitando seu acompanhamento por meio de gráficos de tendência nos intervalos de 1 hora e 24 horas. Essa abordagem permitiu verificar tanto a consistência das medições quanto a resposta automática dos atuadores frente às variações ambientais simuladas e ao comportamento do ciclo de cultivo.

Palavras-chave: Agricultura *indoor*; Internet das Coisas; Raspberry Pi; Aplicativos móveis; Banco de dados.

ABSTRACT

This work presents the development of a greenhouse prototype for indoor environments based on the Internet of Things (IoT), designed to monitor and control environmental variables critical to cultivation. The system was structured with sensors for light intensity, temperature, and air and soil humidity, interconnected to a Raspberry Pi 4 microcontroller responsible for data centralization and actuator activation. The actuators consisted of fans, a grow light, a heating resistor, and a peristaltic pump for irrigation. The software architecture, implemented with Firebase, Google's mobile and web application development platform, ensured data storage and real-time communication, enabling integration between the physical prototype and the mobile application. This application, developed in React Native, a framework for creating mobile applications with JavaScript, using the Expo platform, enabled real-time visualization and management of the greenhouse, with configuration of desired variables, actuator calibration, and monitoring through trend charts. The tests included validation of the main functionalities of the system, including photoperiod control of the grow light (exposure time to artificial light in a 24 hours cycle), thermal regulation through the heater and fans, verification of air humidity control, and evaluation of automated irrigation with a peristaltic pump. In addition to these individual tests, a three-day experimental cycle was conducted, corresponding to different cultivation phases, in order to evaluate the integrated operation of the greenhouse under continuous operation. Throughout all tests, the system performed continuous data collection on light intensity, temperature, and air and soil humidity, recording them in real time and enabling monitoring through trend charts at 1 hour and 24 hours intervals. This approach made it possible to verify both the consistency of the measurements and the automatic response of the actuators to simulated environmental variations and the behavior of the cultivation cycle.

Keywords: Indoor farming; Internet of Things; Raspberry Pi. Mobile applications; Database.

LISTA DE IMAGENS

LISTA DE QUADROS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

EMBRAPA	Empresa Brasileira de Pesquisa Agropecuária
FAO	*Food and Agriculture Organization of the United Nations*
IBGE	Instituto Brasileiro de Geografia e Estatística
IoT	*Internet of Things*
JSON	*JavaScript Object Notation*
ODS	Objetivos de Desenvolvimento Sustentável
ONU	*United Nations*
PCB	*Printed Circuit Board*
PIB	Produto Interno Bruto
RFID	*Radio Frequency Identification*

SUMÁRIO

1 INTRODUÇÃO

O crescimento populacional mundial e a consequente elevação da demanda por alimentos têm impulsionado o desenvolvimento de novos métodos e tecnologias na produção agrícola. Nesse contexto, a Internet das Coisas (IoT) surge como uma oportunidade para o monitoramento e controle de variáveis ambientais, mitigando os efeitos das mudanças climáticas e contribuindo para avanços na eficiência produtiva (Stroparo, 2024).

Apesar do grande potencial que a IoT oferece à agricultura, sua implementação ainda enfrenta barreiras significativas, como a complexidade técnica e os custos elevados, que limitam a adoção dessas tecnologias por pequenos e médios produtores.

> Embora existam diversas tecnologias inovadoras disponíveis, e estufas automatizadas não sejam uma novidade, o acesso a essas soluções ainda é limitado, especialmente para pequenos e médios produtores, devido ao elevado custo. Esse fator impede a adoção dessas tecnologias por grande parte dos agricultores, particularmente em regiões onde predomina a Agricultura Familiar (Couto, 2024, p. 16).

Diante deste contexto, torna-se evidente a necessidade de soluções acessíveis, de menor custo e que colaborem com a difusão de tecnologias para o controle e monitoramento de variáveis ambientais. No estudo de Couto (2024), o alto custo ainda é um fator limitante no desenvolvimento e aplicação de estufas automatizadas. Nesse sentido, é notório que a propagação de tecnologias no setor agrícola depende de alternativas de baixo custo, que favoreçam o desenvolvimento tecnológico de pequenos produtores. Conforme apresentado por Mororó (2023), o desenvolvimento de um protótipo de estufa IoT demonstrou a viabilidade da integração de sensores e atuadores para o controle do ambiente de cultivo, operando de forma remota ou manual, e contribuiu para evidenciar o potencial da IoT na agricultura.

Além da relevância tecnológica, a adoção de estufas automatizadas também possui impacto social e ambiental significativo. Como destacado por Soares (2023, p.16), "Ela promove práticas agrícolas que reduzem o impacto no meio ambiente, garantem a viabilidade econômica a longo prazo para os agricultores e melhoram a qualidade de vida das comunidades rurais". Nesse sentido, a aplicação da IoT, alinhada a práticas sustentáveis, favorece pequenos produtores e promove a

diversificação da produção agrícola, contribuindo para o uso mais eficiente dos recursos naturais.

Considerando esse panorama, este trabalho propõe o desenvolvimento de um protótipo de estufa *indoor* baseado em IoT, com foco no monitoramento e controle de variáveis ambientais críticas ao cultivo, como luminosidade, temperatura do ar, temperatura do solo, umidade do ar e umidade do solo. O sistema integra sensores e atuadores ao microcontrolador Raspberry Pi 4, conectado a uma plataforma em nuvem para armazenamento de dados e comunicação em tempo real, além de um aplicativo móvel que possibilita o gerenciamento remoto da estufa.

1.1 OBJETIVOS

1.1.1 Objetivo geral

Desenvolver um protótipo de estufa *indoor* baseado em IoT.

1.1.2 Objetivo específicos

a) Projetar e integrar sensores, atuadores e microcontrolador para a estufa;
b) Desenvolver aplicativo móvel para monitoramento e controle remoto do sistema;
c) Validar o protótipo por meio de ensaios funcionais e simulações de cultivo.

1.2 JUSTIFICATIVA

O crescimento populacional mundial tem apresentado tendência de pressionar o avanço de sistemas de produção agrícola com aplicação da IoT. Prospecções da Organização das Nações Unidas (ONU) indicam que a população global deve atingir cerca de 9,7 bilhões de pessoas em 2050, ampliando significativamente a demanda por alimentos (ONU, 2019). Nesse cenário, a Organização das Nações Unidas para Agricultura e Alimentação (FAO) estima que o volume total de alimentos a ser produzido no mundo deverá crescer em 70%, a fim de atender essa necessidade (FAO, 2017).

No Brasil, a agricultura familiar desempenha papel fundamental na produção de alimentos básicos consumidos pela população. De acordo com o Censo Agropecuário 2017, aproximadamente 77% dos produtores rurais pertencem a esse segmento, responsável por grande parte da produção de feijão, mandioca, leite, aves e suínos (IBGE, 2019). Apesar dessa expressiva representatividade, a renda gerada pela agricultura familiar corresponde a apenas 23% do total da renda agrícola do país, evidenciando sua fragilidade econômica frente ao agronegócio (USP, 2024). Uma das prováveis causas para a menor da participação da agricultura familiar na renda total da produção agrícola do país, pode ser a dificuldade de adoção de tecnologias digitais por pequenos e médios produtores. Nesse contexto, aplicações baseadas em IoT, envolvendo sensores e atuadores para o monitoramento de variáveis ambientais, podem contribuir diretamente para a otimização do uso de água, energia e insumos agrícolas, reduzindo desperdícios e ampliando a eficiência produtiva.

> Outro importante fator que tem contribuído para a intensificação produtiva são os sistemas de cultivo em ambientes protegidos e irrigados. A produção agrícola em ambiente protegido tem experimentado grande crescimento no mundo, com destaque para a China, com mais de 3 milhões de hectares e 90% das estufas utilizadas no mundo (Jank, 2017; Monte et al., 2018, apud EMBRAPA, 2018, p.76).

No tocante à sustentabilidade da produção agrícola, a Empresa Brasileira de Pesquisa Agropecuária (EMBRAPA) explica que o desenvolvimento agrícola sustentável se baseia no manejo e na conservação dos recursos naturais, aliado a mudanças tecnológicas capazes de assegurar o atendimento contínuo das necessidades humanas (EMBRAPA, 2018). Apesar do protagonismo do Brasil como potência agrícola, dados do Instituto Brasileiro de Geografia e Estatística (IBGE) mostram que, no último trimestre de 2023, cerca de 27,6% dos domicílios brasileiros enfrentavam algum grau de insegurança alimentar, sendo 4,1% em situação grave (IBGE, 2024). Nesse cenário, a aplicação da IoT em sistemas de monitoramento e automação de estufas *indoor* apresenta-se como alternativa viável para ampliar os níveis de produção agrícola e reduzir desperdícios.

Diante desse cenário, o desenvolvimento do protótipo de uma estufa automatizada com IoT justifica-se como uma solução prática para demonstrar a aplicação dos conceitos da Agricultura 4.0, que integra tecnologias digitais à produção e à gestão agrícola, em ambientes de cultivo fechados.

1.3 ORGANIZAÇÃO DO TRABALHO

No Capítulo 1 apresenta-se a introdução ao tema, destacando os objetivos, a justificativa e a forma como o texto está organizado. O Capítulo 2 reúne o referencial teórico, abordando a Agricultura 4.0, o uso da IoT no meio agrícola, conceitos sobre automação e monitoramento em estufas, tecnologias de sensoriamento e controle, além de sistemas embarcados e trabalhos correlatos. O Capítulo 3 descreve a metodologia adotada, contemplando os procedimentos de pesquisa, critérios de escolha de componentes, etapas de prototipagem e estratégias de validação. O Capítulo 4 detalha o desenvolvimento do projeto, incluindo a montagem física da estufa, a implementação dos sensores e atuadores, o desenvolvimento da lógica de controle no microcontrolador Raspberry Pi, o aplicativo móvel e a integração do sistema. O Capítulo 5 apresenta a análise e discussão dos resultados obtidos nos testes de validação e operação integrada do protótipo. Por fim, o Capítulo 6 traz as conclusões, destacando as contribuições do trabalho e sugestões de trabalhos futuros.

2 REFERENCIAL TEÓRICO

2.1 AGRICULTURA E DESAFIOS ATUAIS

Ao longo da história, a agricultura ocupou um espaço relevante no desenvolvimento econômico e social, garantindo não apenas a produção de alimentos, mas também o fornecimento de matérias-primas para atender à população. No Brasil, essa relevância é ainda mais evidente. Segundo a EMBRAPA (2018), em 2016 o agronegócio representou 23,6% do Produto Interno Bruto (PIB), dos quais 5% correspondiam à produção agrícola, além de ser responsável por 45,9% das exportações nacionais. Esses números confirmam o papel estratégico da agricultura brasileira, tanto para a economia quanto para a sociedade.

Considerando o papel da agricultura brasileira, o setor enfrenta diversos desafios, como a necessidade de ampliar a produção sem comprometer a sustentabilidade das atividades agrícolas. A EMBRAPA (2018), destaca que o uso racional dos recursos naturais, a conservação do solo e a eficiência na utilização da água estão entre os principais pontos de atenção para o aumento da produção e o desenvolvimento do setor agrícola. Tais fatores mostram que o desenvolvimento da agricultura pode estar diretamente vinculado à aplicação de práticas sustentáveis.

Além dos desafios relacionados à sustentabilidade, a agricultura brasileira apresenta grande desigualdade produtiva e tecnológica entre a agricultura familiar e a agricultura patronal. Enquanto a primeira depende majoritariamente da mão de obra da própria família e de recursos limitados, a segunda se caracteriza por mão de obra assalariada, grandes propriedades e alto investimento em tecnologia e capital. A EMBRAPA (2018) aponta que grande parte dos pequenos agricultores não consegue adotar novas tecnologias devido a fatores como o alto custo de implementação e imperfeições de mercado. Entre essas imperfeições, destaca-se o baixo nível de escolaridade dos produtores, sendo que, "do total de produtores agropecuários, 15% declararam que nunca frequentaram escola; 14% frequentaram até o nível de alfabetização, e 43%, no máximo, o nível fundamental." (IBGE, 2019, p.68).

2.2 AGRICULTURA 4.0 E INTERNET DAS COISAS

Esta seção aborda os conceitos de Agricultura 4.0 e IoT, destacando seu papel no setor agrícola. Enquanto a Agricultura 4.0 representa o uso de diversas tecnologias digitais nos processos produtivos do campo, a IoT constitui-se como uma de suas principais ferramentas, permitindo a integração de sensores, atuadores e sistemas embarcados para coleta e análise de dados em tempo real.

2.2.1 Agricultura 4.0

A Agricultura 4.0, também denominada agricultura digital, é considerada a quarta revolução agrícola, sendo sucessora da Revolução Verde, que, segundo Mororó (2023), ocorreu no século XX e foi caracterizada pela modernização do setor, pela expansão no uso de fertilizantes e pesticidas e pelo desenvolvimento de máquinas agrícolas. Esta nova revolução está diretamente associada ao uso de tecnologias que integram o ambiente de cultivo a plataformas digitais, possibilitando a tomada de decisão em tempo real.

> A Agricultura 4.0 utiliza métodos computacionais de alto desempenho, rede de sensores, comunicação máquina para máquina, conectividade entre dispositivos móveis, computação em nuvem, métodos e soluções analíticas que processam grandes volumes de dados e constroem sistemas de suporte à tomada de decisões (Lisbinski et al., 2020, p. 423).

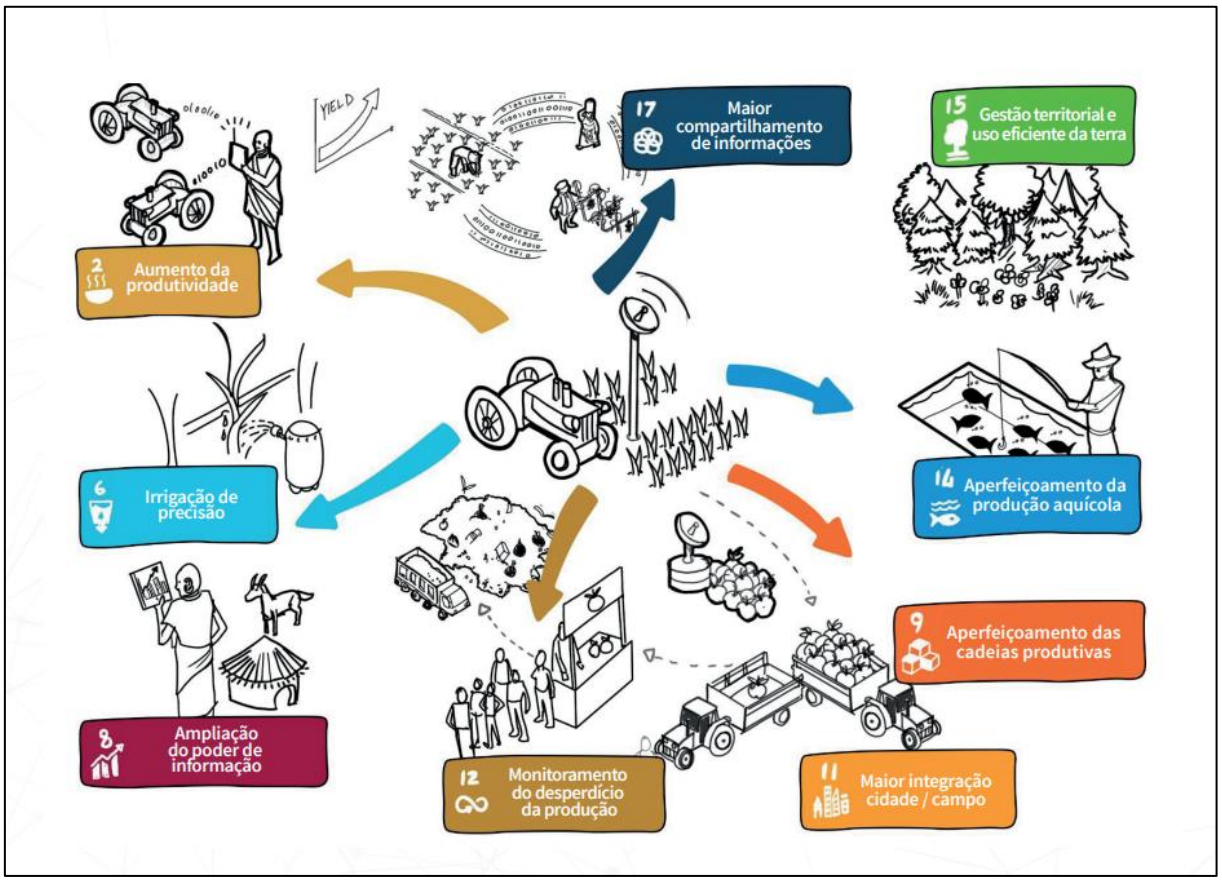
Juntamente com a forte integração de tecnologias digitais ao cultivo para a tomada de decisão, a Agricultura 4.0 possibilita ampliar a produtividade e a competitividade no campo. A EMBRAPA (2018) destaca que a chave para alcançar melhor aproveitamento e eficiência está em integrar o *hardware* a sistemas capazes de capturar dados precisos, utilizar algoritmos para torná-los aplicáveis e direcionar o uso dos equipamentos agrícolas de forma mais eficiente e produtiva. No contexto brasileiro, onde a demanda por alimentos tende a crescer nas próximas décadas, essa abordagem é essencial para assegurar o aumento da produção.

De forma complementar, Silva e Cavichioli (2020) destacam que a incorporação de recursos como automação, IoT, robótica, *big data* e inteligência artificial pode ampliar os índices de rendimento, reduzir custos e otimizar a gestão de recursos no campo. Segundo os autores, “a agricultura 4.0 é considerada muito importante, visto que traz diversos benefícios e oportunidades para o produtor rural, e todas as

informações e dados coletados em tempo real passam a estar à disposição do produtor" (Silva; Cavichioli, 2020, p. 618). No entanto, os autores demonstram que a adoção dessas tecnologias ainda encontra alguns obstáculos, como a limitação da conectividade em áreas rurais, o alto custo para implantação e o alto índice de mão de obra desqualificada para trabalhar com novas tecnologias (Silva; Cavichioli, 2020).

Diante do que foi apresentado, a Agricultura 4.0 evidencia uma série de benefícios e desafios. A Imagem 1 apresenta esses aspectos em alinhamento aos Objetivos de Desenvolvimento Sustentável (ODS) propostos pela ONU, destacando avanços como o aumento da produtividade, a adoção de sistemas de irrigação de precisão, o uso mais racional da terra e dos recursos naturais e a aproximação entre campo e cidade (EMBRAPA, 2018).

Imagem 1 — Benefícios da agricultura digital alinhado aos ODS



Fonte: EMBRAPA (2018).

Portanto, observa-se que a Agricultura 4.0 reúne tanto oportunidades quanto desafios, podendo oferecer maior produtividade e sustentabilidade ao setor agrícola.

Essas oportunidades reforçam a relevância de projetos que buscam soluções acessíveis e aplicáveis à realidade dos produtores.

2.2.2 Internet das coisas (IoT)

O termo IoT foi utilizado pela primeira vez em 1999 por Kevin Ashton, em uma apresentação na Procter & Gamble (P&G). Na ocasião, Ashton associou o conceito de identificação por radiofrequência (RFID) ao crescente potencial da internet, ressaltando a possibilidade de conectar objetos físicos em uma rede digital de comunicação. Anos depois, reforçou a ideia no RFID Journal, definindo a IoT como a possibilidade de conectar objetos físicos à internet para que possam coletar e compartilhar informações sem intervenção humana (ASHTON, 2009).

Esta tecnologia está cada vez mais presente em áreas como saúde, logística e indústria, consolidando-se como uma tecnologia essencial para o avanço da transformação digital. Na agricultura, seu uso permite acompanhar variáveis ambientais e integrar diferentes atuadores, facilitando a automação e o controle das operações no campo. Segundo Jesus (2021), o uso da IoT no ambiente agrícola vem se tornando indispensável, assumindo papel central no aumento da eficiência e da produtividade. De modo semelhante, Stroparo (2024), destaca que esta abordagem contribui diretamente para ganhos de produtividade e maior eficiência no uso de recursos agrícolas, tornando-se um dos principais pilares da Agricultura 4.0.

A IoT é um dos pilares centrais da Agricultura 4.0, pois permite integrar sensores, atuadores e sistemas embarcados a plataformas digitais. De acordo com Lisbinski et al. (2020, p. 428), "O fenômeno da agricultura 4.0 pode ser compreendido como uma aplicação massiva de tecnologias digitais na produção de alimentos e outros produtos agrícolas visando ganhos de eficiência". Entre essas tecnologias, a IoT atua em conjunto com outras inovações digitais que, integradas, contribuem para o aumento da produtividade. Além disso, Silva e Cavichioli (2020), ressaltam que, associada a recursos como *big data*, agricultura de precisão, automação e robótica, essa tecnologia favorece a redução de custos e a otimização do uso de insumos, consolidando-se como uma ferramenta importante para impulsionar o desenvolvimento do agronegócio.

Estudos recentes evidenciam o potencial da IoT na agricultura por meio do desenvolvimento de protótipos de cultivo. Soares (2023), por exemplo, propôs uma horta *indoor* equipada com sensores e atuadores integrados a plataformas digitais para monitoramento remoto. De forma semelhante, Mororó (2023), apresentou uma estufa inteligente baseada em IoT, que integra microcontroladores, banco de dados e *dashboards* (interface gráfica) de controle para automatizar variáveis ambientais e permitir a gestão remota do cultivo. Essas experiências demonstram que a aplicação dessa tecnologia já é uma realidade em projetos acadêmicos, reforçando seu potencial para contribuir com uma agricultura mais eficiente e sustentável.

Apesar do potencial apresentado, a adoção da IoT na agricultura ainda enfrenta barreiras significativas. Lisbinski et al. (2020), ressaltam que, entre os principais desafios, estão a necessidade de investimentos em infraestrutura e conectividade, aspectos que exigem maior atenção da comunidade acadêmica e de gestores públicos. Lisbinski et al. (2020), apontam que, embora a busca por eficiência agrícola seja um dos fatores que impulsionam a Agricultura 4.0, ela não garante o acesso às inovações por parte dos agricultores com menor poder aquisitivo, podendo ampliar desigualdades já existentes no meio rural. Tais constatações confirmam o que foi discutido na seção 2.1, indicando que a desigualdade produtiva e tecnológica ainda representa um obstáculo importante para o avanço da agricultura familiar.

2.3 ESTUFAS AGRÍCOLAS

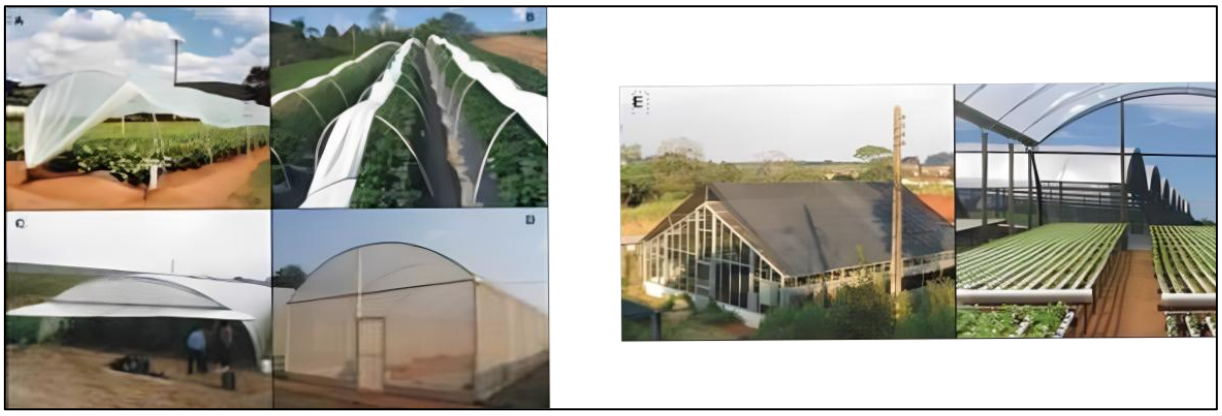
Esta seção apresenta o conceito de cultivo protegido, destacando sua evolução e principais características climáticas. Em seguida, são abordados os diferentes modelos de estufas, desde as estruturas tradicionais até aquelas automatizadas e integradas com a tecnologia da IoT.

2.3.1 Cultivo protegido

O cultivo protegido pode ser compreendido como um ambiente de produção agrícola planejado para reduzir a influência das condições climáticas externas sobre as plantas. Segundo Purquerio e Tivelli (2006), a década de 1980 foi um marco para a horticultura brasileira, em razão do avanço da indústria petroquímica, responsável

pela introdução de materiais como tubos gotejadores, vasos, silos, sistemas de impermeabilização e filmes plásticos utilizados na cobertura de túneis e estufas. Esses insumos possibilitaram a consolidação do cultivo protegido no país, viabilizando estruturas que funcionavam como abrigo, conforme ilustrado na Imagem 2.

Imagem 2 — Modelos de estruturas de cultivo protegido



Fonte: Purquerio e Tivelli (2006).

Nesse contexto, Guedes et al. (2024), destacam que a produção em ambientes protegidos representa um avanço importante nas práticas agrícolas, pois reduz os impactos das condições climáticas adversas e garante maior estabilidade produtiva ao longo do ano. De acordo com essa perspectiva, o controle das variáveis ambientais em estruturas de cultivo protegido possibilita a produção mesmo em períodos críticos para o desenvolvimento das plantas. Conforme Sentelhas e Santos (1995, p.108), “O emprego de estufas plásticas é responsável por alterações em diversos elementos meteorológicos, tornando viável a produção de vegetais em épocas ou lugares cujas condições climáticas são críticas”.

Do ponto de vista técnico, as variáveis ambientais em estruturas de cultivo protegido apresentam comportamento característico. Entre os principais efeitos observados estão a redução da radiação solar incidente, o aumento da temperatura do ar durante o período diurno e sua acentuada queda no período noturno, além da elevação da temperatura do solo. Observa-se também que a umidade relativa do ar tende a variar de forma inversamente proporcional à temperatura do ambiente (Sentelhas; Santos, 1995).

Embora o cultivo protegido proporcione condições favoráveis à produção durante todo o ano, Guedes et al. (2024), ressaltam que grande parte dos problemas

observados decorre da concepção equivocada de que a cobertura plástica serve apenas para proteger contra a chuva, desconsiderando as alterações que ocorrem no ambiente interno. Essas alterações exigem adaptações importantes no manejo e no controle das variáveis ambientais.

Diante do que foi apresentado, para o melhor aproveitamento do cultivo protegido torna-se necessário o monitoramento das variáveis ambientais. Nesse contexto, a automação e o uso de tecnologias digitais, como a IoT, mostram-se como alternativas capazes de suprir essa necessidade.

2.3.2 Modelos de estufa

Os modelos de estufa evoluíram de estruturas simples, destinadas apenas à proteção física das plantas, para sistemas que integram recursos tecnológicos e digitais. Dessa forma, essa evolução demonstra como o cultivo protegido acompanha as inovações visando eficiência e sustentabilidade no setor agrícola.

2.3.2.1 Estufas tradicionais

As estufas tradicionais, cujo modelo está ilustrado na Imagem 3, são compostas por estruturas básicas de madeira, metal ou plástico, tendo como objetivo principal proteger os cultivos contra variações extremas das condições climáticas, como chuvas intensas, ventos e variações de temperatura (Purquerio; Tivelli, 2006; Sentelhas; Santos, 1995).

Imagem 3 — Modelo de estufa tradicional

Fonte: Agrícolas (2025).

2.3.2.2 Estufas automatizadas

Os modelos de estufas automatizadas incorporam sistemas de ventilação, irrigação e, em alguns casos, controle de temperatura e umidade. A Imagem 4 apresenta um modelo de estufa automatizada. Essas tecnologias ampliam a eficiência do cultivo, favorecendo ganhos em produtividade (EMBRAPA, 2018).

Imagem 4 — Modelo de estufa automatizada

Fonte: Agrícolas (2025).

2.3.2.3 Estufas IoT

O estágio mais avançado dentro das estufas é representado pelos modelos de estufas IoT, cujo exemplo está ilustrado na Imagem 5. Nelas, sensores e atuadores monitoram e controlam variáveis como temperatura, umidade, luminosidade e irrigação, integrados a plataformas digitais que permitem o acompanhamento remoto em tempo real (Soares, 2023; Mororó, 2023; Couto, 2024; Jesus, 2021).

Imagem 5 — Modelo de estufa IoT

Fonte: Polygreenhouse (2025).

2.4 TECNOLOGIAS DE MONITORAMENTO E CONTROLE

O funcionamento das estufas inteligentes depende diretamente da integração das tecnologias da Agricultura 4.0. Para que operem com eficiência, é importante que possuam a capacidade de monitorar e controlar as variáveis ambientais, garantindo condições adequadas ao cultivo. Para tornar isso possível, é necessário o uso combinado de sensores, atuadores, sistemas embarcados e plataformas digitais, que, em conjunto, possibilitam a tomada de decisão assegurando o funcionamento

adequado e os benefícios já discutidos na Subseção 2.2. As subseções a seguir apresentam cada um desses componentes e seu papel no contexto da agricultura digital.

2.4.1 Sensores

Os sensores ocupam uma posição de destaque em sistemas IoT, pois são responsáveis pela obtenção dos dados das variáveis ambientais. Jesus (2021), ressalta que esses dispositivos permitem o monitoramento preciso dessas variáveis e fornecem informações essenciais para apoiar a tomada de decisão no manejo agrícola.

O uso de sensores na agricultura está associado aos benefícios propostos pela Agricultura 4.0, que incluem ganhos de rendimento, melhoria da qualidade e maior sustentabilidade da produção.

> A tecnologia 4.0 na agricultura pode promover a melhoria do rendimento, o aumento da qualidade dos produtos e do processamento, a sustentabilidade das culturas, a melhoria das condições de trabalho entre outras vantagens como aumento da produtividade, redução de custos e desperdícios (Lisbinski et al., 2020, p. 428).

Além desses benefícios gerais, pesquisas recentes demonstram a aplicação prática dos sensores em protótipos de estufas inteligentes com IoT. Nesse sentido, Couto (2024), destaca o desenvolvimento de um protótipo voltado à coleta de informações ambientais, integrando diferentes sensores para viabilizar um sistema de monitoramento mais preciso e eficiente.

De maneira geral, os sensores fornecem a base de dados para a automação agrícola e permitem a integração com os atuadores, que transformam essas informações em ações no ambiente de cultivo.

2.4.2 Atuadores

No que se refere aos atuadores, Mororó (2023), aponta que esses dispositivos exercem a função inversa aos sensores, convertendo sinais de controle em ações diretas no ambiente. Um exemplo é o acionamento de equipamentos responsáveis pela aplicação de insumos, em que o sistema, a partir de uma decisão automatizada,

atua diretamente sobre o ambiente de cultivo. De maneira complementar, Soares (2023, p.30), define que, “Os atuadores são componentes essenciais em sistemas automatizados, pois são responsáveis por realizar ações físicas com base em comandos recebidos do sistema de controle”.

O uso de atuadores na agricultura tem permitido automatizar tarefas que antes exigiam esforço manual e apresentavam menor precisão, como o controle da irrigação. Com base nos dados enviados pelos sensores, esses dispositivos ajustam o funcionamento dos equipamentos conforme a necessidade do cultivo, reduzindo o desperdício e tornando o manejo mais eficiente. Neste sentido, os atuadores contribuem para um uso mais racional da água e da energia, refletindo diretamente na sustentabilidade do sistema agrícola.

> Desta forma, o uso adequado de recursos hídricos no meio rural envolve decisões sobre a irrigação, uso de métodos recomendados para cada tipo de solo e cultura, além do seu manejo a partir do monitoramento preciso da evapotranspiração, utilização de sistemas mais eficientes e adaptados às condições locais, evitando o desperdício de água e energia (EMBRAPA, 2018, p.72).

Na prática, diferentes protótipos de estufas inteligentes têm integrado atuadores a sistemas embarcados para controlar o ambiente de cultivo. Couto (2024), descreve a aplicação desse mecanismo no acionamento automático da ventilação, a partir das leituras de temperatura. Já Mororó (2023), apresenta o uso de atuadores no bombeamento de água, adotando como critério o nível mínimo do reservatório para evitar o acionamento a seco.

2.4.3 Sistemas embarcados

Os sistemas embarcados constituem a base de processamento em aplicações de IoT, integrando sensores, atuadores e protocolos de comunicação. De acordo com Soares (2023, p.24), "Os microcontroladores utilizados em IoT atuam na coleta, processamento e transmissão de dados entre dispositivos conectados à internet". A Imagem 6 ilustra alguns exemplos de sistemas embarcados utilizados em projetos de automação agrícola, como os microcontroladores Arduino e Raspberry Pi, que desempenham funções de controle, comunicação e processamento dentro das estufas inteligentes.

Imagem 6 — Modelos de sistemas embarcados

Fonte: Santos (2021).

No contexto das estufas inteligentes, os sistemas embarcados atuam como unidade central de controle, coordenando a integração entre dispositivos e plataformas. Estudos recentes demonstram sua aplicação em protótipos voltados ao manejo agrícola, contemplando processos como irrigação, ventilação e monitoramento de temperatura (Mororó, 2023; Couto, 2024).

Assim, os sistemas embarcados assumem papel estratégico ao centralizar o controle do ambiente, mas sua eficiência depende diretamente da comunicação, do armazenamento dos dados e das lógicas de controle.

2.4.4 Comunicação e armazenamento

A comunicação em sistemas IoT é o que permite que todos os dispositivos troquem informações entre si de forma organizada. É por meio dela que sensores, atuadores e controladores conseguem trabalhar juntos. Segundo Jesus (2021), os protocolos de comunicação funcionam como uma linguagem comum entre os dispositivos, garantindo que a troca de informação ocorra de maneira eficiente.

No armazenamento, é essencial que tanto os dados coletados pelos sensores quanto as lógicas dos sistemas de uma estufa inteligente sejam organizados de forma estruturada e acessível. Isso garante que as informações estejam disponíveis para apoiar a tomada de decisão. Segundo Jesus (2021), o uso de bancos de dados em sistemas IoT viabiliza a criação de conjuntos de dados que servem de base para diferentes aplicações, como o monitoramento e avaliação do desempenho agrícola.

Além disso, o aumento no volume de informações geradas por sistemas IoT voltados à agricultura reforça a necessidade de soluções de armazenamento confiáveis. Segundo Silva e Cavichioli (2020), a adoção de tecnologias de comunicação e armazenamento possibilita a análise dos dados, permitindo que os produtores tenham acesso a informações precisas sobre o cultivo, do início ao fim do processo.

Assim, a comunicação e o armazenamento constituem a base para a integração dos dados em estufas IoT, possibilitando a atuação eficiente dos sistemas de controle.

2.4.5 Lógicas de controle

O controle em sistemas IoT é a etapa que transforma as informações coletadas em ações inteligentes dentro do ambiente de cultivo. Ele atua como elo entre os sensores, que captam as variáveis ambientais, as estruturas de comunicação e armazenamento, que asseguram a troca de dados, e os atuadores, responsáveis por executar as ações físicas. Em estufas inteligentes, Stroparo (2024), destaca que o monitoramento contínuo dessas variáveis possibilita decisões mais precisas, e que o uso do controle automatizado resulta em melhorias significativas na produtividade e na qualidade dos cultivos.

Entre as diversas estratégias de controle, uma das formas mais utilizadas em protótipos de estufas é o acionamento direto, que alterna o estado dos atuadores entre ligado e desligado. Nessa lógica, os dispositivos podem ser programados para operar em horários pré-definidos ou acionados conforme as leituras dos sensores. Apesar da simplicidade, essa abordagem permite automatizar processos como irrigação,

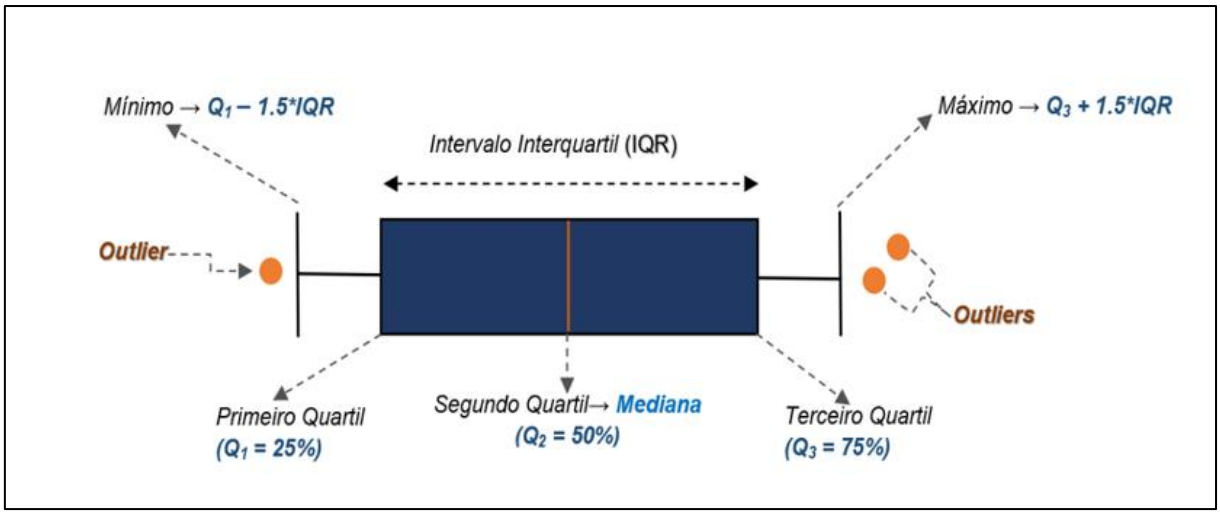
ventilação e iluminação, apresentando resultados satisfatórios com baixo custo de implementação (Mororó, 2023).

Assim, mesmo em aplicações simples, o controle integra todas as etapas de um sistema IoT, conectando sensores, atuadores, plataformas de armazenamento e comunicação. Essa integração assegura que o ambiente de cultivo seja mantido dentro de condições adequadas de forma autônoma, confiável e eficiente.

2.5 MÉTODO DOS QUARTIL

O método dos quartis é uma abordagem estatística utilizada para identificar valores que se distanciam do comportamento esperado dos dados, definidos como *outliers*. Este método é eficaz por utilizar medidas estatísticas robustas e pouco sensíveis a distribuições irregulares (Seo, 2006). O processo inicia com a ordenação dos dados em sequência crescente e sua divisão em quatro partes iguais. O primeiro quartil (Q1) representa os 25% inferiores da amostra, enquanto o terceiro quartil (Q3) corresponde aos 25% superiores. A diferença entre esses dois limites define o intervalo interquartil (IQR), que serve como referência para detectar *outliers*. Com essas medidas, considera-se outlier leve qualquer valor que esteja abaixo de $Q1 - 1{,}5 * IQR$ ou acima de $Q3 + 1{,}5 * IQR$ (Oliveira et al., 2014). A Imagem 7 apresenta uma ilustração desse método.

Imagem 7 — Método do intervalo interquartil



Fonte: Santos (2021).

2.6 TRABALHOS CORRELATOS

A análise de trabalhos correlatos ajuda a compreender como a tecnologia vem sendo aplicada no desenvolvimento de cultivos e estufas inteligentes. Esses estudos servem como base para esta pesquisa, tanto na construção do protótipo quanto na comparação das soluções adotadas.

Mororó (2023), desenvolveu um protótipo de estufa inteligente com base no microcontrolador ESP32, integrando sensores de temperatura, umidade e luminosidade, além de atuadores responsáveis pelo controle do ambiente de cultivo. O sistema utilizou o banco de dados MySQL para gerenciar as informações e a plataforma TagoIO para exibir e controlar os dados por meio de *dashboards*. O projeto se destacou por permitir o controle remoto da estufa em diferentes modos de operação, oferecendo flexibilidade ao agricultor para ajustar o cultivo conforme suas necessidades

Soares (2023), desenvolveu um protótipo de horta *indoor* utilizando os microcontroladores Arduino Uno e ESP8266, integrados a sensores de umidade e temperatura, além de atuadores para o sistema de irrigação e iluminação. O sistema utilizou a plataforma Adafruit IO para monitoramento remoto, permitindo a visualização das variáveis ambientais em *dashboards*. O diferencial apontado pela autora foi a flexibilidade do sistema, que possibilita adaptação a diferentes tipos de cultivo e a expansão com novos sensores e atuadores.

Couto (2024), desenvolveu um sistema de gerenciamento e controle de estufas de pequeno porte, utilizando sensores de temperatura, umidade do ar e do solo, integrados a atuadores como bomba de água e ventilador. O sistema foi controlado por um Raspberry Pi 3 em conjunto com um Arduino Uno, empregando o Node-RED tanto para o gerenciamento das leituras em tempo real quanto para a interface gráfica. O principal resultado do estudo foi demonstrar a viabilidade de soluções acessíveis voltadas a pequenos produtores, validando o protótipo em ambiente real e destacando a praticidade da integração via Node-RED.

Jesus (2021), desenvolveu uma aplicação de IoT voltada à agricultura de precisão, estruturando uma rede de sensores para monitorar variáveis climáticas e do solo, como temperatura, umidade do ar e umidade do solo. O sistema utilizou

microcontroladores conectados via Wi-Fi, armazenando os dados em um banco de dados e disponibilizando-os em tempo real por meio de uma plataforma *web*. O estudo comprovou que a infraestrutura proposta é viável e eficiente para coletar e transmitir dados, reforçando o potencial da IoT como ferramenta de apoio ao monitoramento agrícola.

De modo geral, os trabalhos analisados apresentam o avanço das soluções envolvendo a Agricultura 4.0. Embora utilizem diferentes plataformas e arquiteturas, todos compartilham o mesmo objetivo: aplicar os conceitos da Agricultura 4.0 ao ambiente de cultivo. O presente trabalho diferencia-se ao propor uma abordagem integrada, que reúne o Raspberry Pi 4, a plataforma Firebase e um aplicativo móvel próprio, possibilitando o acompanhamento e o controle das variáveis ambientais e do ciclo de cultivo em tempo real. Essa integração busca não apenas automatizar o ambiente, mas também oferecer ao usuário uma ferramenta prática para a gestão do cultivo em todas as suas etapas.

3 MATERIAIS E MÉTODOS

Este capítulo descreve os materiais e métodos empregados no desenvolvimento do protótipo. Apresentam-se os componentes de *hardware* e as plataformas de *software* selecionadas, seguidos pela metodologia de prototipagem, montagem física e implementação da lógica de controle. Por fim, detalham-se os procedimentos de teste e validação do sistema integrado.

3.1 MATERIAIS

3.1.1 Hardware

Os componentes de *hardware* foram organizados em quatro categorias funcionais: sensores para monitoramento das variáveis ambientais, atuadores para controle ativo do ambiente, microcontrolador para processamento de dados e tomada de decisão, e dispositivos auxiliares para alimentação elétrica, conversão de sinais e comunicação entre componentes.

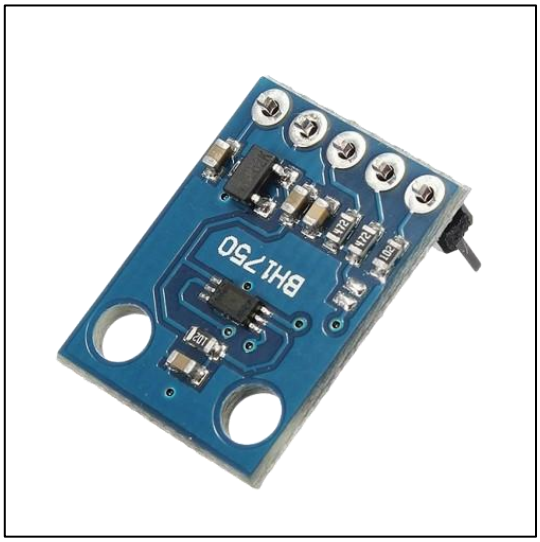
3.1.1.1 Sensores

Os sensores são responsáveis pela coleta periódica das variáveis ambientais da estufa, fornecendo os dados necessários para o monitoramento e controle das condições internas. A seleção dos sensores baseou-se em critérios como precisão, custo-benefício, disponibilidade no mercado e compatibilidade com o microcontrolador ESP32. Foram integrados ao protótipo quatro sensores: BH1750 para luminosidade, DHT22 para temperatura e umidade do ar, DS18B20 para temperatura do solo, e sensor capacitivo para umidade do solo. A seguir, apresentam-se as características técnicas de cada sensor e sua função no sistema.

3.1.1.1.1 Sensor BH1750

O sensor BH1750 mede a iluminância em lux por meio de um fotodiodo que converte a luz incidente em corrente elétrica proporcional. Essa corrente é amplificada e convertida em sinal digital, permitindo leituras diretas via protocolo I²C. A Imagem 8 ilustra o sensor.

Imagem 8 — Sensor de luminosidade BH1750



Fonte: MakerHero (2025).

No protótipo, o BH1750 monitora continuamente a iluminância interna da estufa, exibindo os valores em tempo real na interface do aplicativo móvel. Os dados coletados permitem o ajuste da luminária conforme as demandas de cada fase de cultivo. O Quadro 1 apresenta as principais especificações técnicas do sensor.

Quadro 1 — Especificações técnicas do sensor de luminosidade BH1750

Parâmetro	**Especificação**
Tensão de operação	2,4 V a 3,6 V
Corrente de operação	120 μA
Faixa de medição	1 lx a 65.535 lx
Resolução	1 lx
Precisão	± 20%
Protocolo de comunicação	I²C

Fonte: ALLDATASHEET(2025).

3.1.1.1.2 Sensor DS18B20

O DS18B20 é um sensor de temperatura digital baseado em bandgap, no qual a tensão variável aplicada a um diodo pode ser usada para fornecer leituras de temperatura (Tansley, Fletcher e Longstaff, 2013). A Imagem 9 ilustra o sensor.

Imagem 9 — Sensor de temperatura DS18B20

Fonte: MakerHero (2025).

Empregado para monitoramento da temperatura do solo, o DS18B20 exibe os valores em tempo real na interface do aplicativo móvel. O Quadro 2 apresenta as principais especificações técnicas do sensor.

Quadro 2 — Especificações técnicas do sensor de temperatura DS18B20

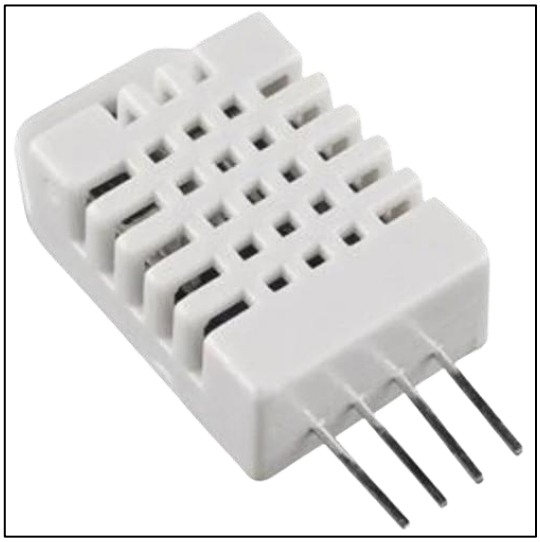
Parâmetro	**Especificação**
Tensão de operação	3,0 V a 5,5 V
Corrente de operação	1,0 mA
Faixa de medição	-55 °C a +125 °C
Resolução	0,5 °C a 0,0625 °C
Precisão	± 0,5 °C
Protocolo de comunicação	1-Wire

Fonte: ALLDATASHEET(2025).

3.1.1.1.3 Sensor DHT22

O DHT22 é um sensor digital que mede simultaneamente temperatura e umidade relativa do ar em um único módulo. Para isso, emprega um elemento capacitivo para detecção de umidade e um termistor para medição de temperatura. A Imagem 10 ilustra o sensor.

Imagem 10 — Sensor de temperatura e umidade do ar DHT22



Fonte: MakerHero (2025).

Integrado para monitoramento da temperatura e umidade do ar, o DHT22 exibe os valores em tempo real na interface do aplicativo móvel. As medições obtidas possibilitam o acionamento dos sistemas de ventilação e aquecimento. O Quadro 3 apresenta as principais especificações técnicas do sensor.

Quadro 3 — Especificações técnicas do sensor de temperatura e umidade do ar DHT22

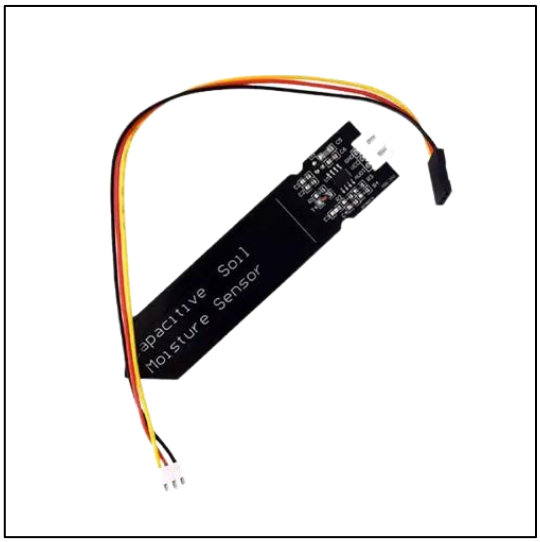
Parâmetro	**Especificação**
Tensão de operação	3,3 V a 5,5 V
Corrente de operação	500 µA
Faixa de medição (umidade)	0% a 99,9%
Faixa de medição (temperatura)	-40 °C a +80 °C
Resolução (umidade)	0,1%
Resolução (temperatura)	0,1 °C
Precisão (umidade)	± 2%
Precisão (temperatura)	± 0,5 °C
Protocolo de comunicação	1-Wire

Fonte: ALLDATASHEET(2025).

3.1.1.1.4 Sensor capacitivo de umidade do solo

O sensor capacitivo de umidade do solo mede o nível de umidade por meio da variação de capacitância entre suas placas internas. Diferentemente dos sensores resistivos, não possui partes metálicas expostas, reduzindo a corrosão e aumentando sua durabilidade. Sua saída é um sinal analógico proporcional à umidade, convertido em valor digital por um conversor analógico-digital externo ao sensor. A Imagem 11 ilustra o sensor.

Imagem 11 — Sensor capacitivo de umidade do solo



Fonte: MakerHero (2025).

Responsável pelo monitoramento da umidade do solo, o sensor capacitivo aciona o sistema de irrigação quando os valores medidos ficam abaixo do limite estabelecido. Os dados coletados são exibidos em tempo real na interface do aplicativo móvel. O Quadro 4 apresenta as principais especificações técnicas do sensor.

Quadro 4 — Especificações técnicas do sensor capacitivo de umidade do solo

Parâmetro	**Especificação**
Tensão de operação	3,3 V a 5,5 V
Corrente de operação	5 mA
Faixa de medição	0% a 100%

Fonte: MakerHero (2025).

Embora o fabricante não informe os parâmetros de precisão e resolução, estudos experimentais relatam que sensores capacitivos apresentam desempenho adequado para o monitoramento da umidade do solo em aplicações agrícolas (Costa, 2020).

3.1.1.2 Atuadores

Os atuadores executam as ações de controle dentro da estufa, permitindo o ajuste das variáveis ambientais para manter as condições de cultivo dentro dos parâmetros estabelecidos. Foram integrados ao protótipo quatro atuadores: resistor de aquecimento, bomba peristáltica, luminária LED e ventoinhas. A seguir, apresentam-se as características técnicas de cada atuador e sua função no sistema.

3.1.1.2.1 Resistor de aquecimento

O resistor de aquecimento eleva a temperatura interna da estufa quando os valores medidos pelo sensor DHT22 ficam abaixo do limite estabelecido, mantendo o ambiente de cultivo dentro dos parâmetros necessários. A Imagem 12 ilustra o componente, e o Quadro 5 apresenta suas principais especificações técnicas.

Imagem 12 — Resistor de aquecimento

Fonte: elaborado pelo próprio autor.

Quadro 5 — Especificações técnicas do resistor de aquecimento

Parâmetro	**Especificação**
Tensão	220 V

Corrente	227,27 mA
Potência	50 W
Resistência	968 Ω

Fonte: elaborado pelo próprio autor.

3.1.1.2.2 Bomba peristáltica

A bomba peristáltica realiza a irrigação da estufa, acionando o fluxo de água quando o sensor capacitivo detecta níveis de umidade abaixo do limite estabelecido. O atuador garante o fornecimento controlado de água para o solo. A Imagem 13 ilustra o componente, e o Quadro 6 apresenta suas principais especificações técnicas.

Imagem 13 — Bomba peristáltica

Fonte: MakerHero (2025).

Quadro 6 — Especificações técnicas da bomba peristáltica

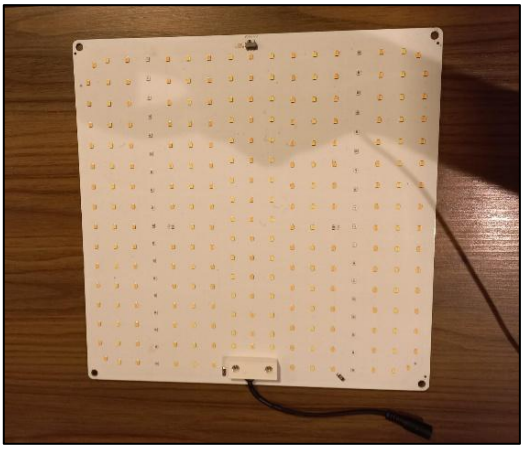
Parâmetro	**Especificação**
Tensão	12 V
Corrente	350 mA
Vazão	100 ml/min

Fonte: MakerHero (2025).

3.1.1.2.3 Luminária led

A luminária LED fornece iluminação artificial controlada, mantendo os níveis de luminosidade conforme os parâmetros estabelecidos. O acionamento é gerenciado pelo fotoperíodo configurado no sistema, que determina os períodos de operação ligada ou desligada. Adicionalmente, o dispositivo possui controle de dimerização, permitindo ajustar a iluminância de acordo com as demandas de cada fase de cultivo. A Imagem 14 ilustra o componente, e o Quadro 7 apresenta suas principais especificações técnicas.

Imagem 14 — Luminária LED



Fonte: elaborado pelo próprio autor.

A escolha por uma luminária com controle de dimerização se justifica pelo fato de que diferentes fases do cultivo (germinação, crescimento e floração) apresentam exigências distintas de iluminância. A dimerização permite ajustar este parâmetro otimizando o desenvolvimento das plantas.

Quadro 7 — Especificações técnicas da luminária LED

Parâmetro	**Especificação**
Tensão de operação	85 V a 265 V
Potência nominal	65 W
Eficiência luminosa	160 lm/W
Fator de potência	0,9
Frequência	50/60 Hz

Fonte: elaborado pelo próprio autor.

3.1.1.2.4 Ventoinha

As ventoinhas promovem a circulação de ar no interior da estufa, auxiliando no controle de temperatura e umidade. O acionamento ocorre conforme as variações detectadas pelo sensor DHT22. Simultaneamente, operam em conjunto com o resistor de aquecimento com intuito de contribuir com a distribuição térmica no ambiente. A Imagem 15 ilustra o componente, e o Quadro 8 apresenta suas principais especificações técnicas.

Imagem 15 — Ventoinha

Fonte: Eletrogate (2025).

Quadro 8 — Especificações técnicas da ventoinha

Parâmetro	**Especificação**
Tensão de operação	12 V
Corrente de operação	0,30 A
Potência nominal	3,60 W

Fonte: elaborado pelo próprio autor.

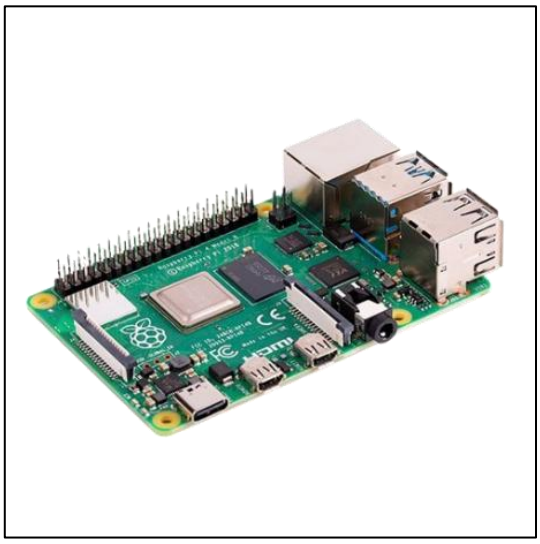
3.1.1.3 Microcontrolador

O Raspberry Pi 4 Model B foi escolhido pela alta capacidade de processamento e pelo suporte nativo a múltiplas *threads*, permitindo ler sensores, acionar atuadores e comunicar com o Firebase simultaneamente sem bloqueios. Diferentemente de microcontroladores como ESP32 ou Arduino, que operam com loop único e multitarefa

limitada, o Raspberry Pi executa um sistema operacional Linux completo, oferecendo paralelismo real.

O dispositivo integra os componentes de *hardware* e *software* do sistema, executando a leitura dos sensores, o processamento dos dados coletados e o acionamento dos atuadores conforme os parâmetros estabelecidos. O microcontrolador gerencia também a comunicação com o Firebase, possibilitando o envio e recebimento de informações em tempo real entre a estufa e o aplicativo móvel. A Imagem 16 ilustra o componente, e o Quadro 9 apresenta suas principais especificações técnicas.

Imagem 16 — Microcontrolador Raspberry Pi 4 Model B



Fonte: MakerHero (2025).

Quadro 9 — Especificações técnicas do microcontrolador Raspberry Pi 4 Model B

Parâmetro	**Especificação**
Processador	Broadcom BCM2711 quad-core Cortex-A72
Memória RAM	4 GB
Armazenamento	Cartão microSD
Conectividade	Wi-Fi 2,4 GHz / 5 GHz, Bluetooth 5.0, Ethernet Gigabit
Interfaces GPIO	40 pinos de entrada/saída digital

Sistema operacional	Raspberry Pi OS (baseado em Linux)
Alimentação	5 VDC / 3 A via conector USB-C

Fonte: MakerHero (2025).

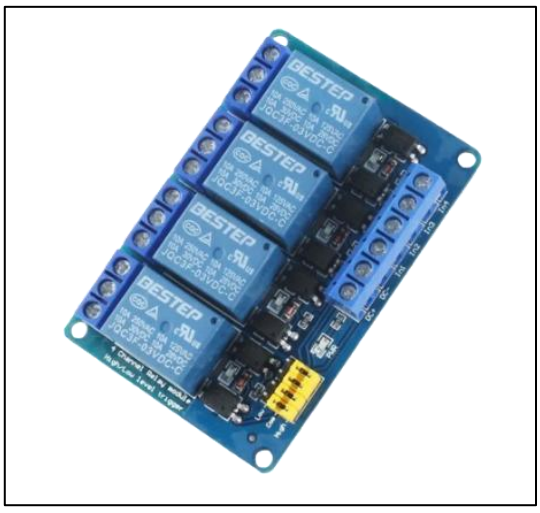
3.1.1.4 Dispositivos auxiliares

Os dispositivos auxiliares garantem a alimentação elétrica, a conversão de sinais e a comunicação entre os componentes do sistema. Neste protótipo foram utilizados quatro componentes: módulos relé, conversor analógico-digital, ponte H e fonte de alimentação. A seguir, apresentam-se as características técnicas de cada dispositivo e sua aplicação no sistema.

3.1.1.4.1 Módulo relé

O módulo relé de quatro canais intermediou o acionamento dos atuadores que operam em diferentes tensões: luminária LED e resistor de aquecimento em 220 V, bomba peristáltica e ventoinhas em 12 V. A Imagem 17 ilustra o componente.

Imagem 17 — Módulo relé com 4 canais



Fonte: Eletrogate (2025).

O acionamento é controlado pelo Raspberry Pi, que aplica tensão na bobina do relé, fechando a chave de contato e energizando o circuito da carga. O Quadro 10 apresenta as principais especificações técnicas do módulo.

Quadro 10 — Especificações técnicas do módulo relé com 4 canais

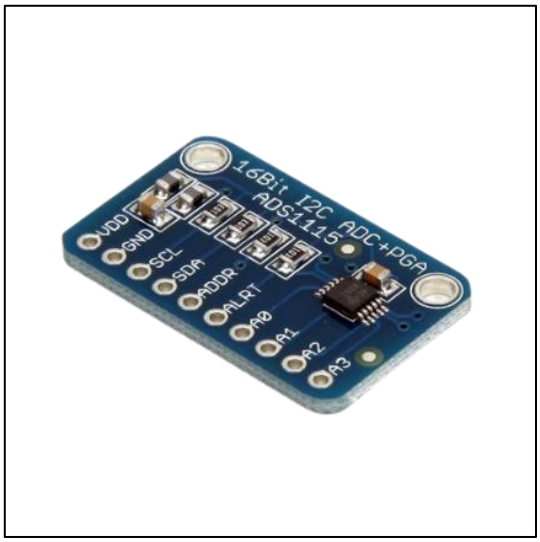
Parâmetro	**Especificação**
Tensão de acionamento	3 V
Tensão nominal	250 V
Corrente nominal	10 A
Canais	4
Saídas	NO, NC e COM

Fonte: elaborado pelo próprio autor.

3.1.1.4.2 Conversor analógico-digital ADS1115

O conversor analógico-digital ADS1115 converte o sinal analógico gerado pelo sensor capacitivo de umidade do solo em valores digitais que podem ser processados pelo Raspberry Pi. Como o microcontrolador não possui entradas analógicas nativas, o conversor é essencial para viabilizar a leitura dos dados do sensor. A Imagem 18 ilustra o componente, e o Quadro 11 apresenta suas principais especificações técnicas.

Imagem 18 — Conversor analógico digital ADS1115



Fonte: MakerHero (2025).

Quadro 11 — Especificações técnicas do conversor analógico digital ADS1115

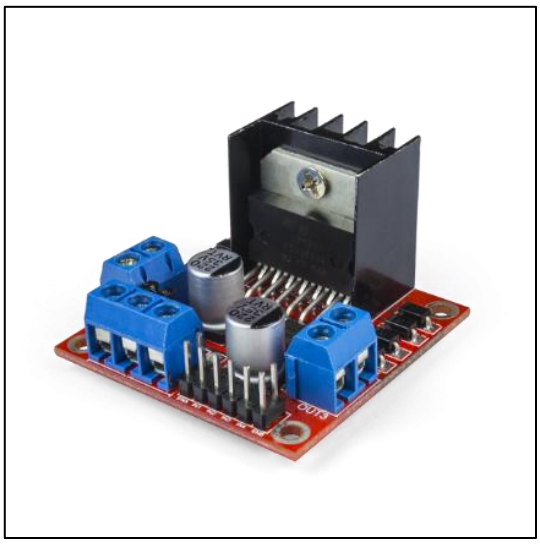
Parâmetro	**Especificação**
Tensão de operação	2,0 V a 5,5 V
Corrente de operação	150 μA
Resolução	16 bits
Protocolo de comunicação	I²C

Fonte: MakerHero (2025).

3.1.1.4.3 Ponte H L298N

A ponte H (módulo L298N) aciona a bomba peristáltica de 12 V, fornecendo a corrente e tensão necessárias para seu funcionamento. O módulo atua como interface entre o Raspberry Pi e o motor da bomba. A Imagem 19 ilustra o componente, e o Quadro 12 apresenta suas principais especificações técnicas.

Imagem 19 — Ponte H L298N



Fonte: MakerHero (2025).

Quadro 12 — Especificações técnicas da ponte H L298N

Parâmetro	**Especificação**
Tensão de entrada	4,8 V a 46 V
Corrente de saída	2 A a 4 A

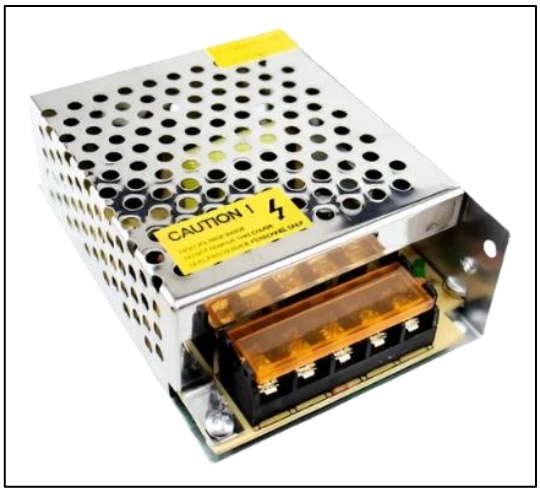
Tensão lógica de controle	5 V
Corrente lógica de controle	0 a 36 mA

Fonte: MakerHero (2025).

3.1.1.4.4 Fonte de alimentação chaveada

A fonte de alimentação chaveada fornece energia aos componentes do sistema que operam em 12 V, alimentando diretamente as ventoinhas e a ponte H que controla a bomba peristáltica. A Imagem 20 ilustra o componente, e o Quadro 13 apresenta suas principais especificações técnicas.

Imagem 20 — Fonte de alimentação chaveada



Fonte: elaborado pelo próprio autor

Quadro 13 — Especificações técnicas da fonte de alimentação chaveada

Parâmetro	**Especificação**
Tensão de entrada	100 V a 240 V
Corrente de entrada	0,60 A a 0,25 A
Tensão de saída	12 V
Corrente nominal de saída	5 A
Potência nominal	60 W

Fonte: elaborado pelo próprio autor.

3.1.2 Software

As plataformas de *software* empregadas no desenvolvimento do protótipo são apresentadas em três categorias: Python, para a lógica embarcada no Raspberry Pi, Firebase, para gerenciamento de dados e comunicação em tempo real, e React Native, para o desenvolvimento do aplicativo móvel. A seguir, detalham-se as características e aplicações de cada plataforma no sistema.

3.1.2.1 Python

Python foi a linguagem escolhida para o desenvolvimento da lógica embarcada, executada diretamente no Raspberry Pi. A escolha ocorreu devido à ampla disponibilidade de bibliotecas e à compatibilidade com os componentes utilizados, facilitando a integração entre sensores, atuadores e plataformas de comunicação.

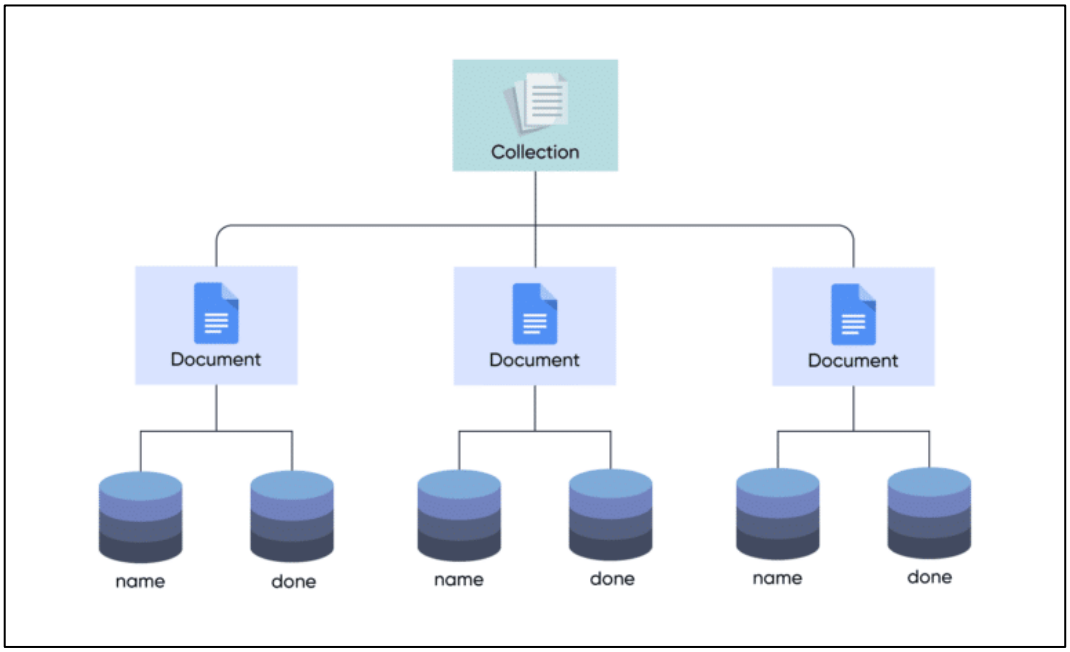
A linguagem gerencia a leitura dos sensores, o processamento dos dados coletados e o acionamento dos atuadores conforme os parâmetros estabelecidos. Por meio dela ocorre também a comunicação com o Firebase, garantindo o envio e recebimento de informações em tempo real entre o protótipo e o aplicativo móvel.

3.1.2.2 Firebase

Firebase foi utilizado como plataforma de comunicação e armazenamento de dados do protótipo, reunindo os serviços de banco de dados em nuvem necessários para integração entre o microcontrolador e o aplicativo móvel. A plataforma garante a sincronização das informações em tempo real e o acesso remoto aos dados coletados.

A plataforma disponibiliza diversos serviços, dos quais foram empregados no protótipo o Firestore Database, o Realtime Database e o Authentication. O Firestore Database é um banco de dados NoSQL orientado a documentos, cuja estrutura é ilustrada na Imagem 21. No protótipo, o Firestore armazena os dados estruturados e históricos do sistema, como informações dos dispositivos, configurações predefinidas e registros de usuários, permitindo a leitura e escrita de forma organizada e escalável.

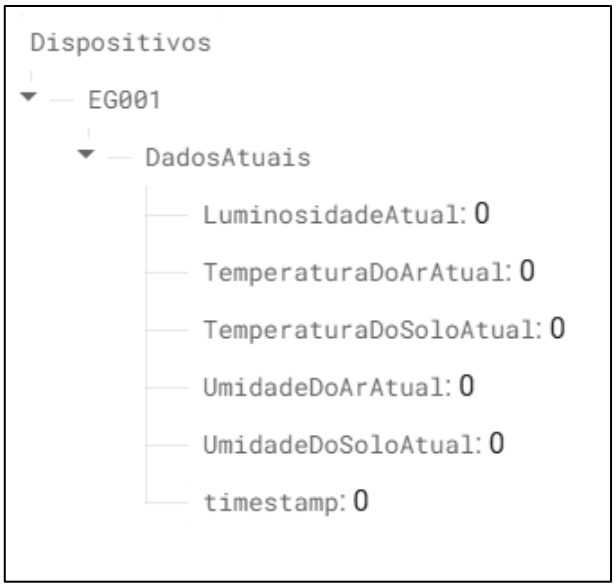
Imagem 21 — Estrutura geral do Firestore Database



Fonte: Mindbowser (2025).

O Firebase Realtime Database é um banco de dados em nuvem que armazena informações em formato *JavaScript Object Notation* (JSON) e mantém sincronização em tempo real entre os dispositivos conectados. Sua estrutura é ilustrada na Imagem 22. No protótipo, registra e atualiza continuamente os valores medidos pelos sensores, como temperatura do ar e solo, umidade do ar e solo, e luminosidade.

Imagem 22 — Estrutura geral do Realtime Database



Fonte: elaborado pelo próprio autor.

Por fim, o Firebase Authentication autêntica e gerencia o acesso dos usuários ao aplicativo

3.1.2.3 React Native

React Native é um *framework* para criar aplicativos móveis para iOS e Android usando JavaScript. Com um único código-fonte, é possível gerar versões do aplicativo para ambas as plataformas. O React Native oferece bibliotecas para navegação, gráficos, gerenciamento de estado e integração com diversos serviços.

O aplicativo móvel foi desenvolvido em React Native com Expo, uma plataforma que fornece ferramentas e serviços para simplificar o desenvolvimento. A interface exibe dados dos sensores em tempo real, mostra o status dos atuadores, permite ajustar valores desejados de temperatura e umidade, e possibilita o gerenciamento de múltiplas estufas cadastradas.

3.2 MÉTODOS

A metodologia de desenvolvimento foi estruturada em etapas sequenciais, iniciando pela prototipagem elétrica e avançando até a comunicação via Firebase. Cada etapa foi planejada para garantir a integração adequada entre *hardware*, *software* e plataformas de comunicação, assegurando o funcionamento de todos os componentes.

3.2.1 Prototipagem

Esta etapa consistiu na definição e validação das conexões elétricas entre sensores, atuadores e o Raspberry Pi, estabelecendo a arquitetura de comunicação e alimentação dos componentes. Inicialmente, cada sensor foi montado individualmente em *protoboard* para validação e, em seguida, realizou-se a integração de todos os componentes simultaneamente ao microcontrolador. Após a validação em *protoboard*, desenvolveu-se uma placa de circuito impresso (PCB), cuja fabricação foi realizada pelo método de transferência térmica e corrosão química.

3.2.2 Montagem física do protótipo

Concluída a prototipagem elétrica, iniciou-se a montagem física dos componentes em uma estrutura que serve como base da estufa. Esta etapa compreendeu a instalação da PCB e dos sensores, além do posicionamento dos atuadores para possibilitar o controle das variáveis ambientais monitoradas.

3.2.3 Lógica embarcada em Python

Com a estrutura física finalizada e todos os componentes conectados, desenvolveuse a lógica embarcada em Python para integrar sensores, atuadores e plataformas de comunicação. A arquitetura do *software* foi organizada em módulos independentes, operando em ciclo contínuo para realizar a leitura dos sensores, processar as informações, acionar os atuadores conforme as regras de controle estabelecidas e manter a sincronização dos dados com o Firebase.

3.2.4 Aplicativo móvel em React Native

O desenvolvimento do aplicativo móvel foi conduzido de forma modular, estruturando as telas de forma isolada, cada uma com funcionalidades específicas. Esta abordagem facilita a implementação de novas funcionalidades, a manutenção e a identificação de possíveis problemas durante o desenvolvimento.

3.2.5 Estruturação do Firebase

A integração com o Firebase foi estruturada para separar dados em tempo real de informações de configuração e registros históricos, utilizando dois serviços complementares. Esta abordagem permite sincronização rápida dos valores dos sensores enquanto mantém as configurações e o histórico organizados para consultas e análises

3.3 TESTES E VALIDAÇÕES

A etapa de testes e validações teve como objetivo verificar o funcionamento das funcionalidades implementadas no protótipo. Foram conduzidos testes individuais de cada módulo e, em seguida, um ciclo experimental para avaliar o desempenho integrado do sistema em operação contínua.

Os testes individuais contemplaram a verificação de funcionalidades específicas, como o controle do fotoperíodo da luminária, a regulação térmica por meio do aquecedor e das ventoinhas em diferentes faixas de temperatura, o controle da umidade do ar e a ativação automatizada da irrigação com base na umidade do solo.

Após essa etapa, foi executado um ciclo experimental de 72 horas, simulando condições de cultivo. O experimento foi dividido em três períodos de 24 horas, correspondentes às fases de germinação, crescimento e floração, cada uma com parâmetros específicos.

Durante os ensaios, o sistema registrou continuamente os dados dos sensores e o estado dos atuadores em arquivo local, permitindo a análise detalhada do comportamento do protótipo e da resposta automática às variações simuladas das condições ambientais.

4 DESENVOLVIMENTO

Este capítulo apresenta o desenvolvimento completo do protótipo, abrangendo a prototipagem elétrica, a montagem física da estrutura, a implementação da lógica embarcada, o desenvolvimento do aplicativo móvel, a integração com Firebase e os testes realizados.

4.1 PROTOTIPAGEM

A prototipagem definiu e validou as conexões elétricas entre os componentes do sistema. Primeiro, os sensores e atuadores foram conectados ao Raspberry Pi e testados. Depois de validar o funcionamento, foi desenvolvida uma placa de circuito impresso para organizar as conexões dos sensores.

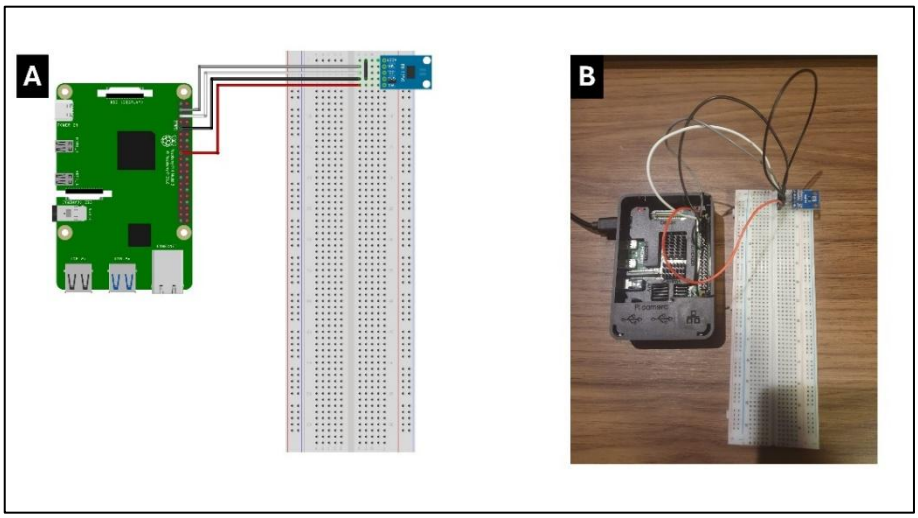
4.1.1 Diagrama de conexão dos sensores

Os sensores foram conectados ao Raspberry Pi seguindo as especificações de cada componente. As subseções a seguir detalham a implementação e validação individual de cada sensor, finalizando com a integração de todos os componentes.

4.1.1.1 Sensor BH1750

O sensor BH1750 foi conectado ao Raspberry Pi através do barramento I²C, utilizando os pinos GPIO 2 (SDA) e GPIO 3 (SCL). A alimentação foi realizada em 3,3 V a partir do pino correspondente do Raspberry Pi. Este sensor utiliza o protocolo I²C, que permite que múltiplos dispositivos compartilhem os mesmos pinos de comunicação através de endereçamento único. O BH1750 possui dois endereços I²C possíveis (0x23 ou 0x5C), definidos pelo estado do pino ADDR: quando conectado ao GND, o endereço é 0x23; quando conectado ao VCC, o endereço é 0x5C. Neste projeto, foi utilizado o endereço padrão 0x23 com o pino ADDR conectado ao GND. A Imagem 23 apresenta o diagrama de conexão e a montagem física na *protoboard*.

Imagem 23 — A) Diagrama de conexão do sensor BH1750; B) Montagem física do sensor BH1750

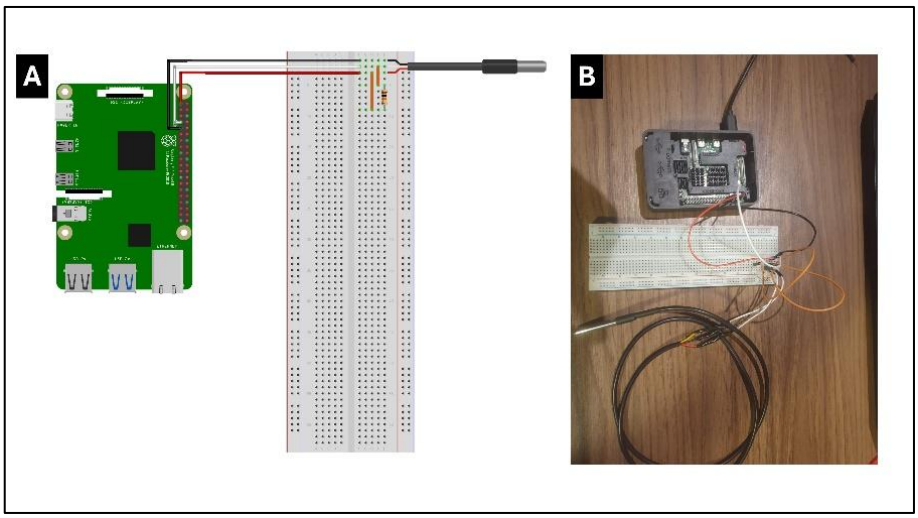


Fonte: elaborado pelo próprio autor.

4.1.1.2 Sensor DS18B20

O sensor DS18B20 foi conectado ao Raspberry Pi utilizando o protocolo 1-Wire, com o pino de dados ligado ao GPIO 4. A alimentação foi realizada em 3,3 V, e um resistor de *pull-up* de 4,7 kΩ foi adicionado conforme requerido pelo protocolo 1-Wire. A Imagem 24 apresenta o diagrama de conexão e a montagem física na *protoboard*.

Imagem 24 — A) Diagrama de conexão do sensor DS18B20; B) Montagem física do sensor DS18B20

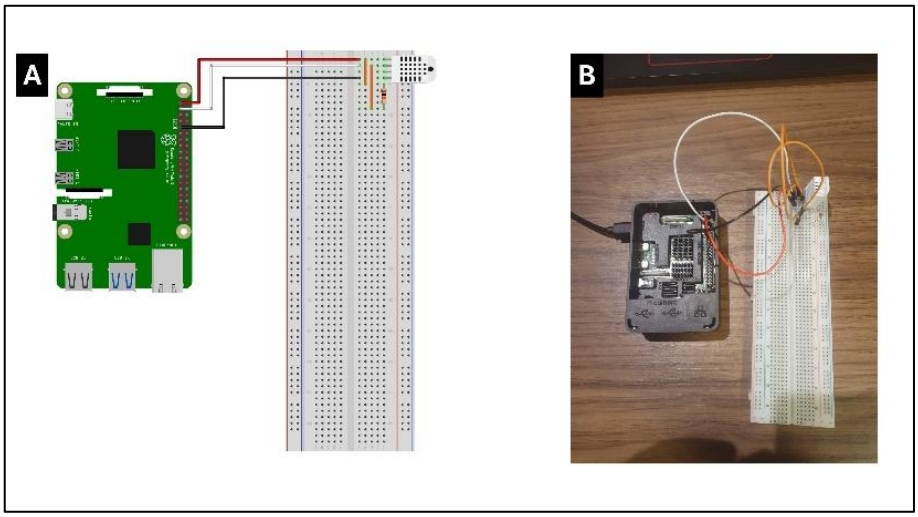


Fonte: elaborado pelo próprio autor.

4.1.1.3 Sensor DHT22

O sensor DHT22 foi conectado provisoriamente ao GPIO 2 do Raspberry Pi. A alimentação foi estabelecida em 3,3 V, e um resistor de *pull-up* de 10 kΩ foi instalado entre o pino de dados e a alimentação. Posteriormente, o sensor foi realocado para o GPIO 17 para evitar conflitos com o barramento I²C utilizado pelos sensores BH1750 e ADS1115. A Imagem 25 apresenta o diagrama de conexão e a implementação prática do DHT22 na *protoboard*.

Imagem 25 — A) Diagrama de conexão do sensor DHT22; B) Montagem física do sensor DHT22

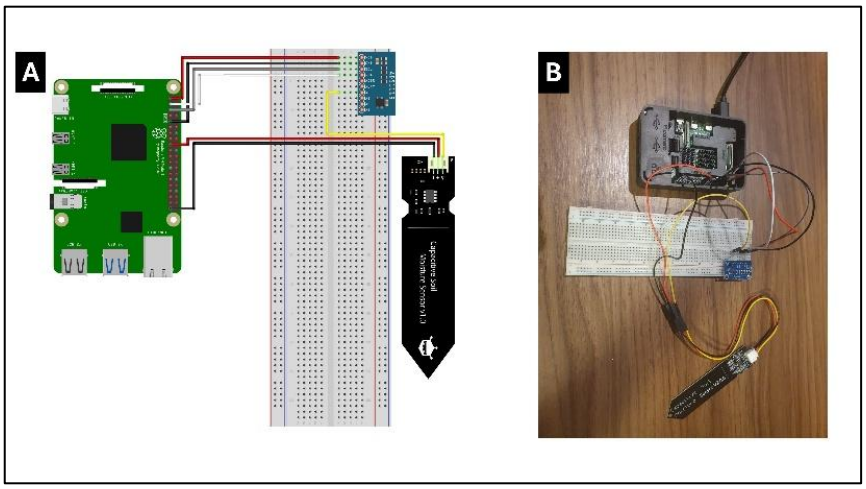


Fonte: elaborado pelo próprio autor.

4.1.1.4 Sensor capacitivo de umidade do solo

O sensor capacitivo de umidade do solo foi conectado ao conversor analógico digital ADS1115, que se comunica com o Raspberry Pi através do barramento I²C. O sinal analógico do sensor foi ligado ao canal A0 do ADS1115. A alimentação foi realizada em 3,3 V. O conversor ADS1115 foi necessário devido à ausência de entradas analógicas no Raspberry Pi e utiliza o endereço I²C 0x48, diferente do BH1750 que utiliza o endereço 0x23. A Imagem 26 apresenta o diagrama de conexão e a montagem física na *protoboard*.

Imagem 26 — A) Diagrama de conexão do sensor capacitivo; B) Montagem física do sensor capacitivo

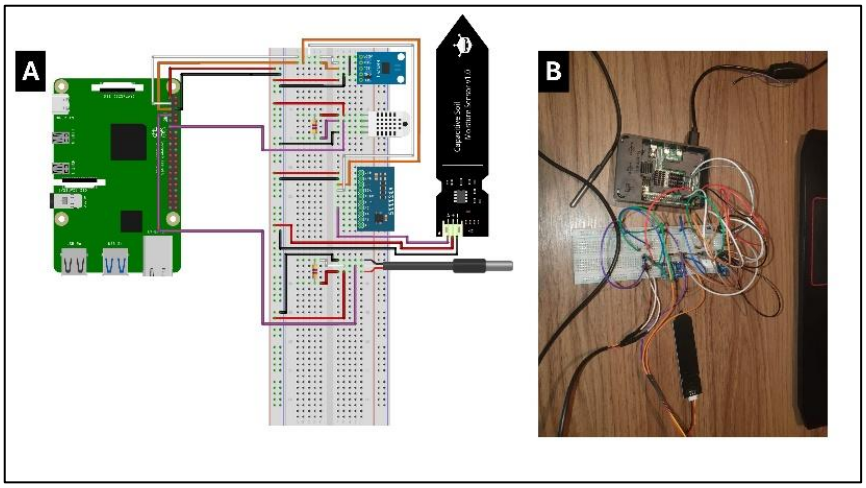


Fonte: elaborado pelo próprio autor.

4.1.1.5 Sensores integrados

Após a validação individual, todos os sensores foram integrados na *protoboard* e conectados simultaneamente ao Raspberry Pi. O BH1750 e o ADS1115 compartilharam os mesmos pinos I²C (SDA e SCL) sem conflitos de endereçamento, o DS18B20 foi conectado ao GPIO 4 através do protocolo 1-Wire, e o DHT22 ao GPIO 17. A Imagem 27 apresenta o diagrama completo e a montagem física dos sensores integrados.

Imagem 27 — A) Diagrama de conexão dos sensores integrados; B) Montagem física dos sensores integrados



Fonte: elaborado pelo próprio autor.

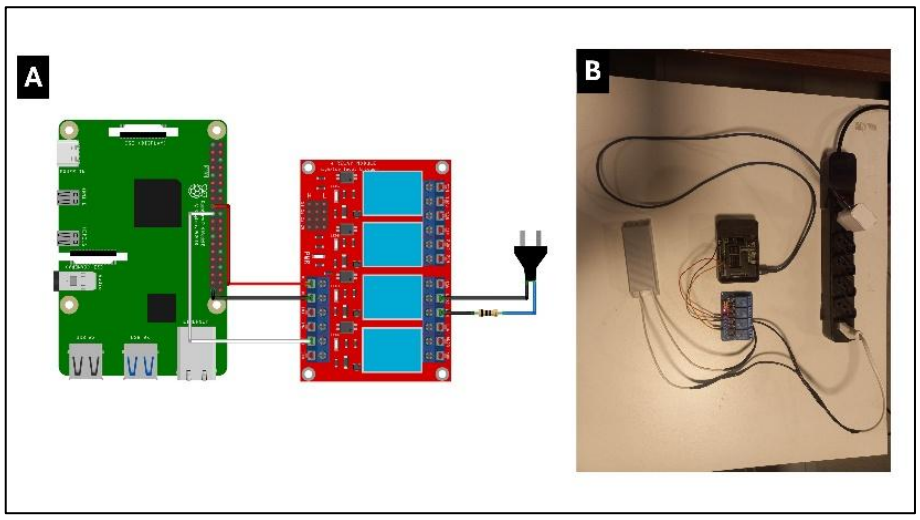
4.1.2 Diagrama de conexão dos atuadores

Os atuadores ajustam as variáveis ambientais dentro da estufa para manter as condições em conformidade com os parâmetros definidos. As subseções a seguir detalham a implementação e validação individual de cada atuador, finalizando com a integração de todos os componentes operando simultaneamente.

4.1.2.1 Resistor de aquecimento

O resistor de aquecimento foi conectado ao Raspberry Pi através do módulo relé de quatro canais, utilizando o GPIO 10 para controle do acionamento. A alimentação do resistor é fornecida em 220 VAC através do relé. A Imagem 28 apresenta o diagrama de conexão e a montagem física na *protoboard*.

Imagem 28 — A) Diagrama de conexão do resistor de aquecimento; B) Montagem física do resistor de aquecimento

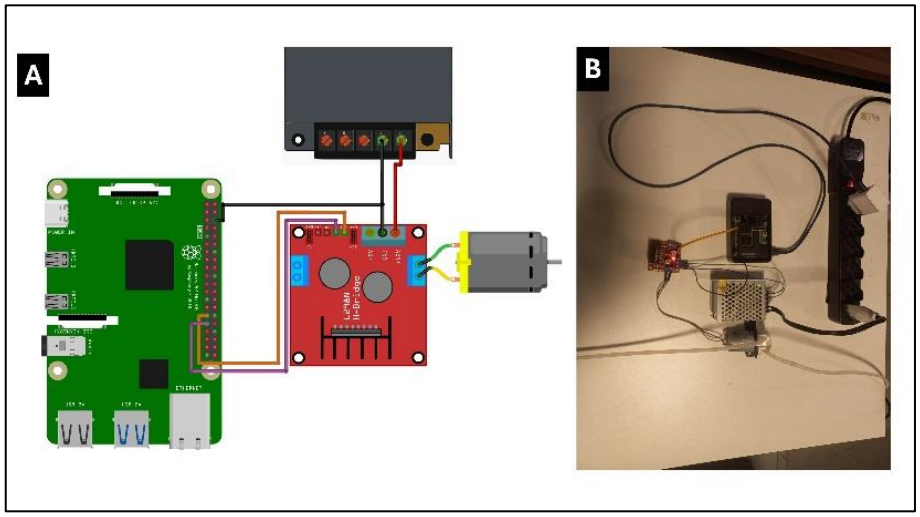


Fonte: elaborado pelo próprio autor.

4.1.2.2 Bomba peristáltica

A bomba peristáltica foi conectada ao Raspberry Pi através do *driver* de motor do tipo Ponte-H L298N, utilizando os pinos GPIO 5 (IN1) e GPIO 6 (IN2) para controle do acionamento. A alimentação da bomba é fornecida em 12 VDC. A Imagem 29 apresenta o diagrama de conexão e a montagem física na *protoboard*.

Imagem 29 — A) Diagrama de conexão da bomba peristáltica; B) Montagem física da bomba peristáltica

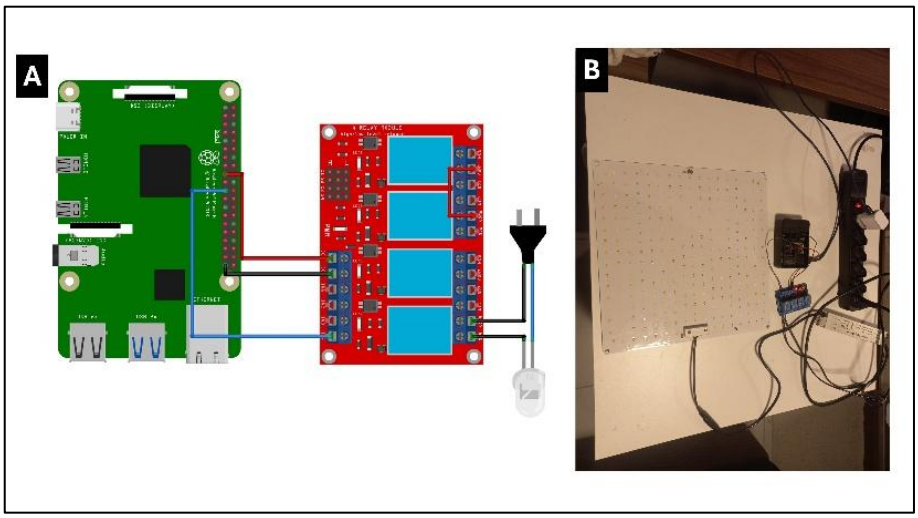


Fonte: elaborado pelo próprio autor.

4.1.2.3 Luminária led

A luminária LED foi conectada ao Raspberry Pi através do módulo relé de quatro canais, utilizando o GPIO 9 para controle do acionamento. A alimentação da luminária é fornecida em 220 VAC através do relé. A Imagem 30 apresenta o diagrama de conexão e a montagem física na *protoboard*.

Imagem 30 — A) Diagrama de conexão da luminária LED; B) Montagem física da luminária LED

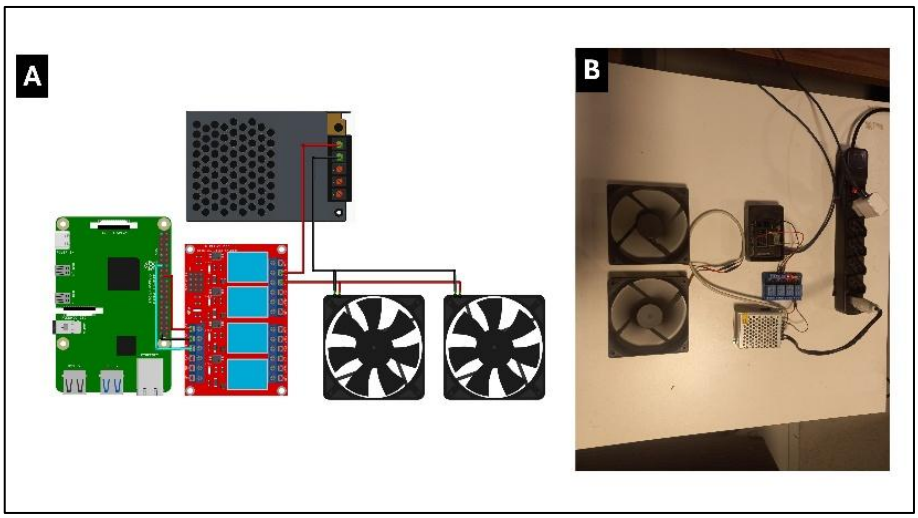


Fonte: elaborado pelo próprio autor.

4.1.2.4 Ventoinha

As ventoinhas foram conectadas ao Raspberry Pi através do módulo relé de quatro canais, utilizando o GPIO 27 para controle do acionamento simultâneo. A alimentação das ventoinhas é fornecida em 12 VDC. A Imagem 31 apresenta o diagrama de conexão e a montagem física na *protoboard*.

Imagem 31 — A) Diagrama de conexão das ventoinhas; B) Montagem física das ventoinhas

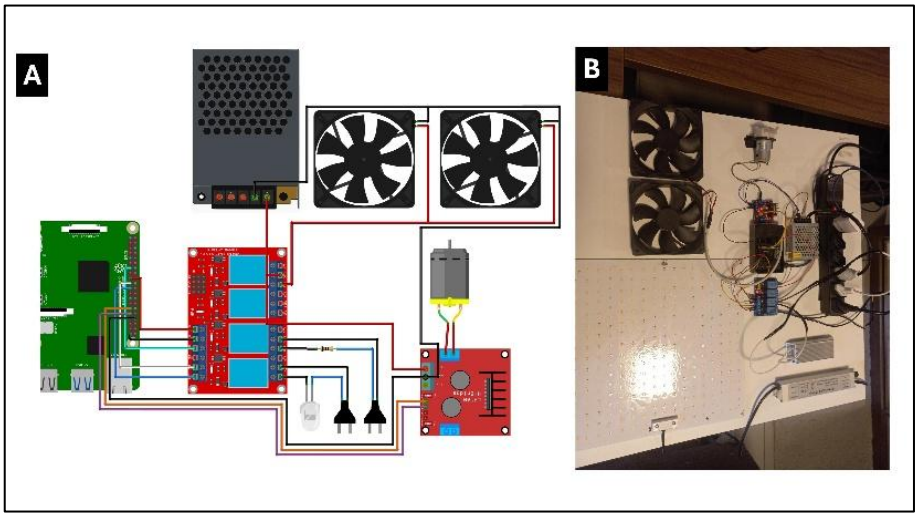


Fonte: elaborado pelo próprio autor.

4.1.2.5 Atuadores integrados

Todos os atuadores foram integrados e conectados ao Raspberry Pi. O módulo relé de quatro canais centraliza o acionamento do resistor de aquecimento (GPIO 10), da luminária LED (GPIO 9) e das ventoinhas (GPIO 27). A bomba peristáltica foi conectada através do *driver* L298N (GPIOs 5 e 6). A Imagem 32 apresenta o diagrama completo e a montagem física dos atuadores integrados.

Imagem 32 — A) Diagrama completo de integração dos atuadores; B) Montagem física completa dos atuadores integrados

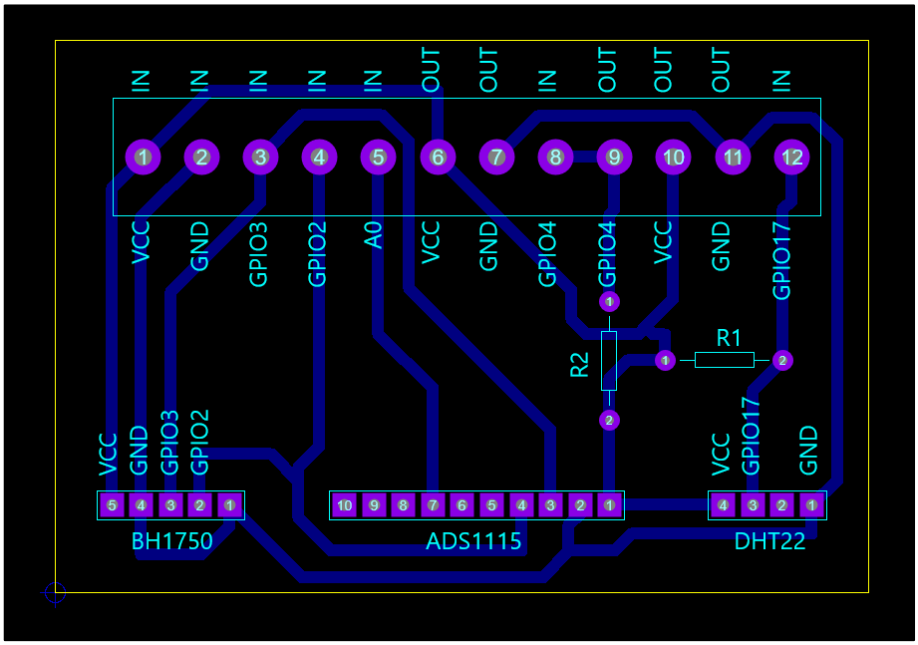


Fonte: elaborado pelo próprio autor.

4.1.3 Desenvolvimento da PCB

Após a validação da prototipagem na *protoboard*, foi desenvolvida uma PCB para organizar as conexões dos sensores. O projeto foi elaborado no *software* Proteus, respeitando as conexões validadas anteriormente. A Imagem 33 apresenta o *layout* da PCB projetada.

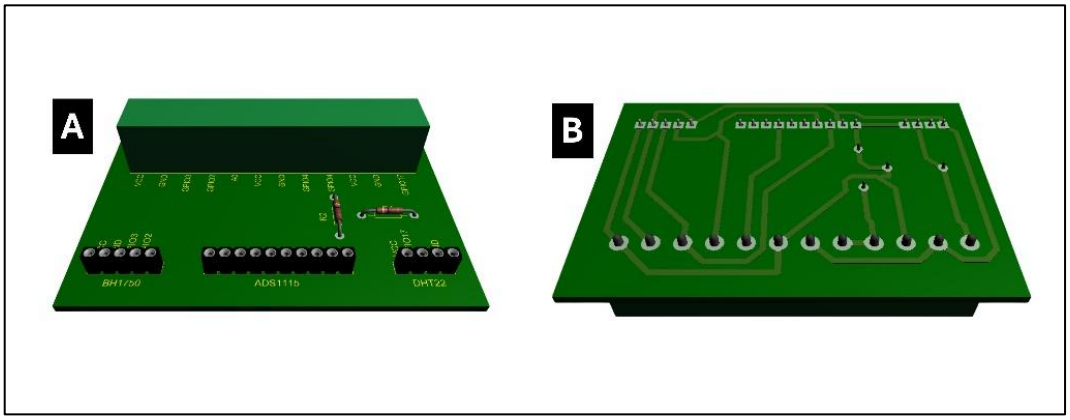
Imagem 33 — *Layout* da PCB projetada no Proteus



Fonte: elaborado pelo próprio autor.

A Imagem 34 apresenta as visualizações tridimensionais do projeto, mostrando a disposição dos componentes e o roteamento das trilhas.

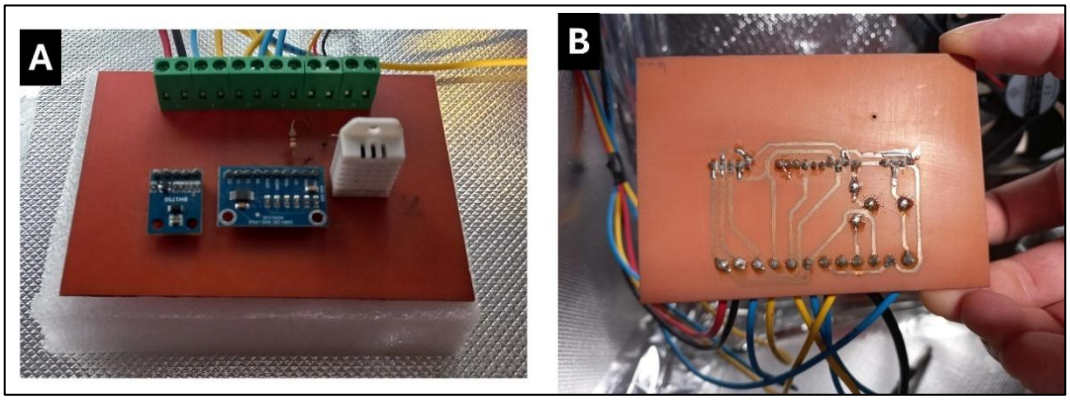
Imagem 34 — A) Vista superior da PCB projetada; B) Vista inferior da PCB projetada



Fonte: elaborado pelo próprio autor.

A fabricação foi realizada pelo método de transferência térmica e corrosão química. O *layout* foi impresso em papel fotográfico e transferido para uma placa de fenolite revestida com cobre por meio de transferência térmica manual. A placa foi corroída com percloreto de ferro, lavada, e o toner residual foi removido com álcool. Após a corrosão, os sensores e conectores foram soldados. A Imagem 35 apresenta a PCB finalizada.

Imagem 35 — A) Vista superior da PCB finalizada; B) Vista inferior da PCB finalizada



Fonte: elaborado pelo próprio autor.

4.2 MONTAGEM FÍSICA DO PROTÓTIPO

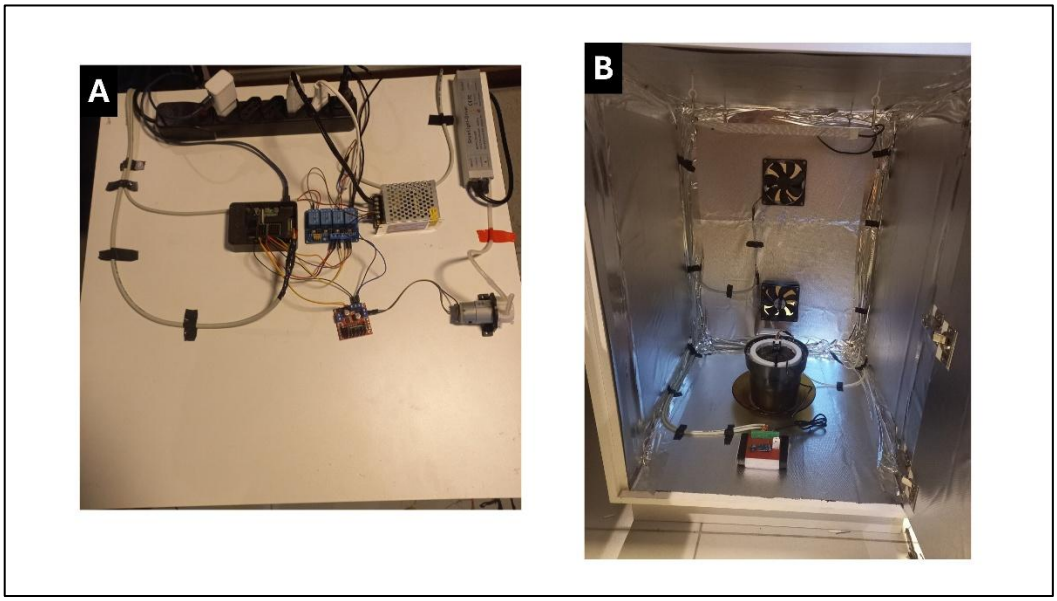
Após a validação dos componentes, foi realizada a montagem física do protótipo em uma estrutura de estufa construída em madeira, com dimensões de 60 cm de largura, 60 cm de profundidade e 75 cm de altura, definidas para acomodar adequadamente todos os dispositivos do sistema.

A PCB fabricada foi posicionada na parte frontal inferior da estufa para centralizar as conexões dos sensores. Os sensores DS18B20 e capacitivo de umidade foram conectados à PCB e inseridos diretamente no vaso de cultivo.

A luminária LED foi instalada no teto da estufa, enquanto as ventoinhas foram posicionadas na parede posterior para circulação do ar. O resistor de aquecimento foi fixado na parte inferior da estrutura, e a bomba peristáltica foi instalada externamente.

O sistema de controle foi organizado na parte superior externa da estufa, onde o Raspberry Pi, o módulo relé de quatro canais, o *driver* L298N e a fonte chaveada de 12 V foram dispostos de forma acessível para facilitar a manutenção e o monitoramento. A Imagem 36 apresenta as vistas interna e superior do protótipo finalizado.

Imagem 36 — A) Vista superior da estufa; B) Vista interna da estufa



Fonte: elaborado pelo próprio autor.

4.3 LÓGICA EMBARCADA EM PYTHON

A implementação do sistema de controle embarcado foi desenvolvida em Python, estruturando-se em módulos onde cada sensor e atuador foi encapsulado em classe dedicada. Este capítulo apresenta as lógicas de leitura dos sensores, as lógicas de controle dos atuadores e o ciclo principal de funcionamento do sistema.

4.3.1 Lógicas de leitura dos sensores

Os sensores monitoram continuamente as condições ambientais da estufa. Cada sensor foi implementado em classe dedicada, contendo métodos específicos para leitura de dados e tratamento de possíveis falhas de comunicação.

4.3.1.1 Lógica do sensor BH1750

O sensor BH1750 mede a iluminância através do protocolo I²C no endereço 0x23, conectado aos pinos GPIO 2 (SDA) e GPIO 3 (SCL) do Raspberry Pi. A implementação, apresentada no Apêndice A.1, utiliza a biblioteca adafruit_bh1750 para comunicação com o dispositivo.

Na inicialização, a classe do sensor tenta estabelecer a conexão I²C através de board.I2C(). Caso o sensor não seja detectado no barramento, a exceção é capturada e o atributo self.sensor recebe None, permitindo que o sistema continue operando. Nas tentativas subsequentes de leitura, o método ler_luminosidade() verifica se o sensor foi inicializado corretamente; caso não tenha sido inicializado, retorna None, evitando erros de execução. Quando o sensor está operacional, o método acessa o atributo lux e retorna o valor medido.

4.3.1.2 Lógica do sensor DS18B20

O sensor DS18B20 mede a temperatura do solo através do protocolo 1-Wire, permitindo comunicação serial utilizando apenas um fio de dados além da alimentação e referência. A implementação, apresentada no Apêndice A.2, utiliza a biblioteca w1thermsensor para comunicação com o dispositivo.

Na inicialização, a classe tenta estabelecer a conexão com o sensor através de W1ThermSensor(). Caso o sensor não seja detectado, a exceção é capturada e o atributo self.sensor recebe None, permitindo que o sistema continue operando. Nas tentativas subsequentes de leitura, o método read_temp() verifica se o sensor foi inicializado corretamente; caso não tenha sido inicializado, retorna None, evitando erros de execução. Quando o sensor está operacional, o método acessa get_temperature() e retorna a temperatura medida.

4.3.1.3 Lógica do sensor DHT22

O sensor DHT22 mede simultaneamente a temperatura e umidade relativa do ar. A implementação, apresentada no Apêndice A.3, utiliza a biblioteca adafruit_dht para comunicação com o dispositivo.

Na inicialização, a classe é configurada através de adafruit_dht.DHT22(pin), recebendo como parâmetro o pino GPIO utilizado. O método ler_dados() acessa os atributos temperature e humidity do sensor, retornando uma tupla (par ordenado de valores) contendo temperatura e umidade medidas. Caso algum dos valores seja inválido, o método retorna None para ambas as variáveis. A implementação incorpora tratamento específico para três tipos de exceções: RuntimeError para falhas de leitura

do protocolo, OverflowError para valores fora da faixa suportada, e Exception para erros inesperados. Este tratamento robusto garante que falhas de comunicação, comuns neste tipo de sensor, não interrompam o ciclo de controle. O método close() finaliza a comunicação com o sensor de forma segura.

4.3.1.4 Lógica do sensor capacitivo de umidade do solo

O sensor capacitivo mede a umidade do solo através da variação da capacitância. A implementação, apresentada no Apêndice A.4, utiliza o conversor analógico-digital ADS1115 de 16 bits para processar o sinal do sensor capacitivo através do protocolo I²C.

Na inicialização, a classe estabelece a conexão I²C através de busio.I2C() e configura o conversor ADS1115. O sensor capacitivo é conectado ao canal A0 do conversor, sendo acessado através de AnalogIn(). Caso o conversor não seja detectado, a exceção é capturada e o atributo self.canal_umidade recebe None, permitindo que o sistema continue operando. A classe recebe como parâmetros os valores de calibração seco e molhado, que representam as leituras do conversor em solo completamente seco e saturado, respectivamente.

O método ler_umidade() verifica se o canal foi inicializado corretamente; caso não tenha sido inicializado, retorna None. Quando operacional, o método lê o valor digital do conversor e aplica normalização Min-Max para transformar a leitura em percentual de umidade. A normalização Min-Max é uma técnica de pré-processamento que redimensiona valores para um intervalo específico, calculada pela Equação (1).

$$X_{norm} = \frac{(X - X_{min})}{(X_{max} - X_{min})} \quad (1)$$

Onde X é o valor original da leitura do conversor ADC, Xmin é o valor de calibração em solo molhado e Xmax é o valor de calibração em solo seco. O resultado normalizado é multiplicado por 100 e subtraído de 100 para inverter a escala, pois o sensor apresenta comportamento inverso onde valores menores do ADC correspondem a maior umidade. O percentual final é limitado ao intervalo de 0% a 100% e arredondado para duas casas decimais para padronização dos dados.

Os valores de calibração foram obtidos experimentalmente através de leituras do sensor em solo seco e saturado, estabelecendo os limites da escala de conversão.

4.3.2 Lógicas de controle dos atuadores

Os atuadores são responsáveis por intervir no ambiente da estufa conforme os parâmetros de cultivo estabelecidos. Cada atuador foi implementado em classe dedicada, contendo métodos de acionamento, desligamento e lógica de decisão baseada nas leituras dos sensores e nos parâmetros de configuração do sistema.

4.3.2.1 Controle do resistor de aquecimento

O resistor de aquecimento é controlado através de módulo relé conectado ao GPIO 10 do Raspberry Pi. A implementação, apresentada no Apêndice A.5, opera com lógica invertida, onde nível lógico baixo ativa o aquecimento e nível alto desativa.

Na inicialização, a classe configura o pino GPIO através de GPIO.setup() e executa o método desligar(), garantindo que o resistor inicie desligado por segurança. Os métodos ligar() e desligar() controlam diretamente o estado do relé através de GPIO.output(), com tratamento de exceções para capturar possíveis falhas.

O método controlar() implementa a lógica de decisão baseada na temperatura do ar e na configuração armazenada no banco de dados. Primeiramente, verifica-se a validade da configuração e da leitura do sensor. Qualquer valor inválido resulta no desligamento do resistor. O sistema suporta dois modos de operação: controle por temperatura desejada específica, definida pelo usuário via aplicativo móvel, ou controle por faixa térmica delimitada pelos parâmetros TemperaturaMin e TemperaturaMax armazenados no banco de dados.

No modo de temperatura desejada, o resistor ativa quando a temperatura medida encontra-se abaixo do valor alvo. No modo automático, o resistor aciona quando a temperatura cai abaixo do limite mínimo e desliga quando atinge ou excede o limite máximo. Temperaturas dentro da faixa intermediária mantêm o resistor desligado. O método retorna uma tupla contendo o estado *booleano* do atuador e uma *string* descritiva justificando a decisão, facilitando monitoramento do sistema.

4.3.2.2 Controle da luminária

A luminária LED é controlada através de módulo relé conectado ao GPIO 9 do Raspberry Pi. A implementação, apresentada no Apêndice A.6, opera com lógica invertida, onde nível lógico baixo ativa a iluminação e nível alto desativa.

Na inicialização, a classe configura o pino GPIO através de GPIO.setup() e executa o método desligar(), garantindo que a luminária inicie desligada por segurança. Os métodos ligar() e desligar() controlam diretamente o estado do relé através de GPIO.output(), com tratamento de exceções para capturar possíveis falhas. A classe define como constante o horário de início da iluminação em HORA_INICIO = "06:00".

O método controlar() implementa a lógica de decisão baseada no conceito de fotoperíodo, parâmetro que define a duração diária de iluminação. Primeiramente, verifica-se a validade da configuração. Caso inválida, a luminária é desligada. O sistema obtém o valor de fotoperíodo (em horas) armazenado no banco de dados e calcula o horário de término somando a duração ao horário de início fixo.

A lógica de controle avalia o fotoperíodo configurado para determinar o acionamento da luminária. Quando o fotoperíodo é igual ou superior a 24 horas, a luminária permanece ligada continuamente. Para fotoperíodos inferiores a 18 horas, cujo horário de término ocorre no mesmo dia do início, a luminária é acionada apenas durante o intervalo programado. Já para fotoperíodos entre 18 e 24 horas, que se estendem além da meia-noite, a luminária permanece ligada tanto no período noturno após o horário de início quanto no período matinal anterior ao horário de término. O método retorna uma tupla contendo o estado *booleano* do atuador e uma *string* descritiva do período de iluminação.

4.3.2.3 Controle das ventoinhas

As ventoinhas são controladas através de módulo relé conectado ao GPIO 27 do Raspberry Pi. A implementação, apresentada no Apêndice A.7, opera com lógica invertida, onde nível lógico baixo ativa a ventilação e nível alto desativa.

Na inicialização, a classe configura o pino GPIO através de GPIO.setup() e executa o método desligar(), garantindo que as ventoinhas iniciem desligadas por segurança. Os métodos ligar() e desligar() controlam diretamente o estado do relé através de GPIO.output(), com tratamento de exceções para capturar possíveis falhas.

O método controlar() implementa lógica de decisão baseada em três parâmetros: temperatura, umidade e estado do resistor de aquecimento. O sistema opera com prioridades definidas, onde a prioridade máxima é atribuída ao resistor de aquecimento. Sempre que este se encontra ativo, as ventoinhas ligam para distribuir o calor dentro da estufa. Na ausência de aquecimento ativo, o sistema verifica a validade das leituras dos sensores, resultando no desligamento das ventoinhas caso os valores sejam inválidos.

O sistema avalia sequencialmente diferentes parâmetros de controle. Primeiramente, verifica-se a umidade desejada configurada pelo usuário via aplicativo móvel. Se definida e a umidade medida a exceder, as ventoinhas acionam. Caso contrário, o sistema verifica se existe temperatura desejada definida pelo usuário. Quando a temperatura medida excede esse valor, as ventoinhas acionam. Na ausência de valores desejados pelo usuário, o sistema utiliza os limites automáticos armazenados no banco de dados. Neste caso, as ventoinhas acionam se a temperatura ou a umidade atingirem ou ultrapassarem seus respectivos limites máximos. Quando nenhuma dessas condições é satisfeita, as ventoinhas permanecem desligadas. O método retorna uma tupla contendo o estado *booleano* do atuador e uma *string* descritiva justificando a decisão.

4.3.2.4 Controle da bomba peristáltica

A bomba peristáltica é controlada através do *driver* de ponte H L298N conectado aos GPIOs 5 (IN1) e 6 (IN2) do Raspberry Pi. A implementação, apresentada no Apêndice A.8, utiliza dois pinos para controlar o sentido de rotação do motor, onde IN1 em nível alto e IN2 em nível baixo acionam a bomba.

Na inicialização, a classe da bomba configura ambos os pinos GPIO através de GPIO.setup() e garante que a bomba inicie desligada, mantendo ambos os pinos em nível baixo. A classe define três constantes: vazão de 1,66 mL/s, volume padrão

de 5 mL por irrigação e período de reação de 120 segundos. O método ligar() recebe como parâmetro a duração da irrigação em segundos e agenda automaticamente o desligamento através de um temporizador. Durante o processo de irrigação, o sistema registra a fase atual do cultivo e bloqueia novos comandos enquanto a bomba estiver ativa.

O método controlar() implementa lógica de decisão que gerencia cinco estados distintos. O primeiro estado corresponde ao bloqueio de novos comandos durante a irrigação ativa. Quando a bomba está irrigando, o sistema ignora novos comandos até conclusão do ciclo, exceto em caso de mudança de fase de cultivo no modo automático. O segundo estado verifica a validade da configuração e da umidade do solo. Valores inválidos resultam no desligamento imediato da bomba.

O terceiro estado implementa período de reação pós-irrigação. Após conclusão de uma irrigação, o sistema aguarda 120 segundos antes de permitir novo acionamento, tempo estimado para que a irrigação anterior seja sensibilizada pelo sensor capacitivo. O quarto estado avalia se existe umidade desejada definida pelo usuário via aplicativo móvel. Quando definida e a umidade medida é inferior ao valor alvo, a bomba aciona pelo tempo calculado baseado na vazão.

O quinto estado opera em modo automático utilizando limites armazenados no banco de dados. Quando a umidade do solo cai abaixo do limite mínimo configurado, a bomba aciona. Se a umidade excede o limite máximo, a bomba permanece desligada para evitar saturação do solo. Valores dentro da faixa adequada mantêm a bomba desligada. Quando a irrigação é concluída, o ciclo principal de leitura dos sensores é executado imediatamente, permitindo que o sistema avalie rapidamente os efeitos da irrigação. O método retorna uma tupla contendo o estado *booleano* do atuador e uma *string* descritiva justificando a decisão.

4.3.3 Ciclo principal de funcionamento

O ciclo principal foi desenvolvido com o intuito de coordenar o sistema embarcado, ele é responsável por executar sequencialmente todas as operações de leitura dos sensores, processamento de controle e sincronização com o banco de

dados. A implementação, apresentada no Apêndice A.9, executa continuamente em loop através da função ciclo_estufa().

O sistema opera com iterações de 30 segundos. Cada iteração do ciclo executa seis etapas sequenciais. Primeiramente, o sistema carrega a configuração local da estufa para refletir o estado atual do banco de dados, a configuração é armazenada em arquivo JSON que reúne as informações da planta em cultivo, a fase atual do ciclo e os parâmetros de controle, obtidos do Firestore Database e ajustados conforme as configurações definidas pelo usuário.

Na segunda etapa, verifica-se a necessidade de avanço automático de fase. O sistema compara o tempo decorrido desde início da fase corrente com duração programada. Quando o tempo de cultivo na fase atual iguala ou excede a duração especificada, o sistema avança automaticamente para a fase subsequente. As fases de cultivo são definidas sequencialmente como "germinação", "crescimento", "floração"e "colheita".

A terceira etapa executa a leitura dos quatro sensores através da função coletar_dados(). Os valores obtidos são arredondados para duas casas decimais para padronização dos dados e são organizados em estrutura de dados contendo os valores atuais e o momento em que foram coletados. Ainda na função, os valores válidos são adicionados a *buffers* para posterior cálculo de médias periódicas.

A quarta etapa processa as lógicas de controle dos quatro atuadores através da função controlar_atuadores(). Esta função executa as lógicas de controle apresentadas na Subseção 4.3.2.

A quinta etapa atualiza o estado de cada atuador no Firestore Database, permitindo visualização em tempo real pelo aplicativo móvel.

A sexta etapa envia os dados atuais dos sensores para o Realtime Database, mantendo disponíveis os valores mais recentes de cada variável.

Após a conclusão de todas as etapas, o sistema aguarda o período configurado até a próxima iteração. A implementação permite que eventos externos interrompam essa espera e forcem a execução antecipada do ciclo. Essa interrupção é realizada por meio de um mecanismo de sinalização entre *threads*, utilizando eventos internos.

Assim, alterações realizadas pelo usuário no aplicativo móvel são refletidas rapidamente no comportamento do sistema.

Paralelamente ao ciclo principal, um processo secundário executa envio periódico de médias históricas. A cada 5 minutos, este processo calcula médias dos valores armazenados nos *buffers* dos sensores e envia o conjunto de médias para o Firestore Database. Esta estratégia reduz volume de escritas no banco de dados mantendo histórico representativo das condições de cultivo.

Caso a comunicação com o Firebase falhe, o ciclo principal mantém a execução normalmente. As leituras de sensores, o processamento das lógicas de controle e o acionamento dos atuadores continuam sendo realizados com base na última configuração válida armazenada no arquivo de configuração local.

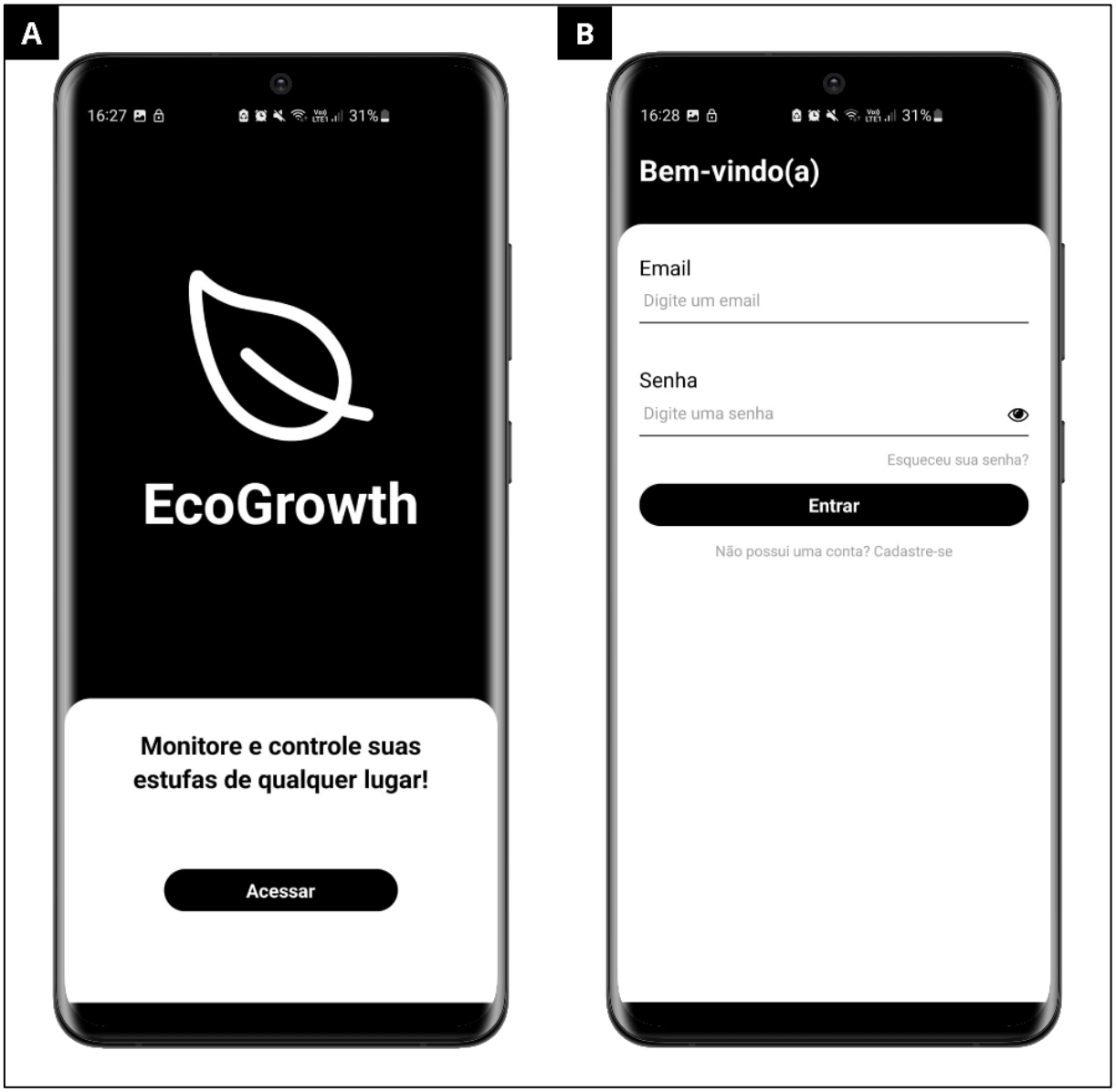
4.4 APLICATIVO MÓVEL EM REACT NATIVE

O aplicativo móvel foi desenvolvido em React Native utilizando o *framework* Expo, proporcionando interface para monitoramento e controle remoto das estufas. O sistema possibilita ao usuário visualizar dados dos sensores, analisar gráficos de tendências, ajustar parâmetros de cultivo e acessar outras funcionalidades apresentadas nas subseções seguintes.

4.4.1 Fluxo de autenticação

O fluxo de autenticação foi implementado utilizando Firebase Authentication para gerenciar o acesso ao sistema. Ao iniciar o aplicativo, o usuário é apresentado à primeira tela do protótipo. Em seguida, os usuários acessam a tela de *login* para autenticação. A Imagem 37 apresenta as telas iniciais do fluxo de autenticação.

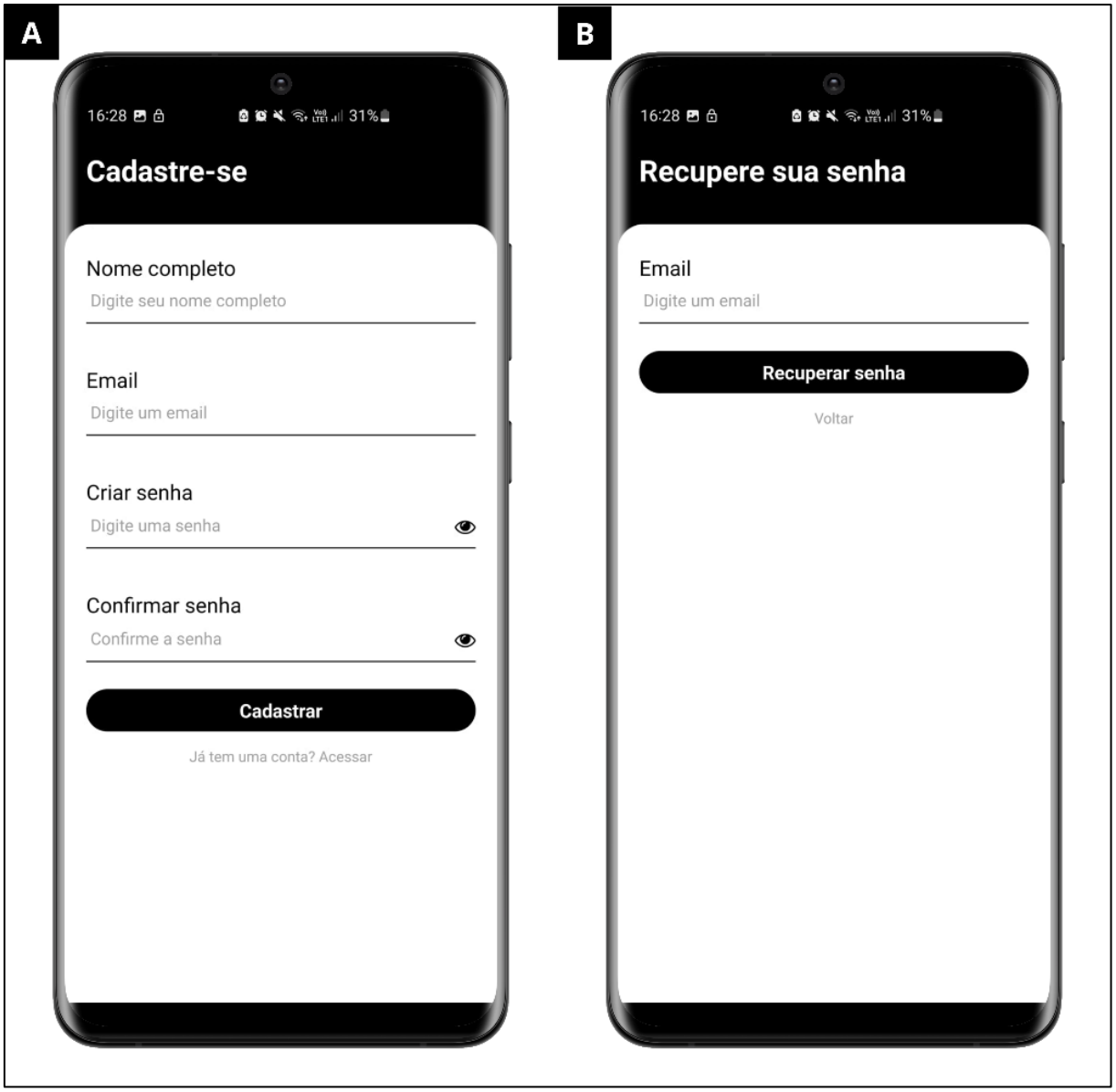
Imagem 37 — A) Tela de apresentação; B) Tela de *login*



Fonte: elaborado pelo próprio autor.

Para novos usuários, o sistema disponibiliza tela de cadastro com campos para inserção de nome completo, endereço de *e-mail* e senha com confirmação. O aplicativo também oferece funcionalidade de recuperação de senha através do envio de *link* de redefinição para o *e-mail* cadastrado. A Imagem 38 apresenta as telas de cadastro e recuperação de senha.

Imagem 38 — A) Tela de cadastro; B) Tela de recuperação de senha



Fonte: elaborado pelo próprio autor.

4.4.2 Tela “Minhas Estufas”

Após a autenticação, o usuário acessa a tela "Minhas Estufas", que centraliza o gerenciamento das estufas cadastradas. A interface apresenta dois estados, exibindo mensagem informativa com orientação para cadastro quando não há dispositivos vinculados, ou apresentando as estufas com nome e código identificador. A barra inferior disponibiliza acesso à listagem de estufas, cadastro de novos dispositivos e configurações da conta através de três ícones específicos. A Imagem 39 apresenta os dois estados da interface.

Imagem 39 — A) Tela sem estufa cadastrada; B) Tela com estufa cadastrada

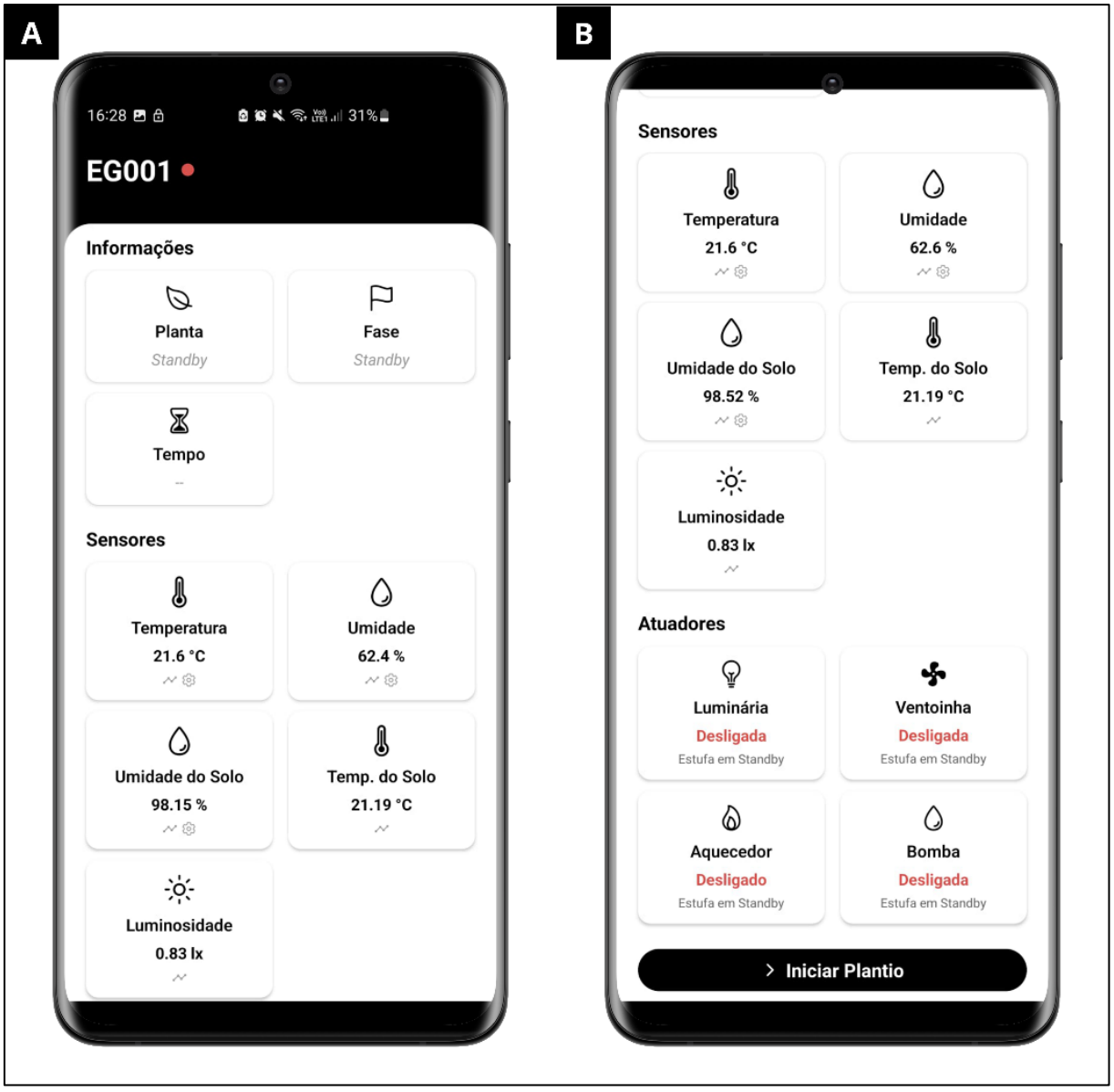
Fonte: elaborado pelo próprio autor.

4.4.3 Tela “Estufa Selecionada”

Ao selecionar uma estufa, o usuário acessa a tela principal de monitoramento, organizada em três seções distintas com informações do cultivo, leituras dos sensores e estados dos atuadores. No modo *standby*, os campos de informações e atuadores são apresentados com indicação *standby*, enquanto os sensores permanecem

monitorando o ambiente da estufa. A Imagem 40 apresenta a interface no modo *standby*.

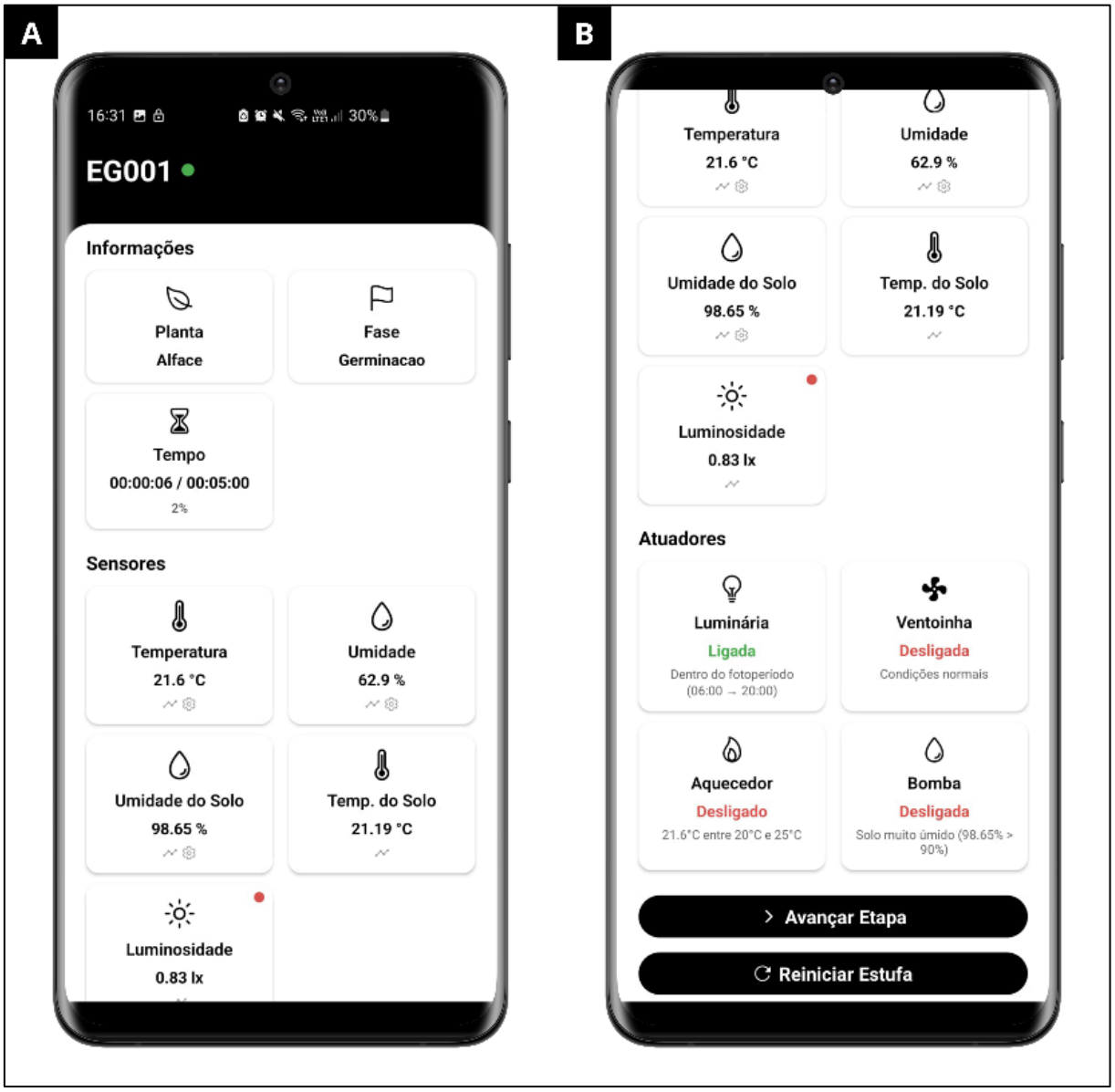
Imagem 40 — A) Tela de informações e sensores em *standby*; B) Tela sensores e atuadores em *standby*



Fonte: elaborado pelo próprio autor.

Com cultivo ativo, a interface exibe nome da planta, fase de desenvolvimento e temporizador de progresso referente a planta selecionada na tela de "iniciar plantio", enquanto os atuadores apresentam seus estados operacionais com justificativas técnicas. A interface disponibiliza botões para avanço de fase e reinicialização da estufa. A Imagem 41 apresenta a interface com cultivo ativo.

Imagem 41 — A) Tela de informações e sensores em operação; B) Tela sensores e atuadores em operação



Fonte: elaborado pelo próprio autor.

4.4.4 Tela “Iniciar Plantio”

A tela "Iniciar Plantio" permite ao usuário configurar um novo ciclo de cultivo através de campos de seleção para escolha da espécie e fase inicial. Os ciclos disponíveis são previamente definidos no banco de dados do sistema. Após a confirmação, o sistema carrega automaticamente os parâmetros correspondentes à configuração selecionada, iniciando o controle automatizado da estufa. A Imagem 42 apresenta a tela de configuração do plantio.

Imagem 42 — A) Tela iniciar plantio; B) Tela de seleção de fase

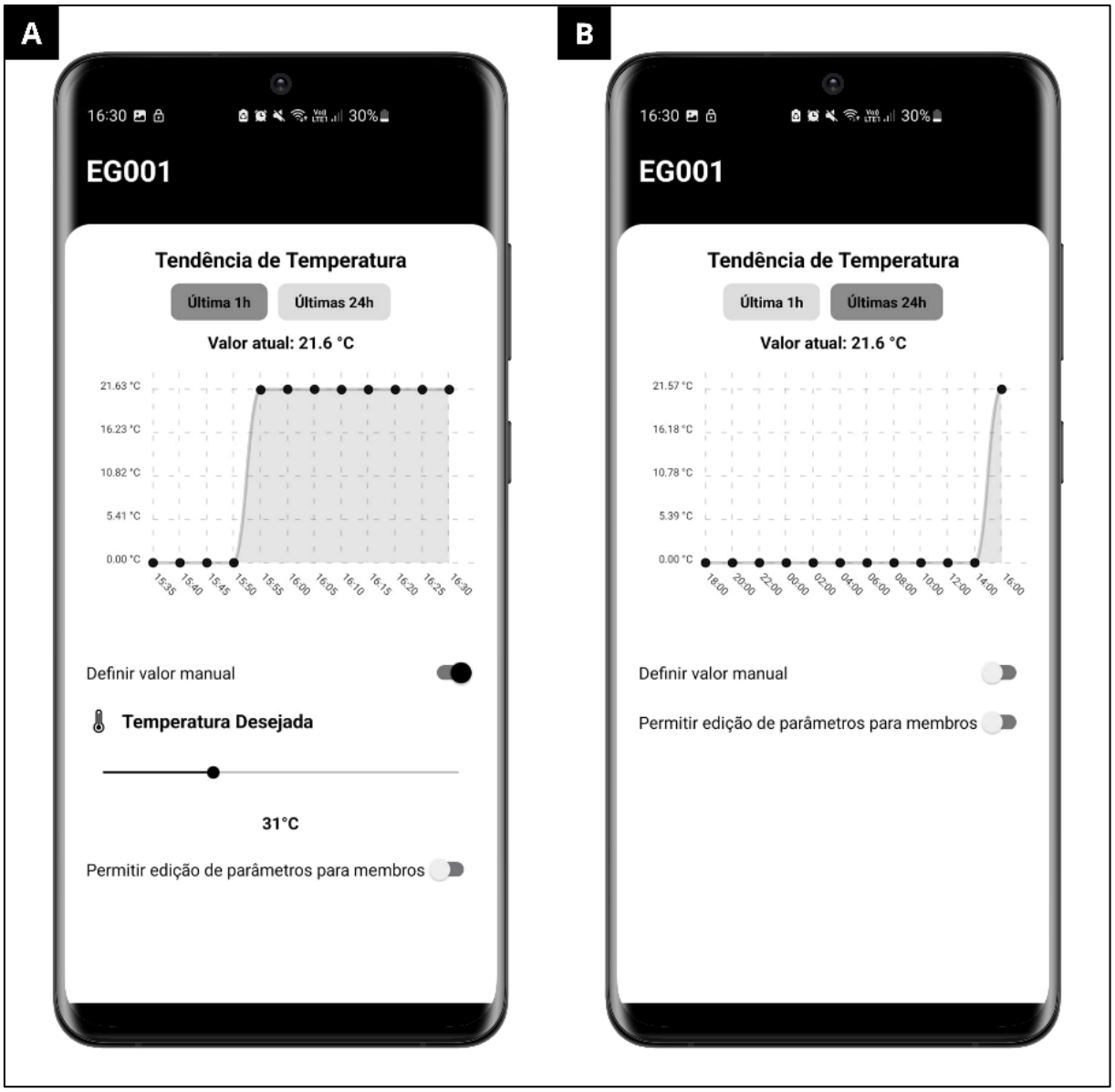
Fonte: elaborado pelo próprio autor.

4.4.5 Tela de gráficos

Cada sensor disponível na tela da estufa selecionada pode ser expandido para visualização detalhada através de gráficos de tendência, permitindo a troca entre períodos de última hora e últimas 24 horas para análise de variações de curto e longo prazo. Os gráficos são atualizados conforme a lógica de controle. A interface disponibiliza controles para ajuste manual dos parâmetros através de comutador e controles deslizantes, permitindo personalização conforme necessidades específicas.

A Imagem 43 apresenta os gráficos de temperatura do ar nos dois períodos disponíveis e a interface de ajuste manual.

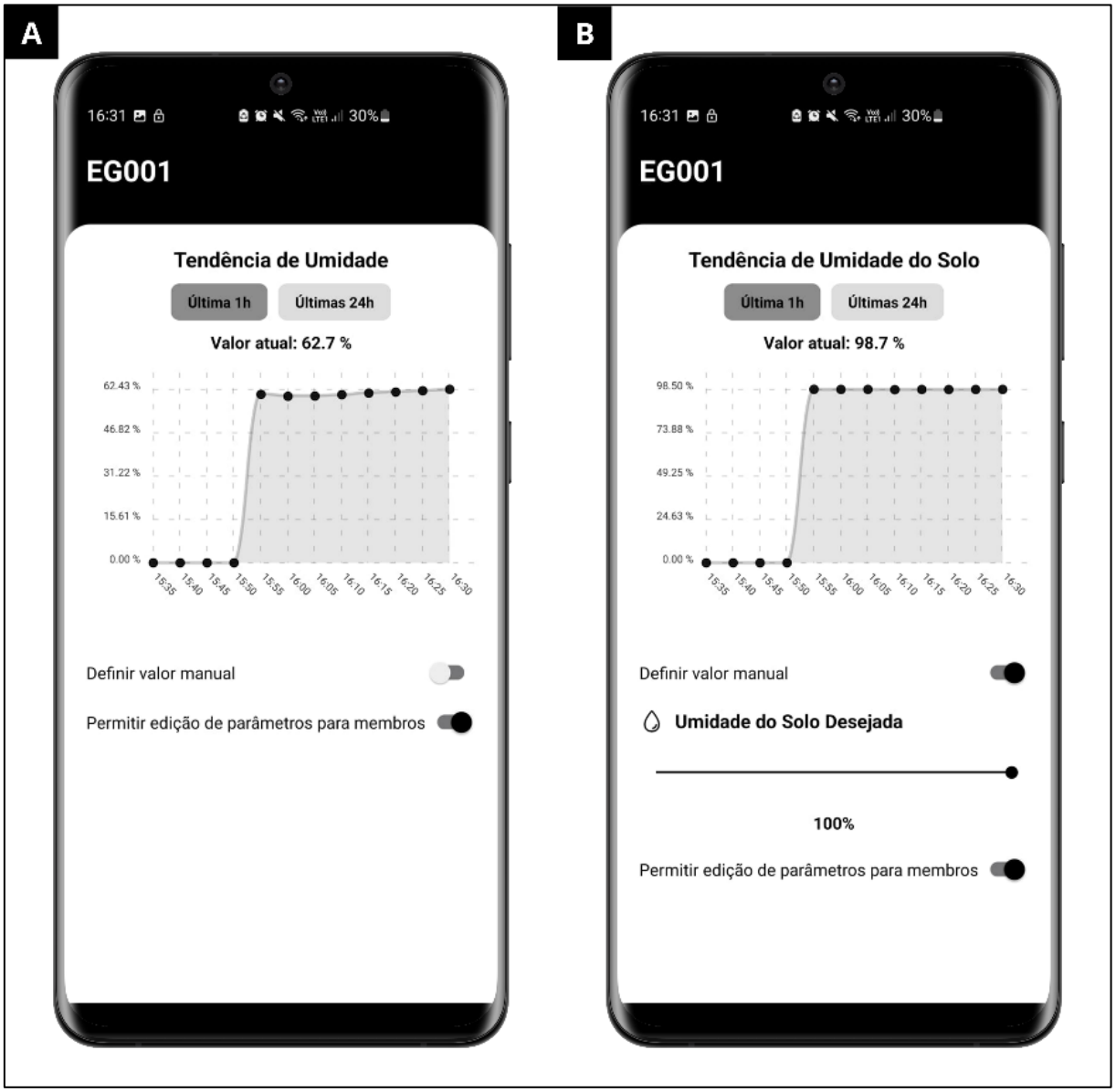
Imagem 43 — A) Temperatura - última 1h; B) Temperatura - últimas 24h



Fonte: elaborado pelo próprio autor.

O aplicativo apresenta gráficos de tendência para umidade do ar e umidade do solo, permitindo acompanhamento individual dessas variáveis ao longo do tempo. A Imagem 44 apresenta os gráficos dessas variáveis.

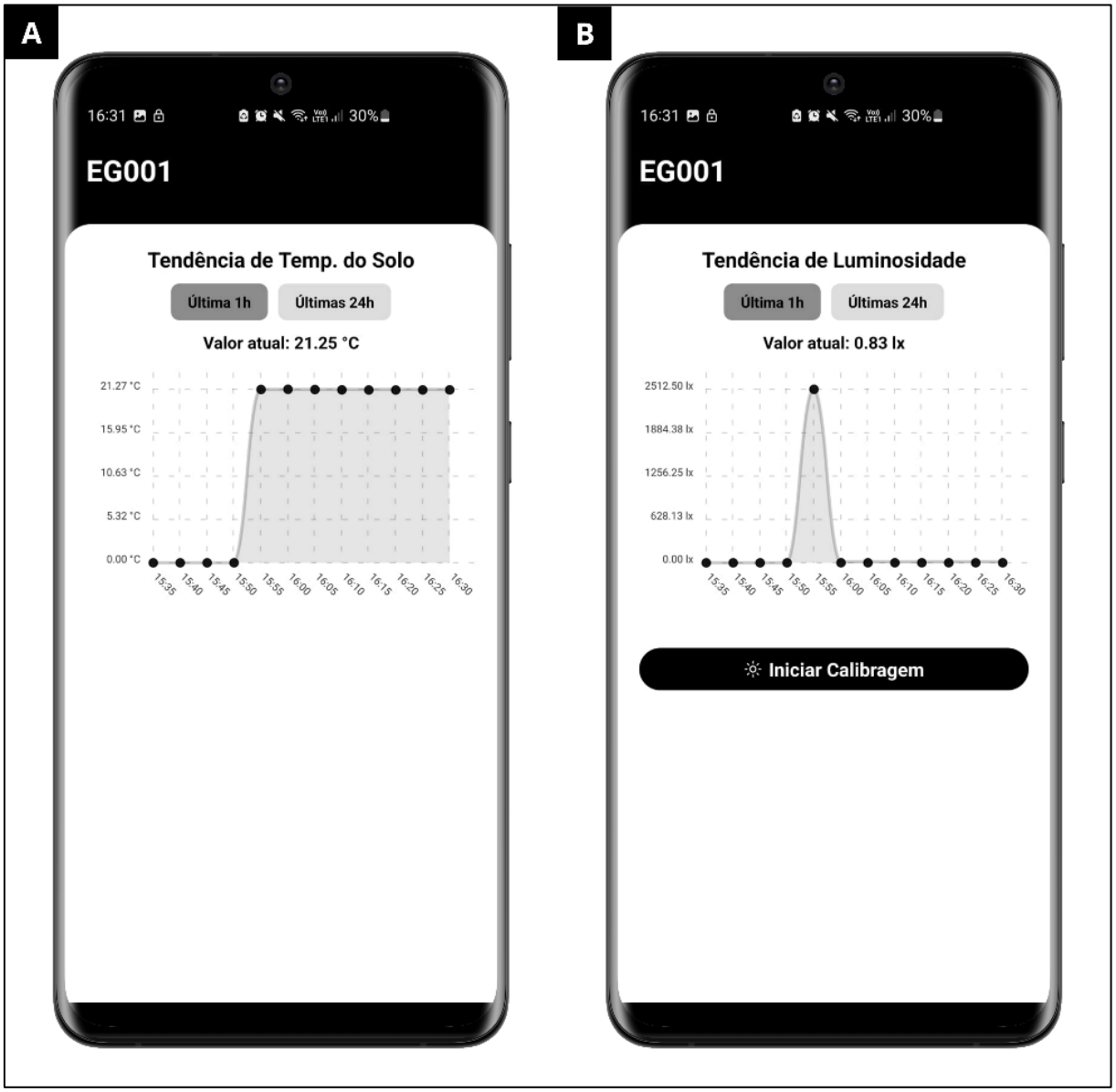
Imagem 44 — A) Tela umidade do ar; B) Tela umidade do solo



Fonte: elaborado pelo próprio autor.

O monitoramento de temperatura do solo e luminosidade é apresentado através de gráficos específicos. A interface de luminosidade inclui funcionalidade de calibração do sensor, visto que a luminária utilizada possui sistema de dimerização, permitindo ajustes para adequação aos níveis de luminosidade definidos pelo banco de dados. A Imagem 45 apresenta os gráficos dessas variáveis.

Imagem 45 — A) Tela temperatura do solo; B) Tela luminosidade



Fonte: elaborado pelo próprio autor.

A visualização gráfica proporciona compreensão clara do comportamento das variáveis ao longo do tempo, facilitando identificação de padrões e anomalias. O sistema de permissões permite que administradores controlem a capacidade de membros editarem valores desejados, garantindo gestão adequada das configurações da estufa.

4.4.6 Telas auxiliares

O aplicativo disponibiliza telas auxiliares para gerenciamento de estufas e configurações de conta. A tela de cadastro permite vincular novos dispositivos através do código de identificação da estufa, sendo que após o cadastro estes aparecem automaticamente na listagem da tela minhas estufas. A tela de configurações disponibiliza funcionalidade de *logout* para encerramento seguro da sessão. A Imagem 46 apresenta as telas auxiliares do aplicativo.

Imagem 46 — A) Tela cadastrar estufa; B) Tela configurações

Fonte: elaborado pelo próprio autor.

4.5 ESTRUTURAÇÃO DO FIREBASE

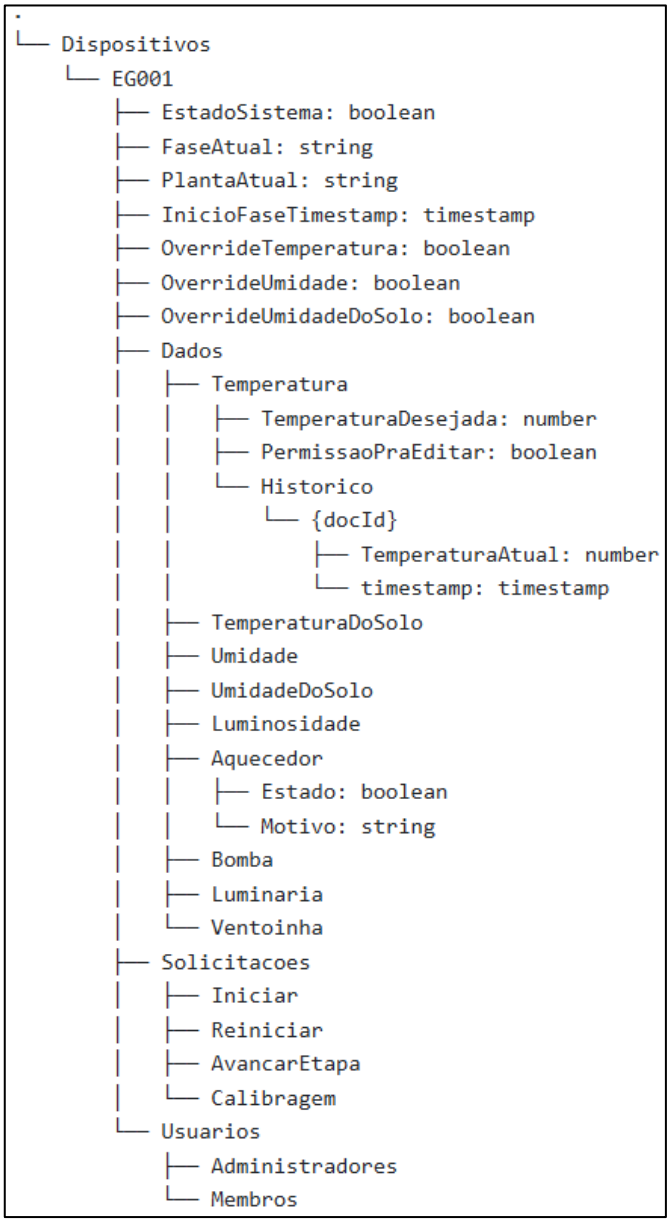
O Firebase foi estruturado para gerenciar a comunicação e armazenamento de dados do protótipo através de dois serviços: Firestore Database e Realtime Database. As subseções seguintes apresentam a estruturação de cada serviço e suas respectivas hierarquias de dados.

4.5.1 Firestore Database

O Firestore Database armazena os dados estruturados do sistema em coleções e documentos organizados hierarquicamente. A estrutura implementada possui três coleções principais: dispositivos, *presets* e usuários.

A coleção Dispositivos concentra as informações das estufas cadastradas, onde cada estufa é identificada por um código único e contém campos que descrevem seu estado atual de operação. As informações são organizadas em três subcoleções: dados, que agrupa sensores e atuadores, solicitações, que gerencia comandos enviados pelo aplicativo, e usuários, que controla as permissões de acesso. A Imagem 47 apresenta a estrutura completa desta coleção.

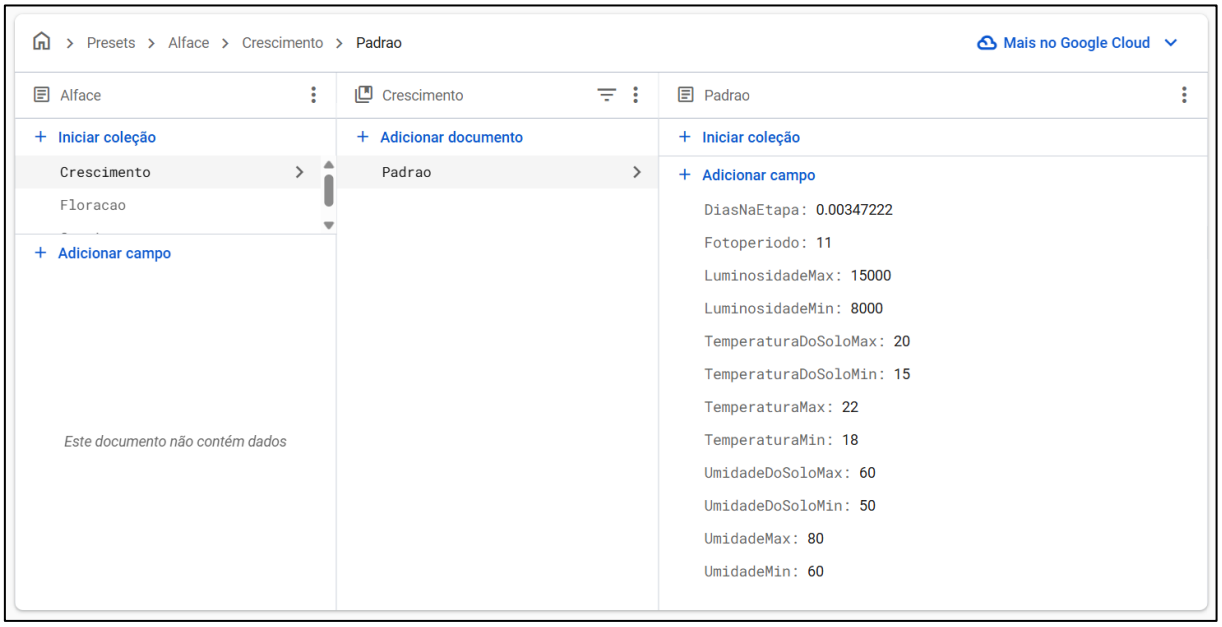
Imagem 47 — Estrutura da coleção dispositivos



Fonte: elaborado pelo próprio autor

A coleção *presets* armazena as configurações recomendadas para o cultivo, cada documento representa uma planta e possui subcoleções que representam as fases: crescimento, floração e germinação. Dentro de cada fase, o documento padrão define os parâmetros ideais como limites de temperatura, umidade, luminosidade e duração da fase. O sistema utiliza essas configurações para ajustar automaticamente os parâmetros de controle conforme a planta e fase selecionadas pelo usuário. A Imagem 48 ilustra esta organização.

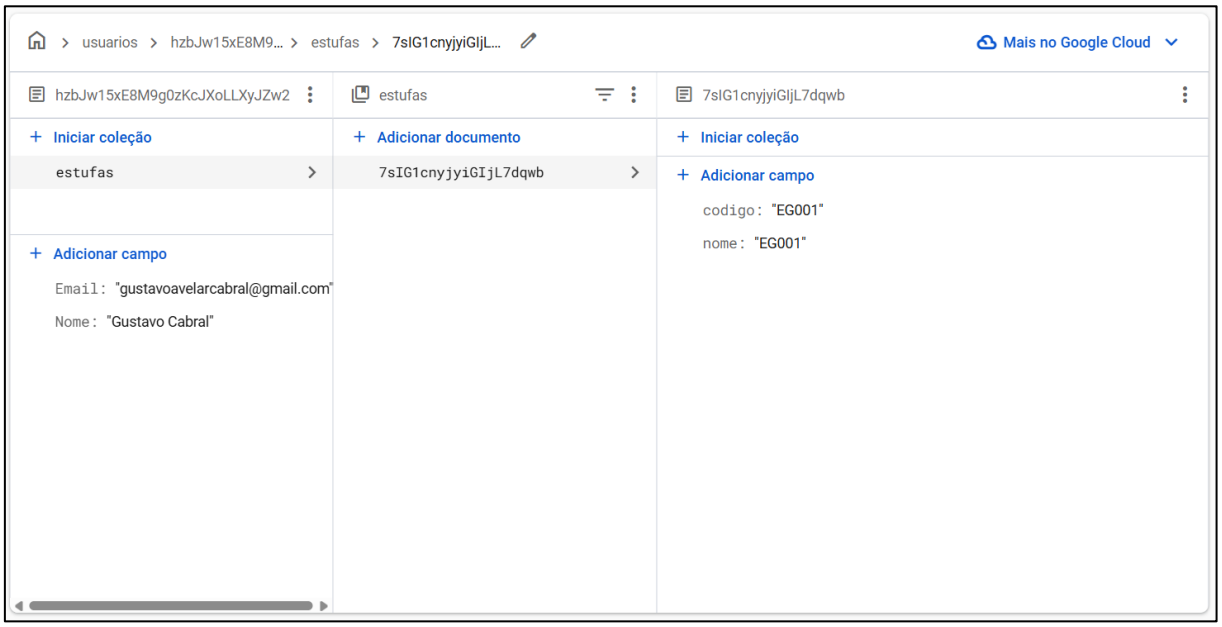
Imagem 48 — Estrutura da coleção *presets*



Fonte: elaborado pelo próprio autor.

A coleção usuários mantém o cadastro dos usuários do sistema. Cada usuário é identificado pelo Firebase Authentication e possui uma subcoleção que lista as estufas às quais tem acesso. Esta lista contém o nome e código de cada estufa. O código funciona como referência para localizar as informações correspondentes na coleção dispositivos. A Imagem 49 apresenta esta estrutura.

Imagem 49 — Estrutura da coleção usuários



Fonte: elaborado pelo próprio autor.

4.5.2 Realtime Database

O Realtime Database armazena os valores atuais dos sensores e mantém sincronização em tempo real com o aplicativo móvel. A estrutura organiza os dados em formato de árvore onde cada estufa é identificada por seu código. Dentro de cada estufa existe o campo dados atuais que contém os valores de luminosidade, temperatura do ar, temperatura do solo, umidade do ar, umidade do solo e o *timestamp* da última atualização. A Imagem 50 apresenta esta organização.

Imagem 50 — Estrutura do Realtime Database

Fonte: elaborado pelo próprio autor

5 ANÁLISE E DISCUSSÃO DOS RESULTADOS

Este capítulo apresenta a análise dos resultados obtidos durante o desenvolvimento e validação do protótipo. A avaliação foi estruturada em testes individuais de cada módulo e em um ciclo experimental de 72 horas que simula diferentes fases de cultivo. Adicionalmente, é apresentada uma análise dos custos envolvidos no desenvolvimento do protótipo.

5.1 TESTES DOS SISTEMAS INDIVIDUAIS

Nesta etapa foram conduzidos testes isolados para validar as principais funcionalidades do protótipo. Foram avaliados os sistemas de fotoperíodo, regulação térmica, umidade do ar e irrigação, permitindo identificar o desempenho individual de cada componente.

5.1.1 Sistema de fotoperíodo

O teste do sistema de fotoperíodo teve como objetivo validar a capacidade do protótipo em manter a iluminância estável dentro de faixas pré-estabelecidas. O ensaio foi estruturado em três etapas de 20 minutos, simulando as fases de germinação, crescimento e floração. A iluminância foi ajustada manualmente por meio de um *dimmer* no início de cada fase.

Os resultados estatísticos apresentados na Tabela 1 demonstram boa estabilidade do sistema nas três fases avaliadas. A fase de germinação obteve coeficiente de variação de 0,84%, demonstrando a capacidade do sistema em manter a luminosidade estável dentro da faixa estabelecida. As fases de crescimento e floração apresentaram coeficientes de variação de 2,23% e 2,03%, respectivamente, mantendo a estabilidade mesmo em níveis mais elevados de luminosidade.

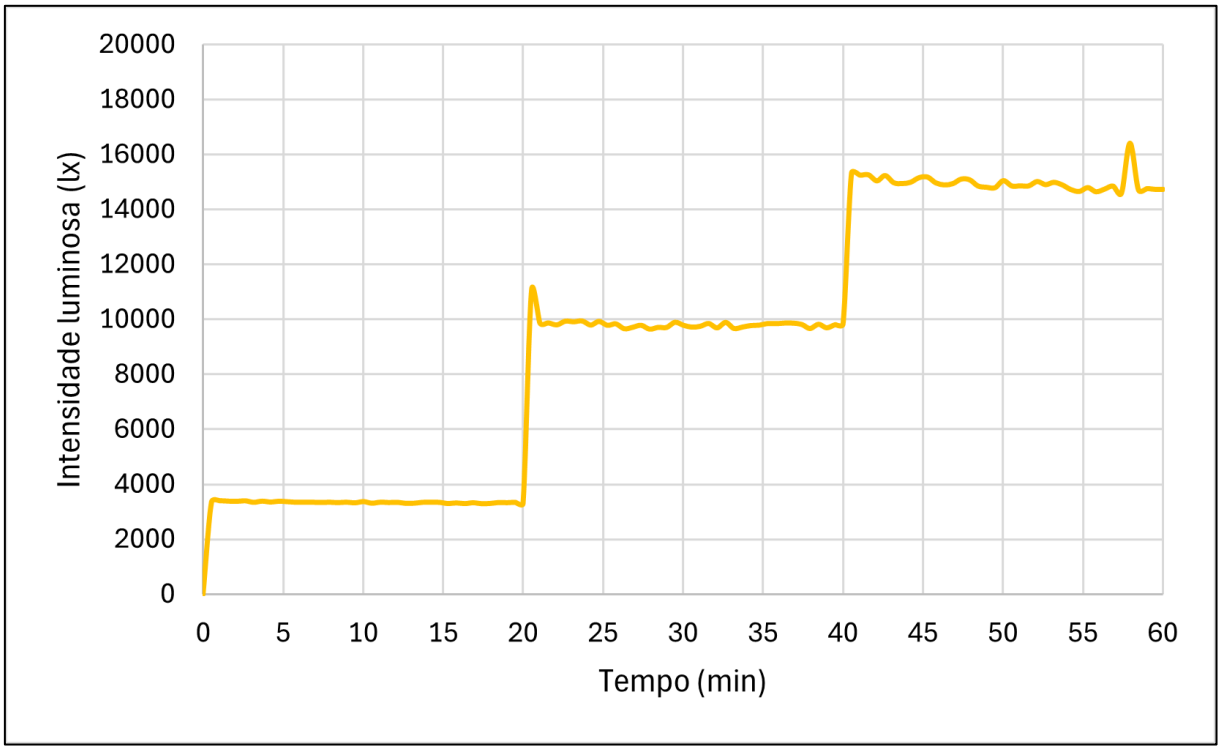
Tabela 1 — Análise estatística do sistema de fotoperíodo

Etapa	Média (lx)	Mediana (lx)	Desvio Padrão (lx)	CV (%)
Germinação	3.345,45	3.347,29	27,95	0,84
Crescimento	9.825,87	9.800,42	218,75	2,23
Floração	14.951,96	14.888,33	303,80	2,03

Fonte: elaborado pelo próprio autor.

A Imagem 51 ilustra o comportamento do sistema ao longo do teste. As transições entre as fases ocorreram de forma abrupta, demonstrando a eficiência do sensor BH1750 no monitoramento da iluminância. Observam-se picos isolados no início da fase de crescimento e ao final do ensaio, atribuídos às transições entre etapas e a possíveis ruídos de leitura.

Imagem 51 — Comportamento do sistema de fotoperíodo durante teste de três fases



Fonte: elaborado pelo próprio autor.

5.1.2 Sistema de regulação térmica

O sistema de regulação térmica foi avaliado por meio de três ensaios distintos. Os dois primeiros testes tiveram duração de 120 minutos e consistiram na avaliação do aquecimento com e sem ventilação forçada. O terceiro ensaio, com duração de 60 minutos, teve como objetivo analisar a capacidade de resfriamento do protótipo.

5.1.2.1 Aquecimento com ventoinha

O primeiro teste avaliou o sistema com o resistor de aquecimento e as ventoinhas ativos. O ensaio foi dividido em duas etapas: estabilização inicial (0-30 min) e aquecimento contínuo (30-120 min). A Tabela 2 apresenta os resultados estatísticos obtidos.

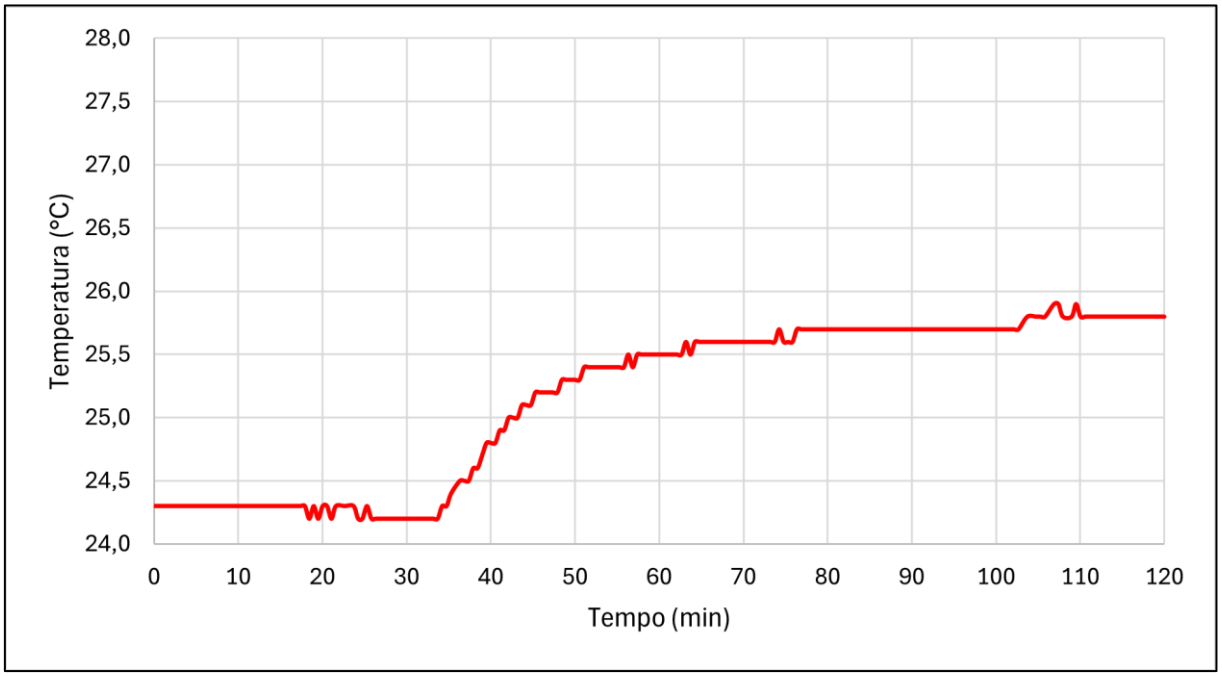
Tabela 2 — Análise estatística do sistema de aquecimento com ventoinha

Etapa	Média (°C)	Mediana (°C)	Desvio Padrão (°C)	CV (%)
Estabilização	24,27	24,30	0,04	0,18
Aquecimento	25,46	25,60	0,43	1,69

Fonte: elaborado pelo próprio autor.

Na etapa de estabilização, a temperatura média foi de 24,27 °C com coeficiente de variação de 0,18%. Na etapa de aquecimento contínuo, a média elevou-se para 25,46 °C com coeficiente de variação de 1,69%. A temperatura mínima registrada foi de 24,20 °C e a máxima de 25,90 °C, resultando em amplitude térmica de 1,70 °C. A Imagem 52 apresenta a variação da temperatura ao longo do tempo.

Imagem 52 — Curva de aquecimento com ventilação forçada



Fonte: elaborado pelo próprio autor.

5.1.2.2 Aquecimento sem ventoinha

O segundo teste avaliou o comportamento térmico do sistema apenas com o resistor de aquecimento ativo, sem ventilação forçada. O ensaio foi dividido em duas etapas: estabilização inicial (0-30 min) e aquecimento contínuo (30-120 min). A Tabela 3 apresenta os resultados estatísticos obtidos.

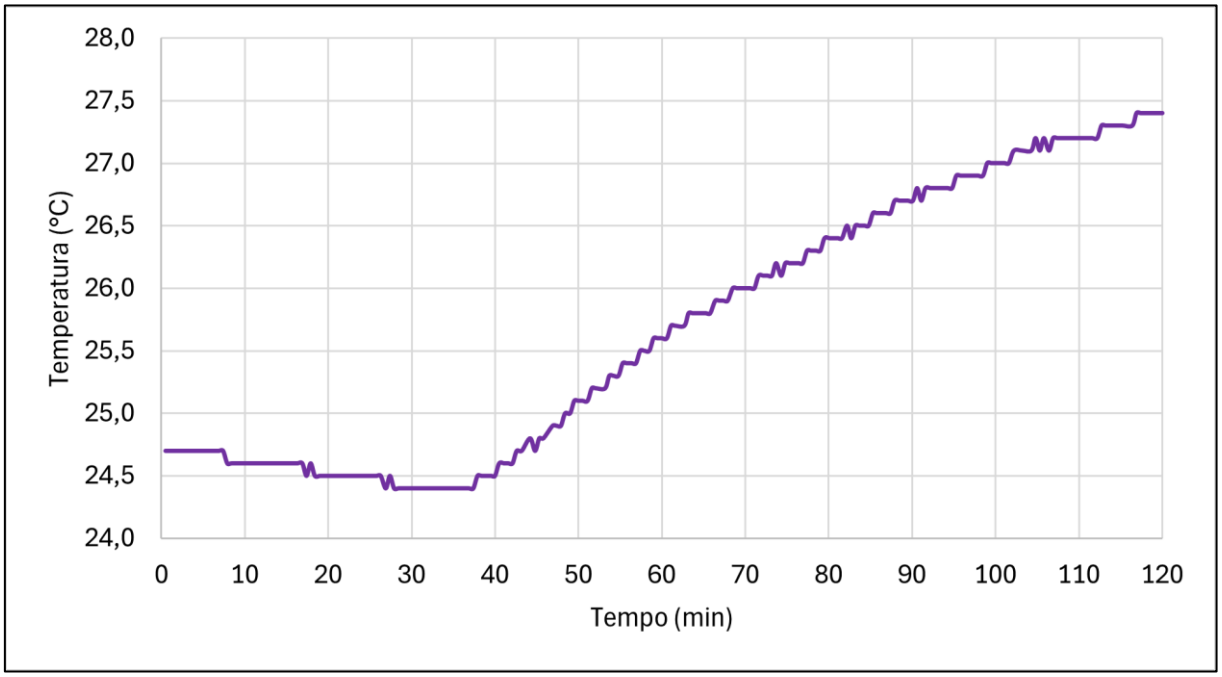
Tabela 3 — Análise estatística do sistema de aquecimento sem ventoinha

Etapa	Média (°C)	Mediana (°C)	Desvio Padrão (°C)	CV (%)
Estabilização	24,57	24,60	0,10	0,40
Aquecimento	26,06	26,20	0,97	3,72

Fonte: elaborado pelo próprio autor.

Na etapa de estabilização, a temperatura média foi de 24,57 °C com coeficiente de variação de 0,40%. Na etapa de aquecimento contínuo, a média elevou-se para 26,06 °C com coeficiente de variação de 3,72%. A temperatura mínima registrada foi de 24,40 °C e a máxima de 27,40 °C, resultando em amplitude térmica de 3,00 °C. A Imagem 53 apresenta a variação da temperatura ao longo do tempo.

Imagem 53 — Curva de aquecimento sem ventilação forçada

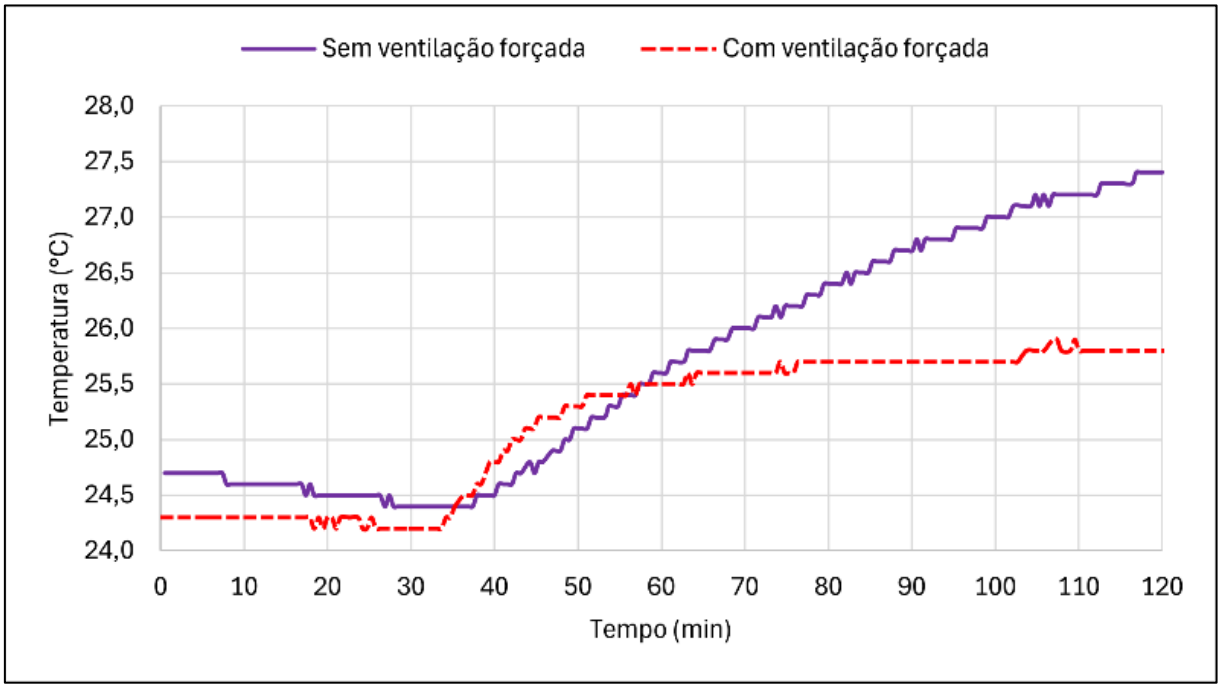


Fonte: elaborado pelo próprio autor.

5.1.2.3 Análise comparativa

A Imagem 54 apresenta a sobreposição das curvas de aquecimento dos dois testes. A curva sem ventilação demonstra aquecimento contínuo e progressivo, enquanto a curva com ventilação apresenta tendência à estabilização térmica em temperatura inferior.

Imagem 54 — Comparação entre aquecimento com e sem ventilação forçada



Fonte: elaborado pelo próprio autor.

Na etapa de aquecimento contínuo, a ventilação forçada promoveu redução de 54,57% no coeficiente de variação (de 3,72% para 1,69%). A temperatura máxima foi de 25,90 °C com ventilação, contra 27,40 °C sem ventilação, representando redução de 1,50 °C. Os resultados evidenciam que a circulação de ar promove aquecimento mais uniforme e controlado do ambiente.

5.1.2.4 Resfriamento

O terceiro teste foi conduzido em duas etapas de 30 minutos. Na primeira etapa, o resistor de aquecimento permaneceu ativo sem ventilação. Na segunda etapa, o aquecedor foi desligado e as ventoinhas foram acionadas para promover o resfriamento. A Tabela 4 apresenta os resultados estatísticos obtidos

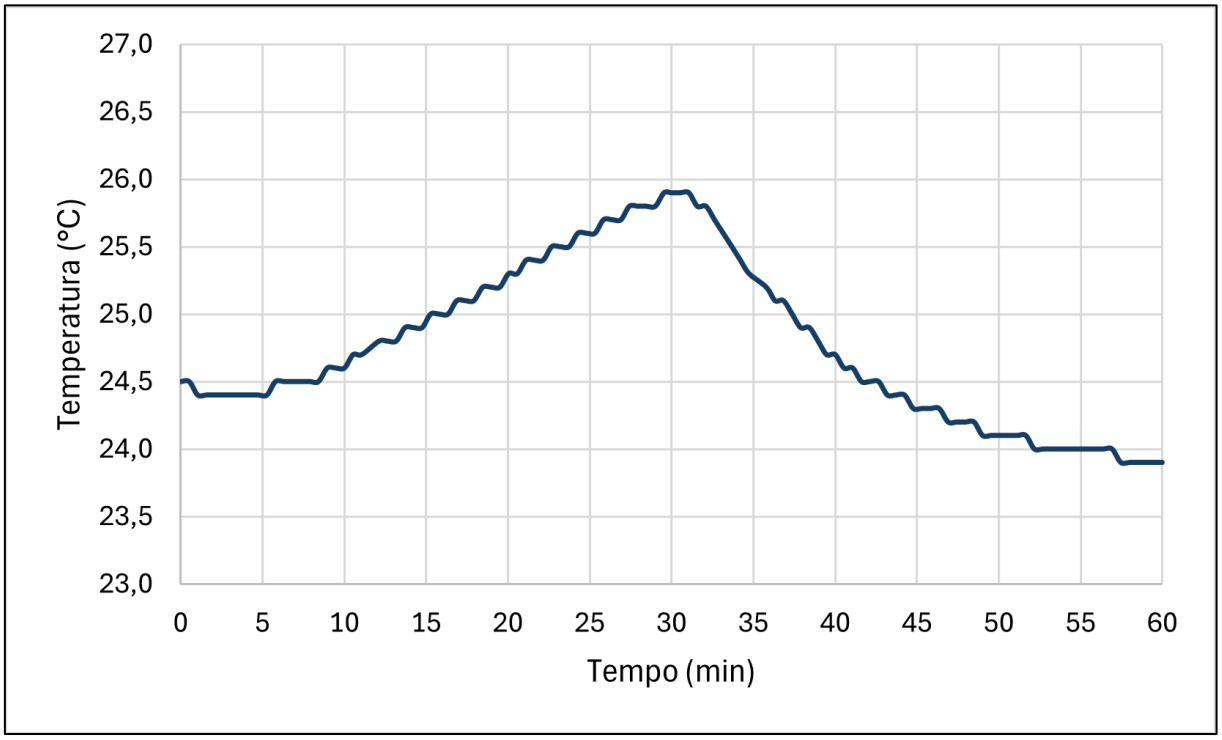
Tabela 4 — Análise estatística do sistema de resfriamento

Etapa	Média (°C)	Mediana (°C)	Desvio Padrão (°C)	CV (%)
Aquecimento	25,02	25,00	0,51	2,02
Resfriamento	24,52	24,30	0,61	2,49

Fonte: elaborado pelo próprio autor.

Na etapa de aquecimento, a temperatura média foi de 25,02 °C com coeficiente de variação de 2,02%. Na etapa de resfriamento, a média foi de 24,52 °C com coeficiente de variação de 2,49%. A temperatura mínima registrada foi de 23,90 °C e a máxima de 25,90 °C. A Imagem 55 apresenta a variação da temperatura ao longo do teste.

Imagem 55 — Curva de aquecimento e resfriamento ativo



Fonte: elaborado pelo próprio autor.

É notória a transição entre as etapas aos 30 minutos, com inversão da tendência térmica. A temperatura elevou-se gradualmente até atingir o pico de 25,90 °C, seguida de decaimento durante a etapa de resfriamento até atingir 23,90 °C.

5.1.3 Sistema de umidade do ar

O sistema de controle de umidade do ar foi avaliado por meio de dois ensaios distintos. No primeiro, foi avaliado o efeito da ventilação forçada sobre a umidade relativa do ar. No segundo, analisou-se o comportamento da umidade sob ação combinada de aquecimento e ventilação

5.1.3.1 Teste com ventilação forçada

O ensaio teve duração de 120 minutos, dividido em duas etapas de estabilização inicial (0-30 min) e ventilação contínua (30-120 min). A Tabela 5 apresenta os resultados estatísticos obtidos.

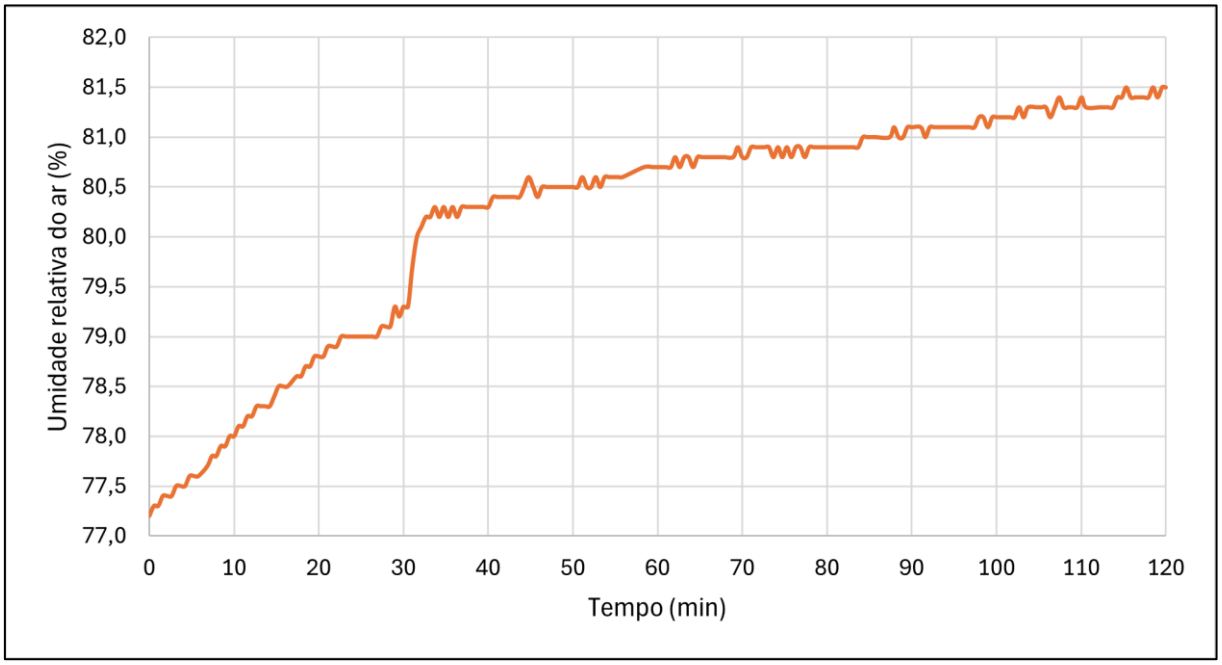
Tabela 5 — Análise estatística da umidade com ventilação forçada

Etapa	Média (%)	Mediana (%)	Desvio Padrão (%)	CV (%)
Estabilização	78,34	78,40	0,63	0,81
Ventilação	80,84	80,90	0,39	0,49

Fonte: elaborado pelo próprio autor.

Na etapa de estabilização, a umidade média foi de 78,34% com coeficiente de variação de 0,81%. Na etapa de ventilação contínua, a média elevou-se para 80,84% com coeficiente de variação de 0,49%. Esse comportamento, contrário ao esperado pela lógica de controle implementada, indica que a ventilação isolada não foi suficiente para reduzir a umidade do ambiente nas condições testadas. A Imagem 56 apresenta a variação da umidade ao longo do tempo.

Imagem 56 — Comportamento da umidade relativa com ventilação forçada



Fonte: elaborado pelo próprio autor.

5.1.3.2 Teste com aquecimento e ventilação

O ensaio teve duração de 120 minutos, dividido em três etapas de estabilização (0-30 min), aquecimento (30-75 min) e resfriamento (75-120 min). A Tabela 6 apresenta os resultados estatísticos de temperatura.

Tabela 6 — Análise estatística da temperatura no teste integrado

Etapa	Média (°C)	Mediana (°C)	Desvio Padrão (°C)	CV (%)
Estabilização	22,68	22,70	0,04	0,16
Aquecimento	23,78	23,80	0,82	3,45
Resfriamento	23,40	23,15	0,61	2,60

Fonte: elaborado pelo próprio autor.

Durante a etapa de aquecimento, a temperatura média elevou-se de 22,68 °C para 23,78 °C. Na etapa de resfriamento, a temperatura média foi de 23,40 °C. A Tabela 7 apresenta os resultados estatísticos de umidade.

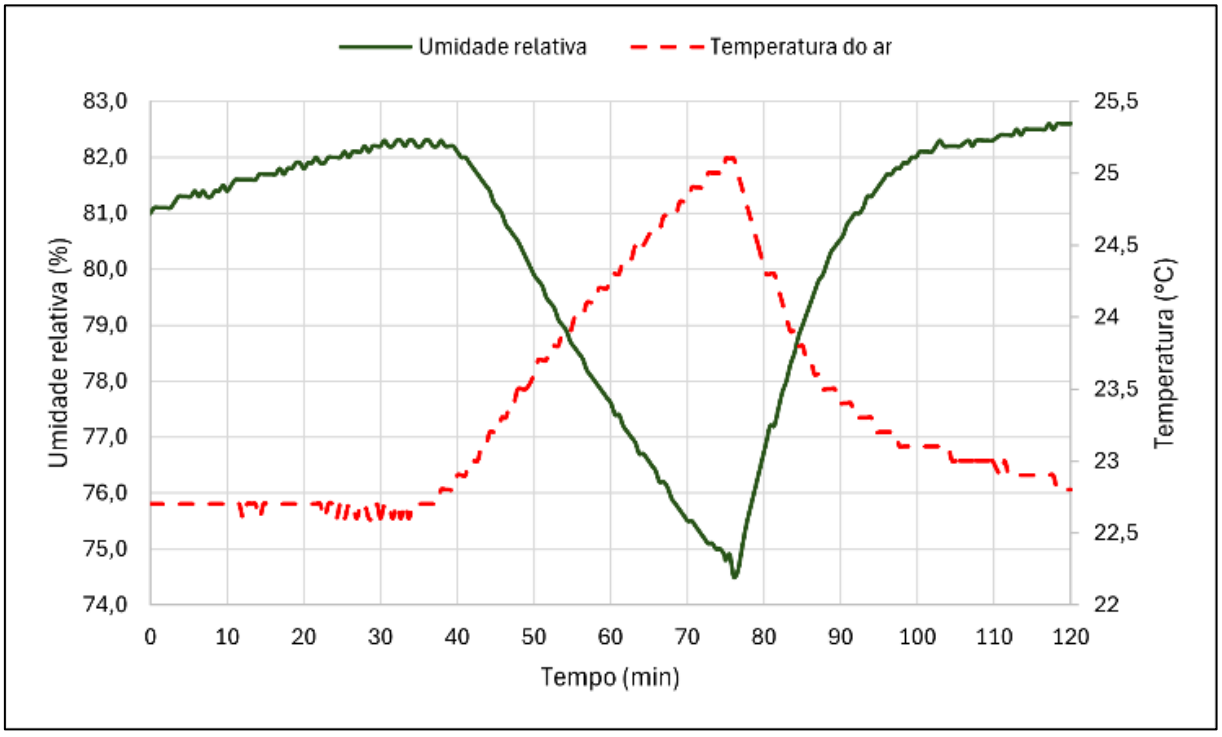
Tabela 7 — Análise estatística da umidade no teste integrado

Etapa	Média (%)	Mediana (%)	Desvio Padrão (%)	CV (%)
Estabilização	81,66	81,70	0,35	0,42
Aquecimento	79,09	79,10	2,63	3,33
Resfriamento	80,65	81,80	2,33	2,89

Fonte: elaborado pelo próprio autor.

A umidade relativa reduziu-se de 81,66% para 79,09% durante o aquecimento e recuperou-se para 80,65% durante o resfriamento. A Imagem 57 apresenta o comportamento simultâneo de temperatura e umidade ao longo do teste, confirmando o comportamento característico discutido na Subseção 2.3.1.

Imagem 57 — Comportamento integrado de temperatura e umidade relativa



Fonte: elaborado pelo próprio autor.

5.1.4 Sistema de irrigação

O sistema de irrigação foi avaliado por meio de um teste com duração de 90 minutos, dividido em três etapas de 30 minutos. A primeira etapa consistiu no monitoramento do solo em condição seca, a segunda realizou irrigações para elevar

a umidade à faixa de 70-80%, e a terceira executou novas irrigações visando atingir 90-100%. A Tabela 8 apresenta os resultados estatísticos obtidos.

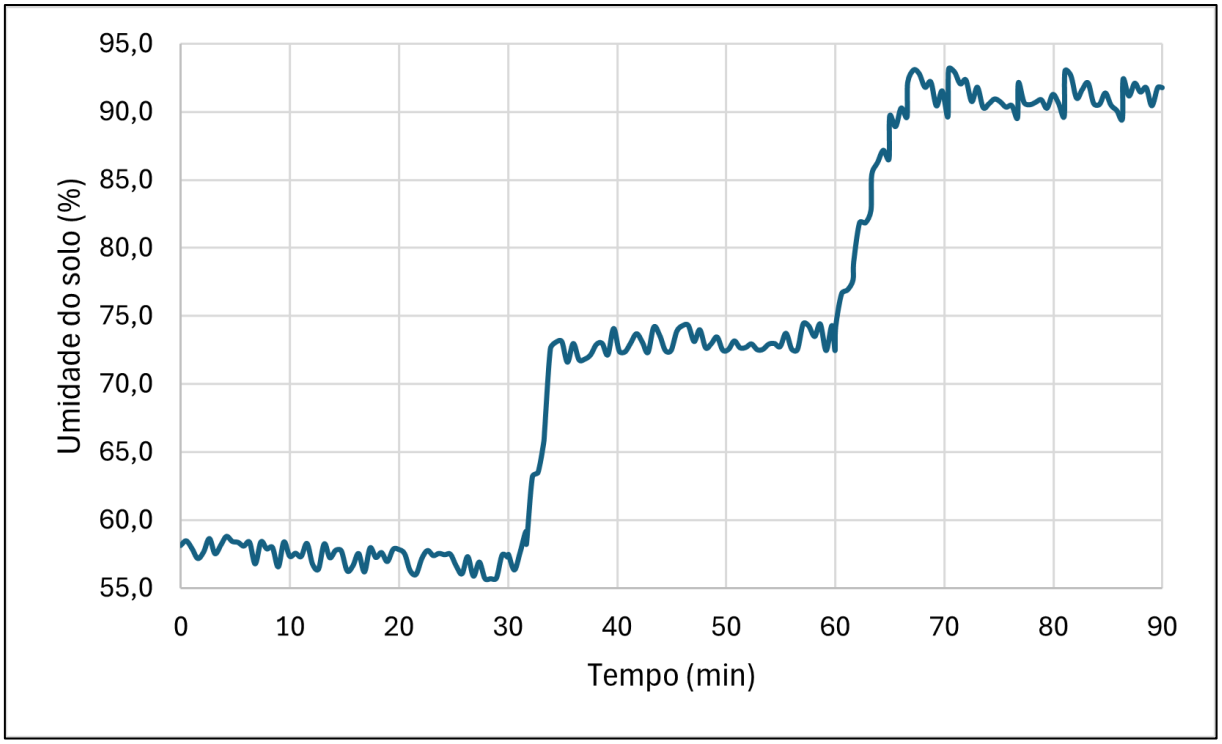
Tabela 8 — Análise estatística do sistema de irrigação

Etapa	Média (%)	Mediana (%)	Desvio Padrão (%)	CV (%)
Primeira	57,36	57,51	0,82	1,43
Segunda	70,93	72,66	4,99	7,04
Terceira	89,08	90,62	4,80	5,38

Fonte: elaborado pelo próprio autor.

Na primeira etapa, a umidade média foi de 57,36% com coeficiente de variação de 1,43%. Durante a segunda etapa, a umidade elevou-se para 70,93%. Na terceira etapa, a umidade atingiu 89,08%. Os resultados demonstram a eficácia do sistema de irrigação em atingir as faixas de umidade desejadas. A Imagem 58 apresenta a variação da umidade do solo ao longo do teste.

Imagem 58 — Comportamento da umidade do solo durante irrigações consecutivas



Fonte: elaborado pelo próprio autor.

5.2 TESTE INTEGRADO

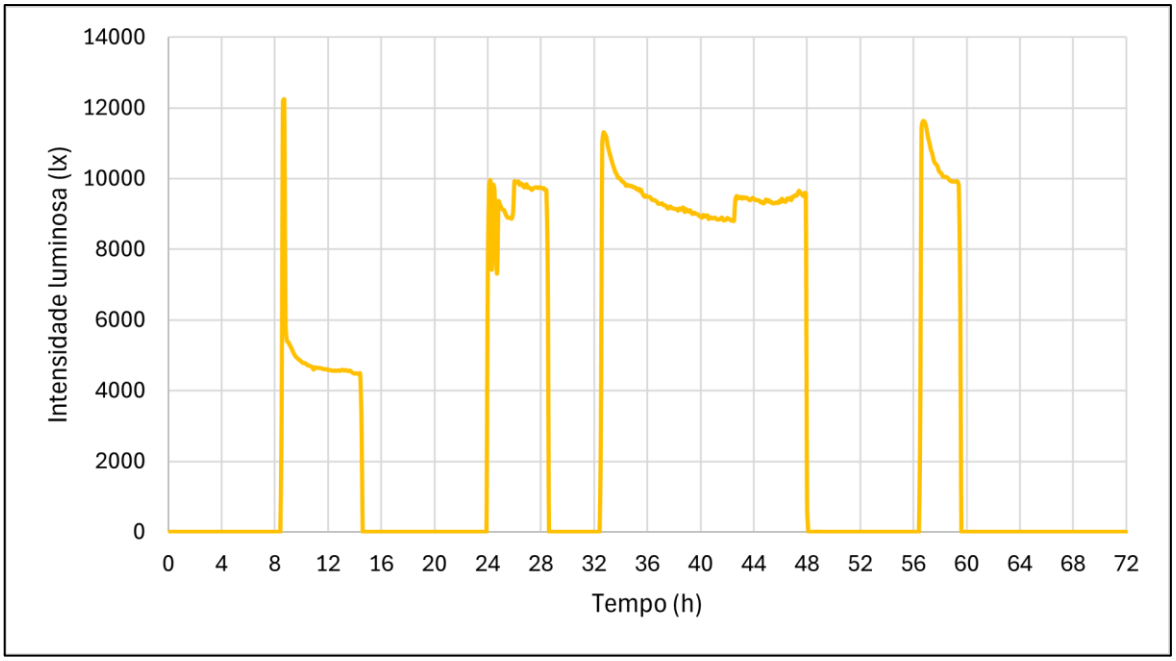
Nesta etapa foram conduzidos testes integrados com duração de 72 horas para validar as principais funcionalidades do protótipo. Foram avaliados os sistemas de fotoperíodo, regulação térmica, umidade do ar, irrigação e temperatura do solo, permitindo identificar o desempenho integrado de cada sistema.

5.2.1 Sistema de fotoperíodo

O sistema de fotoperíodo foi submetido a um teste de validação com duração de 72 horas, dividido em três etapas de 24 horas. O objetivo principal foi avaliar o funcionamento do sistema em diferentes cenários e verificar a estabilidade ao longo do tempo.

Na primeira etapa (0-24h), foi considerado um fotoperíodo de 6 horas com faixa de iluminância entre 4.000 e 6.000 lux. Durante a segunda etapa (24-48h), o fotoperíodo foi alterado para 20 horas com iluminância entre 9.000 e 11.000 lux. Para a terceira etapa (48-72h), o fotoperíodo foi reduzido para 3 horas mantendo a mesma faixa de iluminância da etapa anterior. A Imagem 59 ilustra o comportamento do sistema ao longo do teste.

Imagem 59 — Comportamento do sistema de fotoperíodo durante teste de 72 horas



Fonte: elaborado pelo próprio autor.

A Tabela 9 apresenta a análise estatística dos dados. Os coeficientes de variação elevados indicam valores discrepantes (*outliers*) que comprometem a representatividade dos resultados. Esses valores ocorreram principalmente durante transições de estado da luminária e em momentos de calibração da iluminância.

Tabela 9 — Análise estatística do sistema de fotoperíodo durante teste de 72 horas

Etapa	Média (lx)	Mediana (lx)	Desvio Padrão (lx)	CV (%)
Primeira	4.693,30	4.605,50	1.403,26	29,90
Segunda	9.325,36	9.385,15	933,27	10,01
Terceira	10.095,53	10.138,25	1.557,19	15,42

Fonte: elaborado pelo próprio autor.

Para obter resultados mais representativos, foi aplicado o método dos quartis para identificação e remoção de *outliers*. A Tabela 10 apresenta a análise estatística após o tratamento realizado.

Tabela 10 — Análise estatística do sistema de fotoperíodo após o tratamento de *outliers*

Etapa	Média (lx)	Mediana (lx)	Desvio Padrão (lx)	CV (%)
Primeira	4.641,10	4.595,30	137,31	2,96
Segunda	9.382,32	9.383,64	365,28	3,89
Terceira	10.442,98	10.170,17	602,09	5,77

Fonte: elaborado pelo próprio autor.

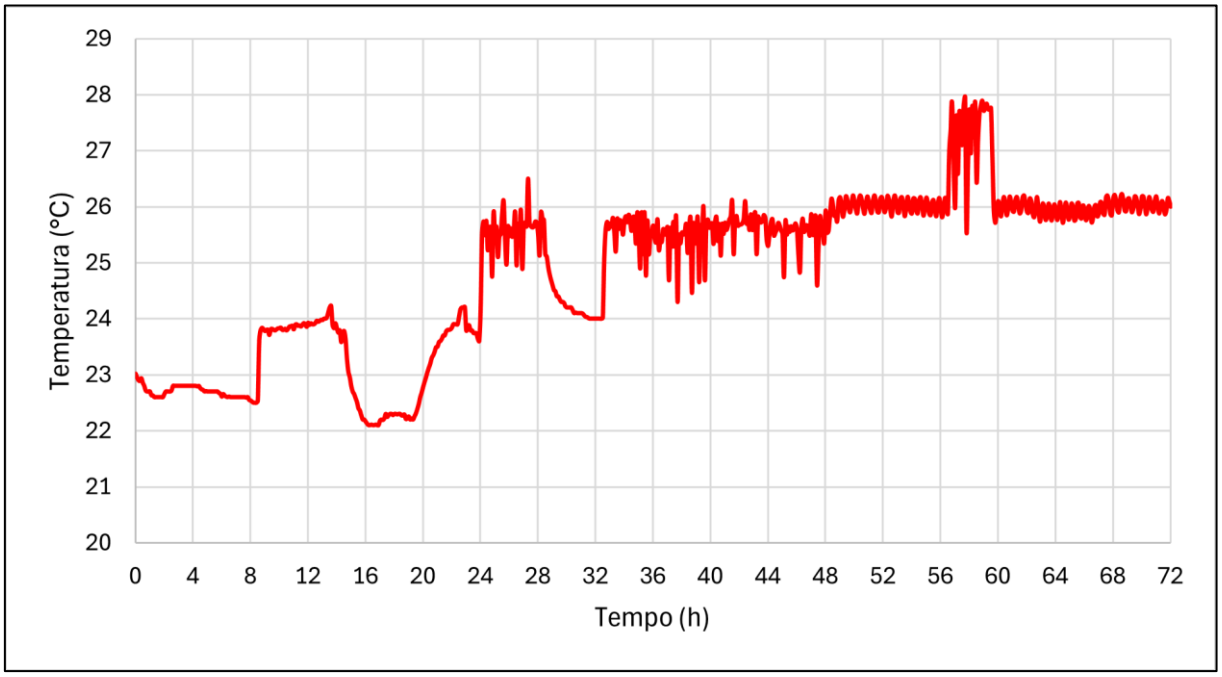
Após o tratamento, observa-se redução significativa dos coeficientes de variação para valores inferiores a 6%, demonstrando estabilidade do sistema em todos os cenários testados.

5.2.2 Sistema de regulação térmica

O sistema de regulação térmica foi submetido a um teste de validação com duração de 72 horas, dividido em três etapas de 24 horas. O objetivo principal foi avaliar a capacidade do sistema em manter diferentes faixas de temperatura e analisar a influência do fotoperíodo sobre o controle térmico.

Na primeira etapa (0-24h), a faixa de temperatura foi mantida entre 22 °C e 24 °C. Durante a segunda etapa (24-48h), a faixa foi elevada para 24 °C a 26 °C. Para a terceira etapa (48-72h), a faixa foi configurada para 26 °C a 28 °C. A Imagem 60 ilustra o comportamento do sistema ao longo do teste.

Imagem 60 — Comportamento do sistema de regulação térmica durante teste de 72 horas



Fonte: elaborado pelo próprio autor.

Os resultados estatísticos apresentados na Tabela 11 demonstram a capacidade do sistema em manter as faixas de temperatura estabelecidas, com coeficientes de variação inferiores a 3% em todas as etapas.

Tabela 11 — Análise estatística da temperatura durante o teste de 72 horas

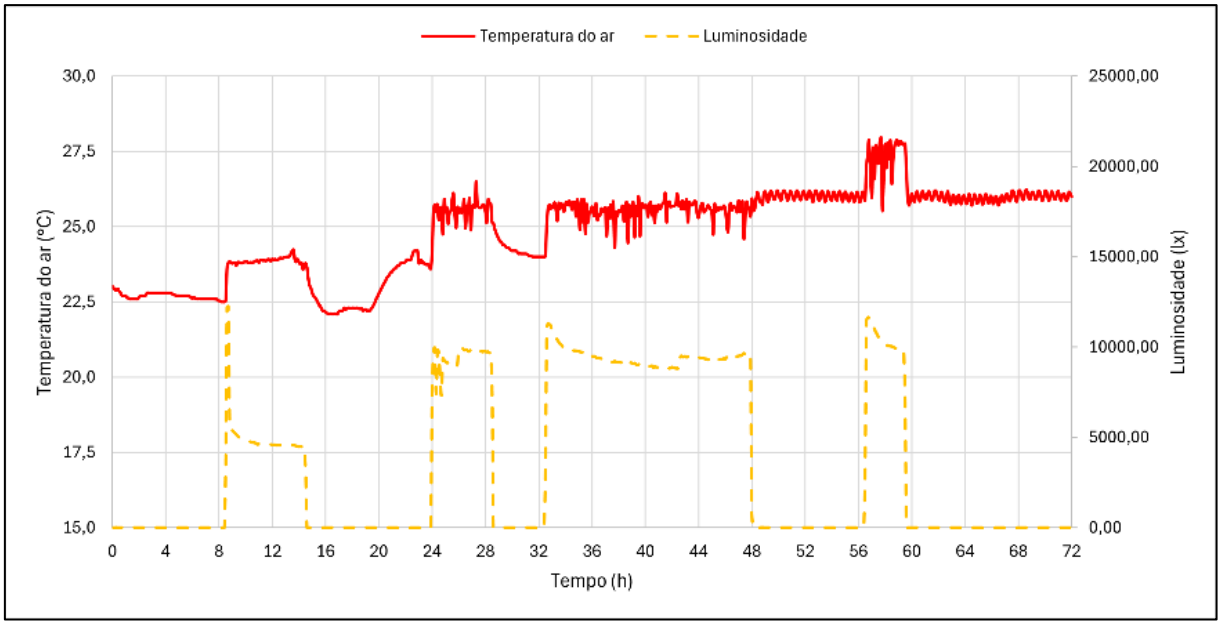
Etapa	Média (°C)	Mediana (°C)	Desvio Padrão (°C)	CV (%)
Primeira	23,08	22,80	0,66	2,85
Segunda	25,34	25,57	0,58	2,30
Terceira	26,16	26,03	0,52	1,97

Fonte: elaborado pelo próprio autor.

A Imagem 61 ilustra a interação entre os sistemas de fotoperíodo e regulação térmica. Durante os períodos de acionamento da luminária, observa-se elevação

térmica da estufa, resultado da dissipação de calor dos componentes eletrônicos da luminária.

Imagem 61 — Comportamento integrado do sistema de fotoperíodo e regulação térmica

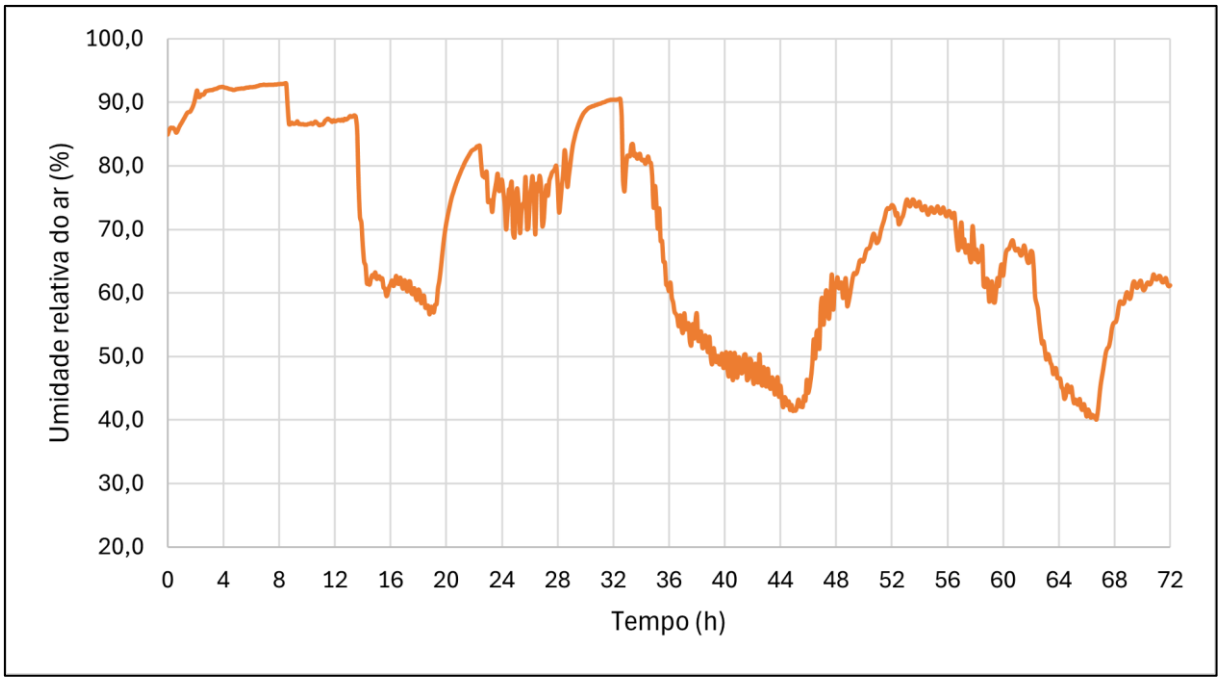


Fonte: elaborado pelo próprio autor.

5.2.3 Sistema de umidade do ar

O monitoramento da umidade relativa do ar foi realizado durante as 72 horas de teste, sem controle ativo. A escolha de realizar o teste sem controle considerou a limitação do sistema de ventilação em regular a umidade, conforme constatado na Subseção 5.1.3.1. A Imagem 62 ilustra o comportamento da umidade ao longo do teste.

Imagem 62 — Comportamento do sistema de umidade do ar durante teste de 72 horas



Fonte: elaborado pelo próprio autor.

Observa-se redução progressiva da umidade ao longo das etapas, diretamente relacionada ao aumento da temperatura nas diferentes fases do teste. Na primeira etapa (0-24h), a umidade média foi de 80,36%. Durante a segunda etapa (24-48h), a umidade reduziu para 64,93%. Para a terceira etapa (48-72h), a umidade manteve-se em 61,65%. A Tabela 12 apresenta os resultados estatísticos obtidos.

Tabela 12 — Análise estatística da umidade do ar durante teste de 72 horas

Etapa	Média (%)	Mediana (%)	Desvio Padrão (%)	CV (%)
Primeira	80,36	86,51	12,20	15,18
Segunda	64,93	61,33	16,30	25,11
Terceira	61,65	62,58	9,97	15,88

Fonte: elaborado pelo próprio autor.

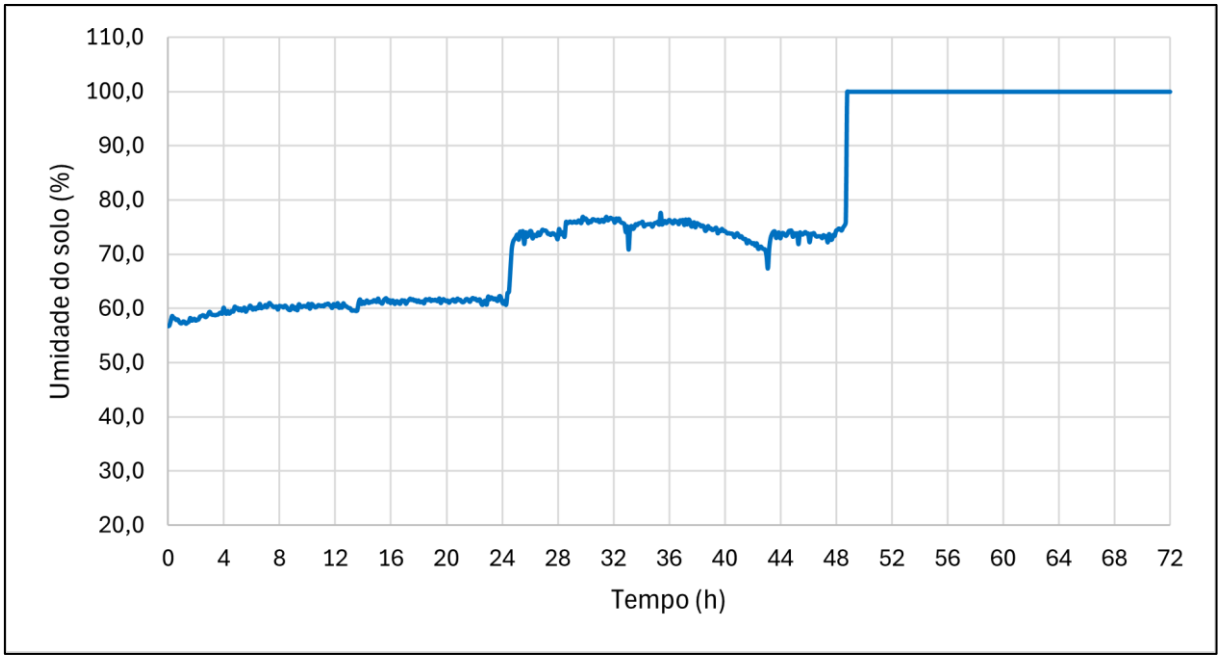
Os coeficientes de variação elevados refletem a ausência de controle ativo, limitando o sistema ao monitoramento passivo da umidade.

5.2.4 Sistema de irrigação

O sistema de irrigação foi submetido a um teste de validação com duração de 72 horas, dividido em três etapas de 24 horas. O objetivo principal foi avaliar a capacidade do sistema em elevar e manter diferentes níveis de umidade do solo em diferentes cenários.

Na primeira etapa (0-24h), a faixa de umidade foi mantida entre 50% e 60%. Durante a segunda etapa (24-48h), a faixa foi elevada para 70% a 80%. Para a terceira etapa (48-72h), a umidade foi definida entre 90% e 100%. A Imagem 63 ilustra o comportamento do sistema ao longo do teste.

Imagem 63 — Comportamento do sistema de irrigação durante teste de 72 horas



Fonte: elaborado pelo próprio autor.

A Tabela 13 apresenta a análise estatística dos dados. O coeficiente de variação elevado da terceira etapa indica valores discrepantes (*outliers*) que comprometem a representatividade dos resultados, ocorrendo principalmente durante períodos de transição

Tabela 13 — Análise estatística do sistema de irrigação durante teste de 72 horas

Etapa	Média (%)	Mediana (%)	Desvio Padrão (%)	CV (%)
Primeira	60,35	60,58	1,22	2,02
Segunda	74,13	74,24	2,39	3,22
Terceira	99,27	100,00	4,24	4,27

Fonte: elaborado pelo próprio autor.

Para obter resultados mais representativos, foi aplicado o método dos quartis para identificação e remoção de *outliers*. A Tabela 14 apresenta a análise estatística após o tratamento.

Tabela 14 — Análise estatística do sistema de irrigação durante teste de 72 horas (com tratamento)

Etapa	Média (%)	Mediana (%)	Desvio Padrão (%)	CV (%)
Primeira	60,46	60,64	1,09	1,80
Segunda	74,46	74,30	1,47	1,97
Terceira	100,00	100,00	0,00	0,00

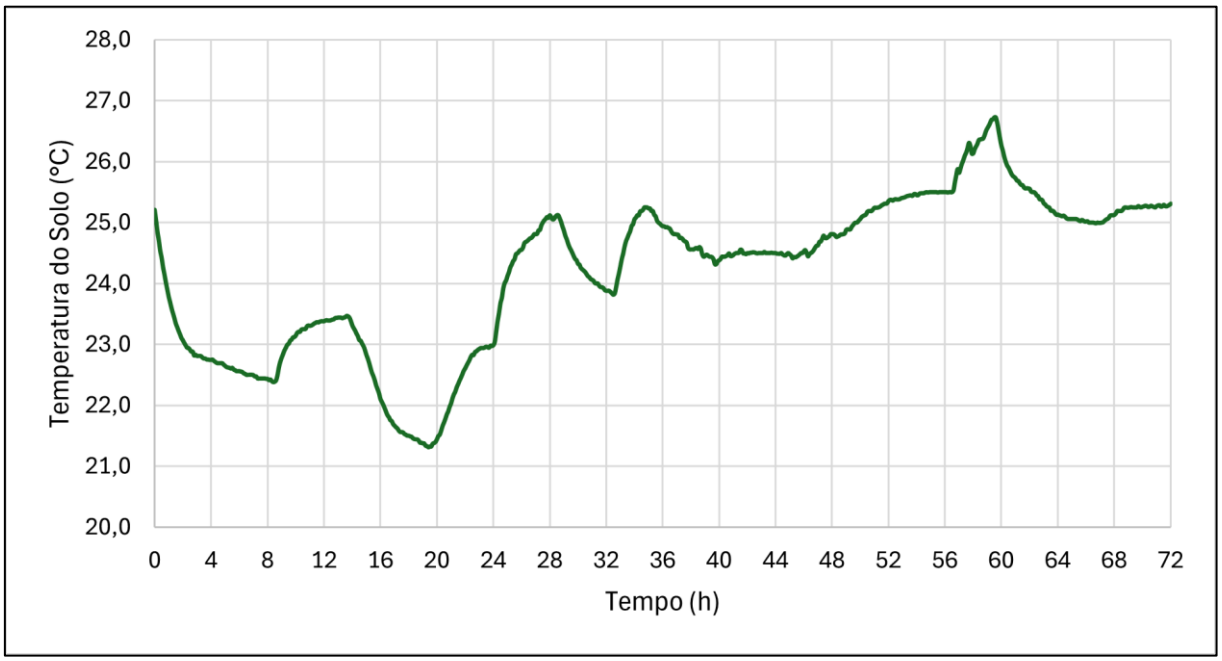
Fonte: elaborado pelo próprio autor.

Após o tratamento, observa-se redução significativa dos coeficientes de variação, demonstrando estabilidade do sistema em todos os cenários testados. A terceira etapa apresenta coeficiente de variação nulo, resultado da saturação completa do solo ao redor do sensor capacitivo.

5.2.5 Sistema de temperatura do solo

O monitoramento da temperatura do solo foi realizado durante as 72 horas de teste, sem controle ativo. Esta variável apresenta comportamento passivo, sendo influenciada pela temperatura ambiente. A Imagem 64 ilustra o comportamento ao longo do teste.

Imagem 64 — Comportamento do sistema de temperatura do solo durante teste de 72 horas



Fonte: elaborado pelo próprio autor.

Observa-se elevação gradual da temperatura do solo ao longo das etapas, seguindo a mesma tendência da temperatura ambiente. Na primeira etapa (0-24h), a temperatura média foi de 22,68 °C. Durante a segunda etapa (24-48h), a temperatura elevou-se para 24,55 °C. Para a terceira etapa (48-72h), a temperatura manteve-se em 25,42 °C. A Tabela 15 apresenta os resultados estatísticos obtidos.

Tabela 15 — Análise estatística da temperatura do solo durante teste de 72 horas

Etapa	Média (°C)	Mediana (°C)	Desvio Padrão (°C)	CV (%)
Primeira	22,68	22,74	0,74	3,24
Segunda	24,55	24,50	0,37	1,52
Terceira	25,42	25,28	0,44	1,72

Fonte: elaborado pelo próprio autor.

5.3 ANÁLISE DE CUSTOS

A Tabela 16 apresenta o detalhamento dos custos do protótipo discriminados por componente.

Tabela 16 — Análise de custos do protótipo

Componente	Quantidade	Valor (R$)
Sensor BH1750	1	12,90
Sensor DS18B20	1	11,40
Sensor DHT22	1	49,90
Sensor capacitivo de umidade	1	9,90
Resistor de aquecimento	1	79,00
Bomba peristáltica	1	68,90
Luminária LED	1	219,90
Ventoinha	2	45,56
Raspberry Pi 4 Model B	1	667,90
Módulo relé com 4 canais	1	54,51
Conversor ADS1115	1	28,90
Ponte H L298N	1	17,96
Fonte de alimentação	1	22,67
Total		**1.289,40**

Fonte: elaborado pelo próprio autor.

O custo total do protótipo foi de R$ 1.289,40, sendo o Raspberry Pi 4 o componente com maior custo, representando aproximadamente 52% do valor total.

5.4 CONSIDERAÇÕES FINAIS

Os testes realizados validaram o desempenho do sistema em ensaios individuais e em operação integrada de 72 horas.

Os testes individuais demonstraram o funcionamento adequado de cada subsistema isolado. O sistema de fotoperíodo apresentou resposta precisa aos comandos de acionamento com transições entre diferentes iluminâncias. A regulação térmica evidenciou que o aquecimento com ventilação forçada proporciona maior estabilidade. O sistema de umidade do ar, embora limitado ao monitoramento passivo pela ausência de dispositivos específicos de controle, registrou adequadamente as variações ambientais. O sistema de irrigação demonstrou capacidade satisfatória em elevar e manter diferentes níveis de umidade do solo.

O teste integrado confirmou a robustez do sistema em operação contínua e simultânea de todos os subsistemas. O fotoperíodo manteve estabilidade nas três etapas testadas, executando corretamente as transições entre diferentes configurações. A regulação térmica acompanhou as faixas progressivas estabelecidas, demonstrando controle efetivo ao longo do teste. A influência da iluminação sobre a temperatura ambiente foi identificada e adequadamente compensada pelo sistema de controle térmico

A umidade do ar apresentou variações naturais ao longo do teste, refletindo sua dependência da temperatura ambiente e confirmando a necessidade de dispositivos ativos para seu controle. O sistema de irrigação elevou progressivamente a umidade do solo conforme as faixas estabelecidas, demonstrando controle e estabilidade adequados. Por fim, a temperatura do solo seguiu a tendencia de variação da temperatura do ar.

6 CONCLUSÕES

O presente trabalho atingiu o objetivo de desenvolver um protótipo de estufa *indoor* baseado em IoT, integrando sensores, atuadores e plataformas digitais para monitoramento e controle automatizado de variáveis ambientais.

O sistema demonstrou capacidade de monitorar continuamente as variáveis críticas ao cultivo, integrando as informações ao Raspberry Pi 4 para processamento em tempo real. A comunicação com o Firebase possibilitou sincronização bidirecional entre o protótipo físico e o aplicativo móvel desenvolvido em React Native, viabilizando o gerenciamento remoto da estufa.

Os testes individuais validaram o desempenho de cada subsistema isoladamente, enquanto o teste integrado de 72 horas confirmou a operação simultânea e contínua de todos os componentes. O sistema demonstrou capacidade de executar transições automáticas entre diferentes configurações e manter a estabilidade das variáveis ao longo de períodos prolongados.

O fotoperíodo, a irrigação e a regulação térmica operaram conforme esperado, mantendo as condições dentro das faixas estabelecidas. A umidade do ar foi monitorada adequadamente, porém o sistema apresenta limitação quanto ao controle ativo desta variável, evidenciando a necessidade de dispositivos adicionais.

O custo total do protótipo foi de R$ 1.289,40, valor que evidencia a possibilidade de desenvolver sistemas de automação agrícola utilizando componentes de baixo custo e plataformas de código aberto.

Como trabalhos futuros, sugere-se a implementação de sistemas ativos de controle de umidade do ar, a integração de câmeras para monitoramento visual e a realização de estudos agronômicos específicos por cultura para definição das faixas de valores de cada fase de cultivo.

REFERÊNCIAS

AGRÍCOLAS, T. E. **Homepage**. 2025. Disponível em: https://tropicalestufasagricolas.com.br/. Acesso em: 25 set. 2025.

ALLDATASHEET. **Electronic Components Datasheet Search**. 2025. Disponível em: https://www.alldatasheet.com/. Acesso em: 13 out. 2025.

ASHTON, K. That ‘internet of things’ thing. **RFID Journal**, jun. 2009. Disponível em: https://www.rfidjournal.com/expert-views/that-internet-of-things-thing/73881/. Acesso em: 23 set. 2025

COSTA, N. L. **Calibração e avaliação de sensor capacitivo (FDR) para determinação da umidade em diferentes tipos de solos**. 2020. 72 f. Dissertação (Mestrado em Engenharia Agrícola) — Universidade Federal da Grande Dourados, Dourados, 2020. Disponível em: https://www.oasisbr.ibict.br/vufind/Record/BRCRIS_3224574cc3949c75ddcd58d0a8f97da8. Acesso em: 15 dez. 2025.

COUTO, Í. L. S. **Desenvolvimento de um sistema de gerenciamento e controle de estufas de pequeno porte**. 2024. 83 f. Trabalho de Conclusão de Curso (Graduação em Engenharia Eletrônica e de Telecomunicações) — Universidade Federal de Uberlândia, Patos de Minas, 2024. Disponível em: https://repositorio.ufu.br/handle/123456789/4365 9. Acesso em: 18 set. 2025.

ELETROGATE. **Eletrogate**. 2025. Disponível em: https://www.eletrogate.com/. Acesso em: 19 out. 2025.

EMBRAPA. **Visão 2030: o futuro da agricultura brasileira**. Brasília: Empresa Brasileira de Pesquisa Agropecuária, 2018. ISBN 978-85- 7035-799- 1. Disponível em: https://www.embrapa.br/documents/10180/9543845/Vis%C3%A3o+2030+-+o+futuro+da+agricultura+brasileira/2a9a0f27-0ead-991a-8cbf-af8e89d62829. Acesso em: 19 set. 2025.

FAO. **Estudo revela que Brasil é um dos países mais eficientes no uso da terra e insumos agrícolas em função de sua alta produção**. 2017. Disponível em: https://www.fao.org/brasil/noticias/detail-events/fr/c/1070557/. Acesso em: 19 set. 2025.

GUEDES Ítalo M. R. et al. Cultivo protegido de hortaliças em solo e em substrato. **Informe Agropecuário**, Belo Horizonte, v. 45, n. 326, p. 30–37, 2024. Disponível em: https://www.researchgate.net/publication/384043555_Cultivo_protegido_de_hortalicas_em_solo_e_em_substrato. Acesso em: 25 set. 2025.

IBGE. **Censo Agropecuário 2017: Resultados definitivos**. Rio de Janeiro: Instituto Brasileiro de Geografia e Estatística, 2019. ISSN 0103-6157. Disponível em: https://biblioteca.ibge.gov.br/visualizacao/periodicos/3096/agro_2017_resultados_definitivos.pdf. Acesso em: 19 set. 2025.

IBGE. **Segurança alimentar nos domicílios brasileiros volta a crescer em 2023**. 2024. Disponível em: https://agenciadenoticias.ibge.gov.br/agencia-noticias/2012-agencia-de-noticias/noticias/39838-seguranca-alimentar-nos-domicilios-brasileiros-voltaa-crescer-em-2023. Acesso em: 19 set. 2025.

JESUS, K. de. **Aplicação de Internet das Coisas (IoT) na Agricultura de Precisão**. 2021. 54 f. Trabalho de Conclusão de Curso (Graduação em Sistema de Informação) — Universidade Estadual de Goiás, Posse, 2021. Disponível em: http://aprender.posse.ueg.br:8081/jspui/handle/123456789/274. Acesso em: 23 set. 2025.

LISBINSKI, F. C. et al. Perspectivas e desafios da agricultura 4.0 para o setor agrícola. **Anais do VIII Simpósio da Ciência do Agronegócio**, Porto Alegre, p. 422–431, nov. 2020. Disponível em: http://hdl.handle.net/10183/218601. Acesso em: 23 set. 2025.

MAKERHERO. **Componentes Eletrônicos e Impressão 3D**. 2025. Disponível em: https://www.makerhero.com/. Acesso em: 13 out. 2025.

MINDBOWSER. **Firestore Database Vs Realtime Database: Which Is Right For Your Project?** 2025. Disponível em: https://www.mindbowser.com/firestore-databasevs-realtime-database/. Acesso em: 16 out. 2025.

MORORÓ, F. L. **Desenvolvimento de um protótipo de estufa inteligente para a agricultura com tecnologia IoT**. 2023. 84 f. Trabalho de Conclusão de Curso (Graduação em Engenharia Mecânica) — Universidade Federal de Pernambuco, Recife, 2023. Disponível em: https://repositorio.ufpe.br/handle/123456789/55665. Acesso em: 17 set. 2025.

MULATO, D. **Outliers**. 2022. https://www.linkedin.com/pulse/outliers-daniel-mulato/. Acesso em: 01 dez. 2025.

OLIVEIRA, C. D. et al. Detecção de fraudes, anomalias e erros em análise de dados contábeis: Um estudo com base em outliers. **REDECA**, v. 1, n. 1, p. 102–127, 2014. Disponível em: https://revistas.pucsp.br/redeca/article/view/23377. Acesso em: 11 dez. 2025.

ONU. **População mundial deve chegar a 9,7 bilhões de pessoas em 2050, diz relatório da ONU**. 2019. Disponível em: https://brasil.un.org/pt-br/83427-popula%C3% A7%C3%A3o-mundial-deve-chegar-97-bilh%C3%B5es-de-pessoas-em-2050-diz-rel at%C3%B3rio-da-onu. Acesso em: 19 set. 2025.

POLYGREENHOUSE. **Sistemas inteligentes de automação de estufas**. 2025. Disponível em: https://polygreenhouse.net/pt/sistemas-inteligentes-de-automacao-de-estufas/. Acesso em: 25 set. 2025.

PURQUERIO, L. F. V.; TIVELLI, S. W. Manejo do ambiente em cultivo protegido. In: **Manual técnico de orientação: projeto Hortalimento**. São Paulo: Codeagro, 2006. p. 15–29. Disponível em: https://www.bibliotecaagptea.org.br/administracao/educacao/artigos/MANEJO%20DO%20AMBIENTE%20EM%20CULTIVO%20PROTEGIDO.pdf. Acesso em: 25 set. 2025.

SANTOS, G. **Prototipação: Plataformas de hardware e software para sistemas embarcado**s. 2021. Disponível em: https://embarcados.com.br/prototipacao-plataform as-de-hardware-e-software-para-sistemas-embarcados/. Acesso em: 05 out. 2025.

SENTELHAS, P. C.; SANTOS, A. O. Cultivo protegido: aspectos microclimáticos. **Revista Brasileira de Horticultura Ornamental**, Campinas, SP, v. 1, n. 2, p. 108–115, mai. 1995. Disponível em: https://ornamentalhorticulture.com.br/rbho/article/view/99. Acesso em: 25 set. 2025.

SEO, S. **A review and comparison of methods for detecting outliers in univariate data sets**. 2006 — University of Pittsburgh, 2006. Disponível em: http://d-scholarship.pitt.edu/7948/. Acesso em: 20 nov. 2025.

SILVA, J. M. P.; CAVICHIOLI, F. A. O uso da agricultura 4.0 como perspectiva do aumento da produtividade no campo**. Revista Interface Tecnológica**, Taquaritinga, SP, v. 17, n. 2, p. 616–629, dez. 2020. Disponível em: https://revista.fatectq.edu.br/interfacetecnologica/pt_BR/article/view/1068. Acesso em: 23 set. 2025.

SOARES, J. O. **Protótipo de horta indoor com IOT**. 2023. 56 f. Trabalho de Conclusão de Curso (Graduação em Engenharia de Computação) — Universidade Federal de Santa Catarina, Araranguá, 2023. Disponível em: https://repositorio.ufsc.br/handle/123456789/253095. Acesso em: 17 set. 2025.

STROPARO, T. R. Transformação digital na agricultura: impactos da internet das coisas (IoT) na eficiência produtiva e sustentabilidade**. Lumen et Virtus**, v. 15, n. 38, p. 1573–1581, jul. 2024. Disponível em: https://periodicos.newsciencepubl.com/LEV/article/view/121. Acesso em: 17 set. 2025.

TANSLEY, D.; FLETCHER, S.; LONGSTAFF, A. P. Smart sensor for surface temperature measurement on manufacturing machines. **Proceedings of Computing and Engineering Annual Researchers' Conference 2013 : CEARC'13**, Huddersfield, p. 55–60, 2013. Disponível em: https://eprints.hud.ac.uk/id/eprint/19364/. Acesso em: 15 dez. 2025.

USP, J. da. **Estudo revela que Brasil é um dos países mais eficientes no uso da terra e insumos agrícolas em função de sua alta produção**. 2024. Disponível em: https://jornal.usp.br/atualidades/no-pais-agricultura-familiar-representa-77-dos-produtores-e-apenas-23-da-renda-agricola/. Acesso em: 19 set. 2025.

APÊNDICE A - ALGORITMOS

A.1 IMPLEMENTAÇÃO DO SENSOR BH1750

```
import board
import adafruit_bh1750



class BH1750:
    def __init__(self):
        try:
            i2c = board.I2C()
            self.sensor = adafruit_bh1750.BH1750(i2c)
        except Exception as e:
            self.sensor = None
            print(f"BH1750 nao encontrado: {e}")


    def ler_luminosidade(self):
        if not self.sensor:
            return None
        try:
            return self.sensor.lux
        except Exception as e:
            print(f"Erro ao ler BH1750: {e}")
            return None
```

A.2 IMPLEMENTAÇÃO DO SENSOR DS18B20

```
from w1thermsensor import W1ThermSensor



class DS18B20:
    def __init__(self):
        try:
            self.sensor = W1ThermSensor()
        except Exception as e:
            self.sensor = None
            print(f"DS18B20 nao encontrado: {e}")


    def read_temp(self):
        if not self.sensor:
            return None
        try:
            return self.sensor.get_temperature()
        except Exception as e:
            print(f"Erro ao ler DS18B20: {e}")
            return None
```

A.3 IMPLEMENTAÇÃO DO SENSOR DHT22

```
import adafruit_dht
import board
```

```
class DHT22:
    def __init__(self, pin=board.D17):
        self.sensor = adafruit_dht.DHT22(pin)

    def ler_dados(self):
        try:
            temperatura = self.sensor.temperature
            umidade = self.sensor.humidity
            if temperatura is not None and umidade is not None:
                return temperatura, umidade
            return None, None
        except RuntimeError as e:
            print(f"Erro na leitura do DHT22: {e}")
            return None, None
        except OverflowError as e:
            print(f"Overflow na leitura do DHT22: {e}")
            return None, None
        except Exception as e:
            print(f"Erro inesperado no DHT22: {e}")
            return None, None
```

A.4 IMPLEMENTAÇÃO DO SENSOR DE UMIDADE DO SOLO

```
import board
import busio
import adafruit_ads1x15.ads1115 as ADS
```

```
from adafruit_ads1x15.analog_in import AnalogIn



class UmidadeSolo:
    def __init__(self, canal=ADS.P0, seco=20307, molhado=9921):
        try:
            self.i2c = busio.I2C(board.SCL, board.SDA)
            self.ads = ADS.ADS1115(self.i2c)
            self.canal_umidade = AnalogIn(self.ads, canal)
        except Exception as e:
            print(f"Erro ao inicializar ADS1115: {e}")
            self.canal_umidade = None

        self.seco = seco
        self.molhado = molhado

    def ler_umidade(self):
        if not self.canal_umidade:
            return None
        try:
            leitura = self.canal_umidade.value
            umidade = 100 - (
                (leitura - self.molhado) / (self.seco - self.molhado) * 100
            )
            umidade = max(0, min(100, umidade))
            return round(umidade, 2)
```

```
            except Exception as e:
                print(f"Erro ao ler umidade do solo: {e}")
                return None
```

A.5 IMPLEMENTAÇÃO DO CONTROLE DO RESISTOR DE AQUECIMENTO

```
        import RPi.GPIO as GPIO


        class Aquecedor:

            def __init__(self, pino=10):

                self.pino = pino
                try:
                    GPIO.setmode(GPIO.BCM)
                    GPIO.setup(self.pino, GPIO.OUT)
                except Exception as e:
                    print(f"Erro ao inicializar GPIO do aquecedor: {e}")
                self.desligar()

            def ligar(self):

                try:
                    GPIO.output(self.pino, GPIO.LOW)
                except Exception as e:
                    print(f"Erro ao ligar aquecedor: {e}")
```

```
        def desligar(self):

            try:
                GPIO.output(self.pino, GPIO.HIGH)
            except Exception as e:
                print(f"Erro ao desligar aquecedor: {e}")

        def controlar(self, temperatura_ar, config):

            if not isinstance(config, dict):
                self.desligar()
                return False, "Configuração inválida"

            if temperatura_ar is None:
                self.desligar()
                return False, "Temperatura inválida"

            temp_desejada = config.get("TemperaturaDesejada")
            temp_min = config.get("TemperaturaMin", 0)
            temp_max = config.get("TemperaturaMax", 999)

            if temp_min > temp_max:
                print("Configuração inconsistente: TemperaturaMin >
TemperaturaMax")
                self.desligar()
```

```
            return False, "Configuração inconsistente"

        if temp_desejada is not None:
            if temperatura_ar < temp_desejada:
                self.ligar()
                return True, f"{temperatura_ar}C < desejada ({temp_desejada}C)"
            else:
                self.desligar()
                return False, f"{temperatura_ar}C >= desejada ({temp_desejada}C)"

        if temperatura_ar < temp_min:
            self.ligar()
            return True, f"{temperatura_ar}C < mínima ({temp_min}C)"

        if temperatura_ar >= temp_max:
            self.desligar()
            return False, f"{temperatura_ar}C >= máxima ({temp_max}C)"

        self.desligar()
        return False, f"{temperatura_ar}C entre {temp_min}C e {temp_max}C"
```

A.6 IMPLEMENTAÇÃO DO CONTROLE DA LUMINÁRIA

```
import RPi.GPIO as GPIO
from datetime import datetime, timedelta
```

```
class Luminaria:

    HORA_INICIO = "06:00"

    def __init__(self, pino=9):

        self.pino = pino
        try:
            GPIO.setmode(GPIO.BCM)
            GPIO.setup(self.pino, GPIO.OUT)
        except Exception as e:
            print(f"Erro ao inicializar GPIO da luminária: {e}")
        self.desligar()

    def ligar(self):

        try:
            GPIO.output(self.pino, GPIO.LOW)
        except Exception as e:
            print(f"Erro ao ligar luminária: {e}")

    def desligar(self):

        try:
            GPIO.output(self.pino, GPIO.HIGH)
        except Exception as e:
```

```
            print(f"Erro ao desligar luminária: {e}")

    def controlar(self, config):

        if not isinstance(config, dict):
            self.desligar()
            return False, "Configuração inválida"

        fotoperiodo = config.get("Fotoperiodo", 12)
        hora_atual = datetime.now().time()

        hora_inicio = datetime.strptime(self.HORA_INICIO, "%H:%M").time()
        hora_fim_dt = datetime.combine(datetime.today(), hora_inicio) +
timedelta(
            hours=fotoperiodo
        )
        hora_fim = hora_fim_dt.time()

        if fotoperiodo >= 24:
            self.ligar()
            return True, "Fotoperíodo 24h - ligada continuamente"

        if hora_inicio <= hora_fim:
            if hora_inicio <= hora_atual <= hora_fim:
                self.ligar()
                return (
```

```
                    True,
                    f"Dentro do fotoperíodo (06:00 > {hora_fim.strftime('%H:%M')})",
                )
            else:
                self.desligar()
                return (
                    False,
                    f"Fora do fotoperíodo (06:00 > {hora_fim.strftime('%H:%M')})",
                )

        else:
            if hora_atual >= hora_inicio or hora_atual <= hora_fim:
                self.ligar()
                return (
                    True,
                    f"Dentro do fotoperíodo cruzando a meia-noite (06:00 > {hora_fim.strftime('%H:%M')})",
                )
            else:
                self.desligar()
                return (
                    False,
                    f"Fora do fotoperíodo cruzando a meia-noite (06:00 > {hora_fim.strftime('%H:%M')})",
                )
```

A.7 IMPLEMENTAÇÃO DO CONTROLE DAS VENTOINHAS

```
import RPi.GPIO as GPIO


class Ventoinha:

    def __init__(self, pino=27):

        self.pino = pino
        try:
            GPIO.setmode(GPIO.BCM)
            GPIO.setup(self.pino, GPIO.OUT)
        except Exception as e:
            print(f"Erro ao inicializar GPIO da ventoinha: {e}")
        self.desligar()

    def ligar(self):

        try:
            GPIO.output(self.pino, GPIO.LOW)
        except Exception as e:
            print(f"Erro ao ligar ventoinha: {e}")

    def desligar(self):

        try:
```

```
            GPIO.output(self.pino, GPIO.HIGH)
        except Exception as e:
            print(f"Erro ao desligar ventoinha: {e}")

    def controlar(self, temperatura_ar, umidade_ar, aquecedor_ativo, config):

        if not isinstance(config, dict):
            self.desligar()
            return False, "Configuração inválida"

        if aquecedor_ativo:
            self.ligar()
            return True, "Ligada junto com o aquecedor"

        if temperatura_ar is None or umidade_ar is None:
            self.desligar()
            return False, "Leitura inválida de sensores"

        if config.get("OverrideUmidade", False):
            umi_desejada = config.get("UmidadeDesejada")
            if umi_desejada is not None:
                if umidade_ar > umi_desejada:
                    self.ligar()
                    return (
                        True,
```

```
                    f"Override: Umidade {umidade_ar}% > desejada ({umi_desejada}%)",
                )
            else:
                self.desligar()
                return (
                    False,
                    f"Override: Umidade adequada ({umidade_ar}% <= {umi_desejada}%)",
                )

        temp_desejada = config.get("TemperaturaDesejada")
        temp_max = config.get("TemperaturaMax", 999)
        umi_max = config.get("UmidadeMax", 999)

        if temp_desejada is not None and temp_desejada > temp_max:
            print("Configuração inconsistente: TemperaturaDesejada > TemperaturaMax")
            self.desligar()
            return False, "Configuração inconsistente"

        if temp_desejada is not None and temperatura_ar > temp_desejada:
            self.ligar()
            return (
                True,
                f"Temperatura {temperatura_ar}C > desejada ({temp_desejada}C)",
            )
```

```
        if temperatura_ar >= temp_max:
            self.ligar()
            return True, f"Temperatura {temperatura_ar}C >= limite ({temp_max}C)"

        if umidade_ar >= umi_max:
            self.ligar()
            return True, f"Umidade {umidade_ar}% >= limite ({umi_max}%)"

        self.desligar()
        return False, "Condições normais"
```

A.8 IMPLEMENTAÇÃO DO CONTROLE DA BOMBA PERISTÁLTICA

```
import RPi.GPIO as GPIO
from datetime import datetime
import threading
from utils.eventos import ciclo_reset_event


class Bomba:

    VAZAO_ML_POR_SEGUNDO = 1.66
    VOLUME_POR_IRRIGACAO = 5
    TEMPO_REACAO_UMIDADE = 120

    def __init__(self, pino_in1=5, pino_in2=6):
```

```
        self.pino_in1 = pino_in1
        self.pino_in2 = pino_in2

        try:
            GPIO.setmode(GPIO.BCM)
            GPIO.setup(self.pino_in1, GPIO.OUT)
            GPIO.setup(self.pino_in2, GPIO.OUT)
            GPIO.output(self.pino_in1, GPIO.LOW)
            GPIO.output(self.pino_in2, GPIO.LOW)
        except Exception as e:
            print(f"Erro ao inicializar GPIO da bomba: {e}")

        self.fim_ultima_irrigacao = None
        self.fase_inicio_irrigacao = None
        self.is_irrigando = False
        self._timer = None

    def ligar(self, duracao: float, fase_atual: str = None):

        if self.is_irrigando:
            return

        self.is_irrigando = True
        self.fase_inicio_irrigacao = fase_atual
```

```
        try:

            GPIO.output(self.pino_in1, GPIO.HIGH)
            GPIO.output(self.pino_in2, GPIO.LOW)
        except Exception as e:
            print(f"Erro ao ligar bomba: {e}")
            self.is_irrigando = False
            return

        self._timer = threading.Timer(duracao, lambda: self.desligar(reset=True))
        self._timer.start()

    def desligar(self, reset: bool = False, limpar_reacao: bool = False):

        try:

            GPIO.output(self.pino_in1, GPIO.LOW)
            GPIO.output(self.pino_in2, GPIO.LOW)
        except Exception as e:
            print(f"Erro ao desligar bomba: {e}")

        self.is_irrigando = False
        self.fase_inicio_irrigacao = None

        if self._timer:
            self._timer.cancel()
```

```
            self._timer = None

        if limpar_reacao:
            self.fim_ultima_irrigacao = None
        elif reset:
            self.fim_ultima_irrigacao = datetime.now()

        if reset:
            try:
                ciclo_reset_event.set()
            except Exception as e:
                print(f"Erro ao acionar ciclo_reset_event: {e}")

    def controlar(self, umidade_solo, config):

        if not isinstance(config, dict):
            self.desligar(reset=False)
            return False, "Configuração inválida"

        fase_atual = config.get("FaseAtual")

        if self.is_irrigando:
            if config.get("OverrideUmidadeDoSolo", False):

                alvo = config.get("UmidadeDoSoloDesejada")
                if alvo is not None:
```

```
                    return True, f"Override em andamento ({umidade_solo}% < {alvo}%)"
                else:
                    return True, f"Override em andamento (umidade {umidade_solo}%)"
            else:

                if fase_atual == self.fase_inicio_irrigacao:
                    umi_min = config.get("UmidadeDoSoloMin", 30)
                    return (
                        True,
                        f"Irrigação em andamento (umidade {umidade_solo}% < {umi_min}%)",
                    )
                else:

                    self.is_irrigando = False
                    self.fase_inicio_irrigacao = None
                    if self._timer:
                        self._timer.cancel()
                        self._timer = None

        if not isinstance(config, dict):
            self.desligar(reset=False)
            return False, "Configuração inválida"

        if umidade_solo is None:
```

```
                self.desligar(reset=False)
                return False, "Leitura inválida de umidade"

            if self.fim_ultima_irrigacao:
                tempo_passado = (datetime.now() - self.fim_ultima_irrigacao).total_seconds()
                if tempo_passado < self.TEMPO_REACAO_UMIDADE:
                    return (
                        False,
                        f"Aguardando reação ({int(tempo_passado)}s/{self.TEMPO_REACAO_UMIDADE}s)",
                    )

            if config.get("OverrideUmidadeDoSolo", False):
                alvo = config.get("UmidadeDoSoloDesejada")
                if alvo is not None and umidade_solo < alvo:
                    duracao = self._calcular_tempo_irrigacao()
                    self.ligar(duracao, fase_atual)
                    return True, f"Override: {umidade_solo}% < {alvo}%"
                else:
                    self.desligar(reset=False)
                    return False, f"Override: Umidade adequada ({umidade_solo}%)"

            umi_min = config.get("UmidadeDoSoloMin", 30)
            umi_max = config.get("UmidadeDoSoloMax", 80)
```

```
            if umi_min > umi_max:
                self.desligar(reset=False)
                return False, "Configuração inconsistente: Min > Max"

            if umidade_solo < umi_min:
                duracao = self._calcular_tempo_irrigacao()
                self.ligar(duracao, fase_atual)
                return True, f"Umidade baixa ({umidade_solo}% < {umi_min}%)"

            elif umidade_solo > umi_max:
                self.desligar(reset=False)
                return False, f"Solo muito úmido ({umidade_solo}% > {umi_max}%)"

            self.desligar(reset=False)
            return False, f"Umidade adequada ({umidade_solo}%)"

        def _calcular_tempo_irrigacao(self):

            return self.VOLUME_POR_IRRIGACAO / self.VAZAO_ML_POR_SEGUNDO
```

A.9 IMPLEMENTAÇÃO DO CICLO PRINCIPAL DE FUNCIONAMENTO

```
    import time
    from services.controle_service import controlar_atuadores
    from services.envio_service import iniciar_envio_periodico, parar_envio_periodico
```

```
from services.fases_service import verificar_e_avancar_fase
from services.coleta_service import coletar_dados
from config.firebase.realtime_utils import enviar_dados_realtime
from config.firebase.firestore_utils import atualizar_status_atuador
from config.local.loader import carregar_configuracao_local
from utils.eventos import ciclo_reset_event


def ciclo_estufa(
    estufa_id: str,
    luminosidade_sensor,
    temperatura_solo_sensor,
    temperatura_ar_sensor,
    umidade_solo_sensor,
    ventoinha,
    luminaria,
    bomba,
    aquecedor,
    tempo_ciclo: int,
) -> None:

    iniciar_envio_periodico(estufa_id)

    try:
        while True:
            try:
```

```
config = carregar_configuracao_local(estufa_id)

nova_fase = verificar_e_avancar_fase(estufa_id, config)

if nova_fase:
    config = carregar_configuracao_local(estufa_id)

dados = coletar_dados(
    luminosidade_sensor,
    temperatura_solo_sensor,
    temperatura_ar_sensor,
    umidade_solo_sensor,
)

status_atuadores = controlar_atuadores(
    ventoinha,
    luminaria,
    bomba,
    aquecedor,
    dados.get("TemperaturaDoArAtual") if dados else None,
    dados.get("UmidadeDoArAtual") if dados else None,
    dados.get("UmidadeDoSoloAtual") if dados else None,
    config,
)
if status_atuadores:
```

```
                for nome, (ativo, motivo) in status_atuadores.items():
                    atualizar_status_atuador(estufa_id, nome, ativo, motivo)

            if dados:
                enviar_dados_realtime(estufa_id, dados)

            print(f"Ciclo da estufa concluído às {time.strftime('%H:%M:%S')}")

        except Exception as e:
            print(f"Erro no ciclo_estufa: {e}")

        print(f"Aguardando próximo ciclo ({tempo_ciclo}s)...\n")
        if ciclo_reset_event.wait(timeout=tempo_ciclo):
            print("Ciclo resetado por listener!")
            ciclo_reset_event.clear()

finally:
    parar_envio_periodico()
```