Brian Andrade Nunes
Marco Aurélio Condé Oliveira Prado
Silas Lima e Silva

Sistema de cultivo hidropônico automatizado em ambiente urbano

São Paulo, SP

2023

Brian Andrade Nunes
Marco Aurélio Condé Oliveira Prado
Silas Lima e Silva

Sistema de cultivo hidropônico automatizado em ambiente urbano

Trabalho de conclusão de curso apresentado ao Departamento de Engenharia de Computação e Sistemas Digitais da Escola Politécnica da Universidade de São Paulo para obtenção do Título de Engenheiro.

Universidade de São Paulo – USP

Escola Politécnica

Departamento de Engenharia de Computação e Sistemas Digitais (PCS)

Orientador: Prof. Dr. Moacyr Martucci Junior

São Paulo, SP

2023

Resumo

Este trabalho busca desenvolver um sistema de cultivo hidropônico de pequeno porte, com a implementação de um sistema embarcado para acompanhamento e controle de variáveis do ambiente de cultivo, como pH da água, luminosidade, temperatura, umidade e nível dos nutrientes. Além disso, haverá a conexão com um aplicativo web de forma a viabilizar o acompanhamento remoto do sistema por parte do usuário.

Palavras-chave: hidroponia. sistema embarcado. aplicativo web.

Abstract

This study aims to develop a small-scale hydroponic cultivation system, implementing an embedded system to monitor and control variables such as water pH, light intensity, temperature, humidity and nutrient levels. In addition, the system will be connected to a web application to allow remote monitoring by the user.

Keywords: hydroponics. embedded system. web application.

Lista de ilustrações

Lista de tabelas

Sumário

1 Introdução

Neste trabalho, descrevemos o projeto de um sistema de cultivo hidropônico automatizado de pequeno porte, desenvolvido para uso em ambientes urbanos. Para isso, nesta seção apresentaremos uma visão geral sobre o problema.

1.1 Motivação

Ao se pensar no ambiente urbano como foco para nosso projeto, a aplicação de uma estufa hidropônica em pequena escala auxiliaria em pontos relacionados principalmente ao gerenciamento de espaço, considerando-se que a mesma pode ser construída em uma área menor do que um jardim convencional ocuparia, ainda mais considerando a indisponibilidade de se ter esse jardim em residências pequenas.

Em relação ao aspecto ambiental, a hidroponia é uma técnica que vem sendo fortemente desenvolvida nos últimos anos, especialmente pela sustentabilidade por dois motivos: gastar menos água que a agricultura tradicional e a possibilidade de ser feita sem o uso de pesticidas e herbicidas. Além disso, o ambiente de cultivo, por ser isolado do ambiente externo, pode ser melhor controlado, recebendo assim a quantidade adequada de água, luz e nutrientes, parâmetros estes que são controlados de forma automática pelo nosso sistema e que impactam diretamente na qualidade do alimento, na eficácia e produtividade do cultivo e na menor incidência de pragas e doenças na plantação.

1.2 Objetivo

O sistema de cultivo hidropônico automatizado é projetado para fornecer condições ideais para o crescimento de plantas sem o uso de solo em um ambiente urbano. Ele consiste em uma estrutura de cultivo, sistemas de iluminação, ventilação, irrigação, controle de pH e controle de nível de nutrientes controlados por um computador. Além disso, possui integração com um aplicativo *web*, permitindo ao usuário monitorar o desempenho e estado do sistema e estágio atual do cultivo, bem como controlar as configurações, receber alertas e criar, editar e selecionar um perfil de cultivo, tornando o processo mais prático.

1.3 Justificativa

O projeto de uma estufa hidropônica automatizada gera grande valor para aqueles que desejam cultivar plantas e/ou alimentos frescos em casa, seja por interesse na qualidade

do cultivo ou por *hobby*, visto que o cultivo de plantas em casa foi uma atividade que muitas pessoas começaram a fazer durante os dois anos da pandemia de Covid-19.

Além disso, ao se analisar o mercado dessa área, existem apenas soluções parciais para essa atividade, como estufas para cultivo tradicional e hortas hidropônicas de pequeno porte, nas quais nenhuma delas envolve os conceitos de automação que serão aplicados neste projeto.

1.4 Organização do Trabalho

Para se iniciar o desenvolvimento do projeto, antes fora necessário entender melhor os conceitos relacionados à hidroponia, para que assim fosse possível dimensionar o que e como seria possível automatizar o processo de cultivo. Estes aspectos conceituais estão descritos no Capítulo 2.

No Capítulo 3, descrevemos o planejamento para o desenvolvimento do trabalho, detalhando os prazos e metas estipulados em cada etapa.

No Capítulo 4, definimos os principais requisitos funcionais e não funcionais do projeto, etapa essencial para se garantir que o projeto seja bem planejado e executado.

No Capítulo 5, detalhamos todo o processo de desenvolvimento do trabalho, incluindo as tecnologias e materiais utilizados, processo da montagem da estrutura física, implementação do sistema embarcado e problemas enfrentados durante a execução do projeto.

No Capítulo 6, analisamos os resultados obtidos neste projeto e detalhamos todas as oportunidades mapeadas para continuação do trabalho, de forma a acrescentar novas funcionalidades e aprimorar as existentes.

2 Aspectos Conceituais

Nesta etapa da documentação, abordaremos os conceitos de hidroponia, bem como a aplicação da automatização deste processo e as vantagens acadêmicas e de negócios associadas.

2.1 Hidroponia

A hidroponia é um método de cultivo de plantas sem o uso do solo, onde as raízes das plantas são mantidas em uma solução nutritiva, em vez de estarem em contato direto com o solo (LEE; LEE, 2015). Nesse formato de cultivo, as plantas então obtêm todos os nutrientes que precisam diretamente da água. Dessa forma, é possível ter um controle preciso dos nutrientes e da água, resultando em um crescimento mais rápido e saudável das plantas, além de uma maior qualidade no produto final, em decorrência do não-uso de inseticidas e produtos químicos agressivos.

Neste contexto, será utilizado o método *Nutrient Film Technique* (NFT), um dos métodos mais comuns de hidroponia (SILVA et al., 2021). Neste método, as plantas são cultivadas em canais inclinados, onde a solução nutritiva é constantemente bombeada e recirculada, formando uma película fina de líquido, onde as raízes repousarão. Esse fluxo é controlado para que as plantas recebam a quantidade ideal de nutrientes e oxigênio (quantidades essas ajustadas automaticamente), sem que haja excesso ou falta de água.

A hidroponia pode ser utilizada em uma variedade de cenários, desde galpões de cultivo para escala industrial à cultivos domésticos. Neste último, hidroponia é uma excelente opção para quem deseja cultivar plantas em espaços limitados, como apartamentos e casas, ou em locais com condições climáticas desfavoráveis. Além disso, a hidroponia doméstica permite estender o cultivo para o ano inteiro, sem que ele seja afetado pela estação e/ou o clima.

Em geral, a hidroponia é uma técnica de cultivo eficiente, sustentável e flexível, que oferece muitas vantagens em relação ao cultivo tradicional. O método NFT é uma excelente opção para quem deseja iniciar um cultivo hidropônico doméstico, em decorrência da sua simplicidade e eficiência.

2.2 Automatização

O processo de automatização da hidroponia em escala residencial visa facilitar o processo de cultivo doméstico, além de permitir um maior controle do ambiente das

plantas, uma vez que o mesmo é medido e ajustado constantemente. Estas medições e ajustes podem ser feitos, por exemplo, através de um micro computador como o *Raspberry Pi* (LUKITO; LUKITO, 2019)

É importante manter em vista vários aspectos ambientais da estufa hidropônica, como: temperatura do ar e da água, ciclo de luz das plantas, pH e condutividade elétrica da água (utilizada como referência para o ajuste dos nutrientes) e umidade do ar. Estes valores são ajustados para faixas pré-definidas a partir de cada planta que pode ser cultivada, utilizando: uma bomba para manutenção do fluxo d'água; bombas peristálticas para ajuste dos níveis de pH e nutrientes; ventilador e exaustor para controle da temperatura e umidade (ARORA et al., 2021). Além disso, outros fatores menos aparentes também devem ser observados, como fluxo e nível de água, para garantir a segurança da bomba utilizada, e também que o reservatório nunca esvazie. Por fim, o sistema não pode ser interrompido em hipótese alguma, devido à fragilidade do cultivo a cortes de água e nutrientes.

Com estes aspectos em mente, é possível extrair do usuário grande parte das responsabilidades de cultivo, bem como diminuir a necessidade de conhecimentos prévios sobre o plantio, tornando toda a experiência que envolve a planta um processo muito mais liso e prazeroso.

2.3 Pesquisa

O processo de automatização de cultivo também gera ganhos para a pesquisa no campo da agricultura. Isso se deve ao fato de que a partir do constante monitoramento do cultivo e do crescimento das plantas é possível traçar a relação entre o ambiente e o resultado obtido, e conforme a base de dados é alimentada, processos de *Machine Learning* podem ser aplicados de modo a reajustar os modelos pré-configurados (SRINIDHI; SHREENIDHI; VISHNU, 2020) para que seu cultivo seja mais rápido, produtivo e econômico.

Uma estufa em pequena escala como a proposta nesse projeto pode ser utilizada como plataforma de pesquisa, onde dados sobre as plantas cultivadas poderão ser coletados de forma padronizada constantemente, sem a necessidade de medições manuais. Esse tipo de plataforma tem, portanto, o potencial de facilitar a rotina de um pesquisador, funcionando como uma importante ferramenta.

Além disso, o processo científico tem fortes ganhos em plantas que hoje não possuem grandes bases de cultivo, e, portanto, possuímos poucos conhecimentos de qual o melhor ambiente e condições para elas.

2.4 Produto

Com a crescente demanda por cultivos domésticos, as maiores barreiras que o público encontra são a dificuldade de manutenção do cultivo e a necessidade de grande conhecimento prévio. A automatização do processo ataca diretamente esses dois pontos, uma vez que a estufa se encarrega de grande parte da manutenção, e sinaliza ao usuário exatamente o que ele deve fazer nas poucas interações que são necessárias. Além disso, a pré-configuração de perfis para plantas reduz drasticamente a necessidade do usuário conhecer e entender as necessidades individuais das espécies e se responsabilizar pelo ajuste do ambiente para as mesmas.

Desta forma, do ponto de vista de negócios e produto, a estufa tem potencial para atingir este público alvo. Entretanto, ainda é importante ressaltar que um ponto negativo deste tipo de cultivo ainda não é sanado: o custo. Note que o custo inicial para uma produção hidropônica não é baixo, principalmente quando a automatizamos com sensores e atuadores. Entretanto, é importante frisar também que o público em questão não busca um produto que seja financeiramente benéfico, mas sim, a inovação do equipamento e a possibilidade do cultivo do próprio alimento sem a necessidade de muita atenção e estudo.

Desta forma, a estufa é um projeto de produto viável, principalmente ao vermos que no mercado não há nada que atinja os problemas trabalhados aqui.

3 Método do trabalho

Neste capítulo, será descrito o planejamento adotado para o desenvolvimento do nosso projeto, detalhando as atividades, os prazos e as metas que foram definidos para cada etapa, conforme Tabela 1.

Etapa	**Tarefa**	**Período**
I	Estudo do problema e levantamento de requisitos	Janeiro - Fevereiro
II	Definição da estrutura física da estufa	Março - Abril
III	Construção da estrutura física da estufa	Maio - Julho
IV	Desenvolvimento da aplicação *front-end*	Agosto - Outubro
V	Desenvolvimento da aplicação *back-end*	Agosto - Outubro
VI	Desenvolvimento do sistema embarcado	Setembro - Outubro
VII	Integração do sistema	Outubro - Novembro
VIII	Testes e refinamento	Novembro - Dezembro

Tabela 1 – Cronograma das etapas do projeto

3.1 Diagrama de Gantt

A partir da especificação do projeto, foi possível definir um diagrama de Gantt a fim de organizarmos o desenvolvimento e termos uma perspectiva ampla dos objetivos e dependências.

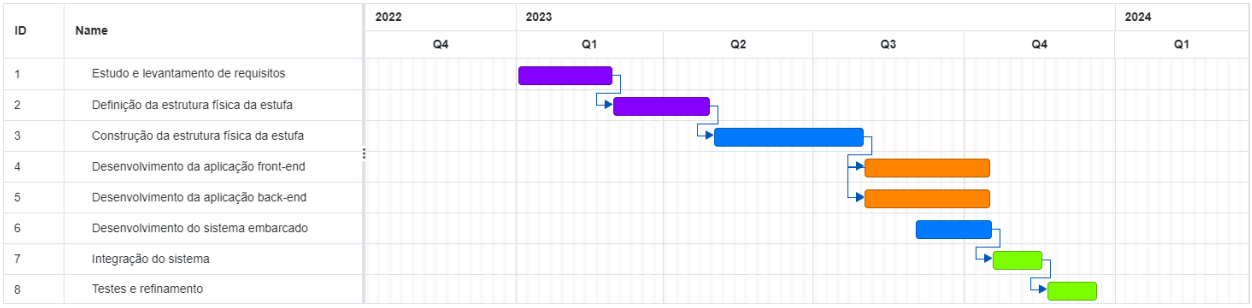

Figura 1 – Diagrama de Gantt inicial

3.2 Diagrama PERT

Assim como com o diagrama de Gantt, também elaboramos um diagrama PERT para melhor planejamento e definição do projeto, evitando assim problemas inesperados que afetassem diretamente as datas de entrega do projeto.

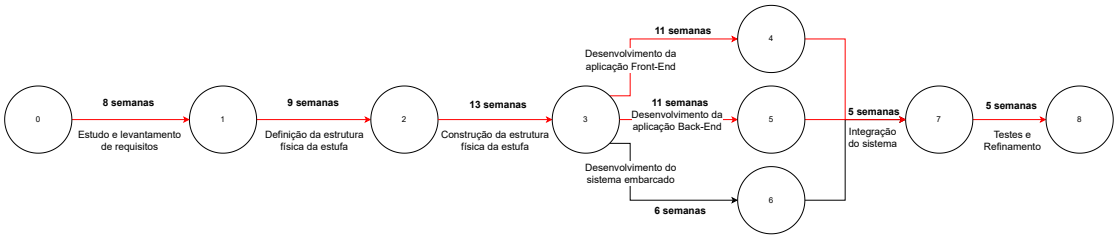

Figura 2 – Diagrama PERT inicial

3.3 Estudo do problema e levantamento de requisitos

Nesta etapa, o principal foco foi entender o problema e, partindo disso, definir todos os requisitos essenciais para o funcionamento correto do projeto. Assim, após conversar com o professor orientador e pesquisar sobre os conceitos fundamentais descritos no Capítulo 2, elencamos as principais funcionalidades que nosso sistema deveria executar e, então, definimos os requisitos funcionais e não funcionais, disponíveis no Capítulo 4.

3.4 Definição da estrutura física da estufa

Antes de iniciar o desenvolvimento do sistema embarcado, consideramos essencial ter pronto o espaço físico onde o cultivo é feito. Para isso, buscamos modelos de estufas hidropônicas caseiras que atendessem os requisitos propostos, pensando sempre em garantir uma estrutura onde a integração do sistema embarcado (e, por consequência, automatização do processo de cultivo) fosse possível. A partir disso, orçamos cada peça necessária para a construção.

3.5 Construção da estrutura física da estufa

Após a definição de como seria a estufa, o próximo passo foi comprar as peças e começar a montagem. Nesta etapa, estão inclusos tanto os processos necessários para certos componentes da estufa, como a montagem da estrutura de circulação da água e a impermeabilização do reservatório, quanto a montagem da estrutura completa.

3.6 Desenvolvimento da aplicação *front-end*

Nesta etapa, foi desenvolvido o módulo *front-end* do aplicativo para *web*. A escolha de optar pela plataforma *web* em vez de *mobile* foi feita pensando na possibilidade do usuário final poder se conectar ao sistema de qualquer dispositivo, incluindo computadores,

laptops, *tablets* e celulares. Para isso, nos preocupamos com a responsividade da aplicação, de forma a garantir uma boa usabilidade independente do aparelho de acesso.

3.7 Desenvolvimento da aplicação *back-end*

Com a definição de todos os requisitos, foi possível iniciar o desenvolvimento de um sistema *back-end*. Sua responsabilidade é de integrar a comunicação dos diversos agentes envolvidos no projeto: sensores, atuadores, aplicação e serviço *web*, entre outros. Utilizando *Flask,* um popular *microframework* da linguagem *Python*, foi desenvolvida uma API que promove a integração dos sensores e atuadores com os agentes externos, diponibilizando dados referentes às medições dos sensores, estados dos atuadores, imagens de câmera, entre outros. Além disso, o servidor *back-end* também é responsável por gerir a manutenção da estufa, isto é, a periódica leitura de sensores e ação de atuadores para mantimento dos parâmetros dentro dos limites desejados.

3.8 Desenvolvimento do sistema embarcado

Nesta etapa, foi montado o sistema embarcado do nosso projeto, que conta com os sensores e atuadores necessários para o controle e sensoriamento da estufa, além do *Raspberry Pi* e módulos auxiliares utilizados para implementação dos algoritmos necessários.

3.9 Integração do sistema

Nesta etapa, foram ajustadas e validadas todas as integrações entre as diferentes partes do sistema (aplicações *front-end* e *back-end*, sistema embarcado e estrutura física). Vale ressaltar que parte dessa etapa naturalmente se dividiu nas etapas anteriores, conforme certos passos do desenvolvimento foram sendo atingidos.

3.10 Testes e refinamento

Nesta última etapa, foram realizados diferentes cenários de testes, buscando garantir que todos os comportamentos e objetivos do projeto fossem alcançados. A partir disso, eventuais correções e/ou oportunidades de melhoria foram mapeadas e executadas.

4 Especificação de Requisitos

As entradas do sistema incluem os parâmetros de cultivo desejados, como temperatura, umidade, intensidade de luz, pH da água e nutrientes. O sistema também suporta a entrada de perfis de plantas pré-configurados, permitindo ao usuário selecionar as configurações ideais para uma determinada espécie sem precisar configurar manualmente cada parâmetro.

Os sensores instalados no sistema medem continuamente as condições ambientais e transmitem esses dados para o sistema de controle, que é capaz de ajustar automaticamente os parâmetros de cultivo para manter as condições ideais para o crescimento das plantas de uma forma mais eficiente. Esses dados também são transmitidos para o aplicativo, permitindo que o usuário acompanhe as condições de cultivo.

As saídas do sistema incluem o estado detalhado sobre as condições de cultivo, bem como alertas de falhas ou anormalidades detectadas pelos sensores, tudo disponível no aplicativo, permitindo que o usuário acompanhe assim o progresso do plantio. O sistema possui também uma câmera, responsável por transmitir imagens da estufa para o aplicativo quando requisitado pelo usuário. Também são realizadas capturas periódicas do espaço, gerando assim um *time-lapse* com a evolução do cultivo.

Os requisitos de desempenho incluem a capacidade de manter as condições de cultivo dentro de uma faixa específica, a capacidade de ajustar automaticamente os parâmetros de cultivo em resposta a mudanças nas condições ambientais e a entrada de perfis de plantas, a capacidade de fornecer dados atualizados e alertas de falhas, além da capacidade de integrar com um aplicativo *web*, permitindo ao usuário uma maior integração com o sistema mesmo à distância.

4.1 Requisitos funcionais

- Leitura e ajuste de umidade, temperatura d'água, temperatura do ar, pH e luminosidade dentro da tenda de cultivo;
- Ajuste de faixas desejadas para os parâmetros de cultivo via aplicativo;
- Criação e uso de perfis pré-configurados de parâmetros de cultivo para diferentes tipos de planta;
- Visualização de imagens da estufa;

- Envio de alertas via notificação *push* em caso de falhas e/ou situações adversas no sistema;
- Integração com a *Alexa* (assistente virtual da *Amazon*) para acompanhamento do status da estufa.

4.2 Requisitos não funcionais

- Precisão na leitura das variáveis do ambiente:
  - Margem de erro de 0.1 para a leitura do pH
  - Margem de erro de 0.5 °C para as leituras de temperatura do ar e da água
  - Margem de erro de 5 % para a leitura da umidade
  - Margem de erro de 20 ppm para a leitura da condutividade elétrica
- Tempo de resposta dos atuadores inferior à 1 s;
- Tempo de envio de imagens da câmera inferior à 1 s;
- Disponibilidade dos servidores *back-end* e *front-end* superior à 99 %;
- Usabilidade intuitiva do aplicativo *web*;
- Responsividade do aplicativo *web*;

5 Desenvolvimento do Trabalho

5.1 Diagrama de Gantt

Ao longo do desenvolvimento, devido aos problemas enfrentados e também aos passos que se mostraram mais simples do que inicialmente se foi pensado, o diagrama de Gantt sofreu alterações em relação ao planejamento inicial. Abaixo, é possível verificar o diagrama de Gantt que descreve o desenvolvimento real do projeto:

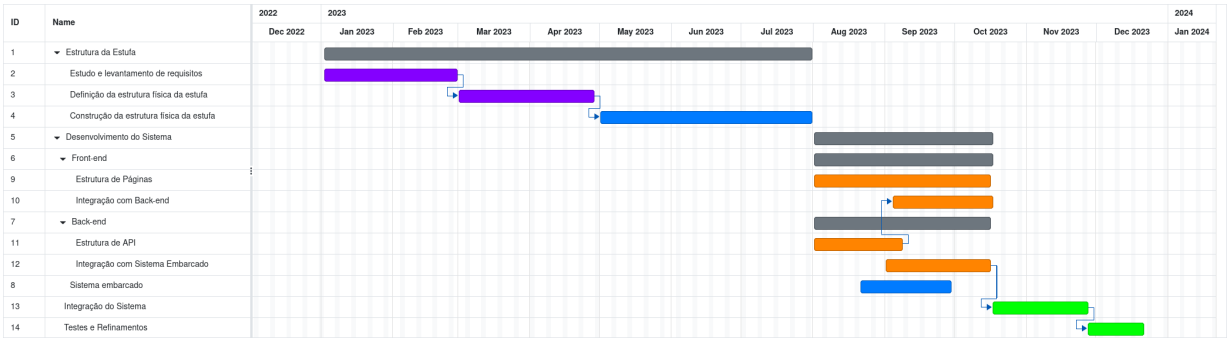

Figura 3 – Diagrama de Gantt final

5.2 Tecnologias e Materiais Utilizados

5.2.1 Estrutura física da estufa

O passo inicial para o desenvolvimento do projeto foi a construção da estrutura física de uma estufa para cultivo hidropônico com controle manual, visto que as etapas seguintes são a automatização desse processo através da integração com o sistema embarcado. Para isso, foram necessários os seguintes materiais:

- 1 Tenda para cultivo - 60 cm x 60 cm x 140 cm
- 1 Mangueira - 1 m de comprimento com 40 mm de diâmetro
- 1 Mangueira - 1 m de comprimento com 15 mm de diâmetro
- 1 Microtubo - 20 m de comprimento com 5 mm de diâmetro
- 1 Cano de PVC - 60 cm de comprimento com 32 mm de diâmetro
- 4 Canos de PVC - 60 cm de comprimento com 75 mm de diâmetro

- 2 Canos de PVC - 60 cm de comprimento com 150 mm de diâmetro
- 2 Tampas para cano de PVC - 32 mm de diâmetro
- 4 Tampas para cano de PVC - 75 mm de diâmetro
- 3 Tampas para cano de PVC - 150 mm de diâmetro
- 1 Caixa d'água - 56 l
- 1 Bomba d'água - 1500 l/h
- 4 Bombas peristálticas - 12 V
- 1 Painel de LED - 100 W
- 1 Ventilador - 200 mm de diâmetro
- 1 Filtro de carvão para estufas
- 1 Exaustor para estufas
- 1 Chapa de MDF - 600 mm x 300 mm x 3 mm
- 20 Parafusos - 3 mm x 15 mm
- 10 Parafusos - 3 mm x 20 mm
- 30 Porcas sextavadas - Rosca de 3 mm
- Abraçadeiras de *nylon* - 3 mm x 150 mm
- Corda plástica - 2 m

5.2.2 Circuito eletrônico

Para a montagem do circuito eletrônico, que inclui os sensores e atuadores relacionados com o principal objetivo do nosso projeto, buscamos montar um protótipo utilizando *protoboard* e cabos para interligação dos componentes.

- 1 *Raspberry Pi* 4
- 1 Arduino Uno
- 1 Câmera Logitech C270
- 1 Sensor de nível d'água
- 1 Sensor de temperatura d'água (DS18B20)

- 1 Sensor de pH (pH-4502C)
- 1 Sensor de condutividade elétrica d'água (TDS Meter V1.0)
- 1 Sensor de temperatura do ar e umidade (DHT22)
- 1 Fonte DC 12 V 5 A
- 1 *Plug* P4
- 1 Resistor 4.7 kΩ
- 1 Resistor 10 kΩ
- 1 *Protoboard*
- 2 Módulos *relé* serial 4 canais
- 2 Rabichos 10 A - 2 m
- 1 Cabo - 15 m e 0,3 mm de seção
- 1 Conjunto de tomada tripla - 10 A
- 1 Filtro de linha 6 tomadas - 10 A
- *Jumpers* macho-fêmea
- *Jumpers* macho-macho
- Kit de conectores *Dupont*

5.2.3 Aplicações *front-end* e *back-end*

Em paralelo à construção da estufa física, a fim de otimizarmos o tempo disponível, foi iniciado o desenvolvimento da plataforma digital da estufa, que possibilita monitoramento e interação com o ambiente de cultivo de forma remota, rápida e intuitiva. O desenvolvimento das aplicações foi separado entre as plataformas *front-end* e *back-end*, e as tecnologias utilizadas para cada foram:

- ***Front-end***
  - *JavaScript*
  - *TypeScript*
  - *React*
  - *Node.js*

- ***Back-end***
  - *Python*
  - *Flask*
  - *Gunicorn*
  - *Ntfy*

Para garantir a colaboração em equipe, o versionamento do código e o armazenamento em nuvem, utilizamos a plataforma *GitHub*, dividindo o projeto em um repositório para o aplicativo *front-end* e um repositório para o aplicativo *back-end*.

5.3 Projeto e Implementação

Com os devidos materiais e tecnologias necessários para o desenvolvimento do sistema de cultivo hidropônico automatizado, podemos agora discutir como a solução proposta foi, de fato, projetada e implementada.

5.3.1 Estrutura física da estufa

Os canos de PVC listados na seção anterior servem como local para plantio das raízes no sistema de cultivo e para circulação da água. Para a montagem da estrutura de cultivo, eles foram fixados entre si utilizando cola instântanea e, para evitar vazamentos, as mangueiras e os microtubos foram vedados com cola de silicone. Além disso, alguns cortes e furos foram feitos, conforme descrito a seguir:

- Cano de PVC de 32 mm de diâmetro
  - 7 furos de 5 mm de diâmetro para encaixe dos microtubos que se conectam aos canos de 75 mm e ao cano de 150 mm
  - 1 furo de 15 mm de diâmetro para encaixe da mangueira que se conecta à bomba d'água
- 2 Canos de PVC de 75 mm de diâmetro
  - 1 furo de 5 mm de diâmetro para encaixe do microtubo que se conecta ao cano de 32 mm
  - 2 furos de 50 mm de diâmetro para plantio de raízes de pequeno/médio porte
- 2 Canos de PVC de 75 mm de diâmetro
  - 1 furo de 5 mm de diâmetro para encaixe do microtubo que se conecta ao cano de 32 mm

  - 3 furos de 50 mm de diâmetro para plantio de raízes de pequeno/médio porte

- Cano de PVC de 150 mm de diâmetro
  - 3 furos de 5 mm de diâmetro para encaixe dos microtubos que se conectam ao cano de 32 mm
  - 1 furo de 100 mm de diâmetro para plantio de raízes de médio/grande porte
  - 1 corte em V com 150 mm de largura feito em uma das extremidades do cano, de forma a encaixar no outro cano de 150 mm que funciona como calha do sistema

- Cano de PVC de 150 mm de diâmetro
  - 1 corte de 75 mm de largura na lateral do cano, para encaixe dos canos onde acontece o plantio
  - 1 furo de 40 mm de diâmetro para encaixe da mangueira de vazão para a caixa d'água

A estrutura do sistema de cultivo fica completamente dentro da tenda de cultivo, de forma a garantir o isolamento do ambiente a ser controlado. No chão da tenda, se localiza a caixa d'água, com a bomba d'água submersa. Acima dela, a estrutura de PVC descrita anteriormente. No teto da tenda, ficam pendurados o painel de LED e o exaustor. Além disso, a chapa de MDF com a montagem do circuito eletrônico fica presa nas hastes superiores e laterais da parede traseira. Por fim, no canto esquerdo superior da parede frontal, o ventilador se encontra preso na haste por meio de um pregador.

Figura 4 – Montagem completa do sistema de cultivo

5.3.2 Sistema embarcado

5.3.2.1 Sensores

Os sensores do sistema são fundamentais para o controle da estufa, e seu uso é feito em uma parte isolada da água que é alimentada por um dos tubos de distribuição de água do sistema. Dessa forma, evitamos leituras erradas ocasionadas por sensores estarem muito proximos das bombas peristáticas. Para a estufa, utilizamos os seguintes sensores no monitoramento do ambiente:

- Temperatura do ar e umidade (DHT22): O sensor DHT22 é um sensor de temperatura e umidade digital, sendo amplamente utilizado por fornecer leituras precisas de forma simples, tornando-o ideal para aplicações de controle de clima em estufas e sistemas de monitoramento de condições meteorológicas.
- Luminosidade (LDR): O LDR é um sensor que varia sua resistência de forma inversamente proporcial à intensidade da luz incidente, ou seja, quanto mais luz incide sobre o sensor, menor é sua resistência. Isso o torna adequado para detectar níveis de luminosidade em ambientes e é comumente usado em projetos que envolvem

controle de iluminação automática, como luzes de rua que acendem automaticamente ao anoitecer.

- pH (PH4502C): O sensor PH4502C é um sensor de pH que permite medir o nível de acidez ou alcalinidade de um líquido. Ele é frequentemente utilizado em aplicações que envolvem controle de qualidade de água, aquários e sistemas de cultivo hidropônico. O sensor gera leituras de pH precisas e ajuda a manter os níveis adequados de pH em soluções aquosas.

- Condutividade elétrica d'água (TDS): O sensor de condutividade é projetado para medir a condutividade elétrica de uma solução aquosa. Ele fornece informações sobre a quantidade de íons dissolvidos na água, o que é importante para controlar a concentração de nutrientes e monitorar a qualidade da água em sistemas de cultivo hidropônico.

- Temperatura d'água (DS18B20): O sensor DS18B20 é um sensor de temperatura digital à prova d'água e é comumente usado para medir a temperatura de líquidos, tornando-o adequado para aplicações em que é necessário monitorar a temperatura da água, como aquários, sistemas de aquecimento de piscinas e monitoramento ambiental.

- Nível d'água: O sensor de nível d'água é utilizado para detectar o nível de água em reservatórios, tanques e outros recipientes. Ele pode ser empregado em sistemas de controle de enchimento ou esvaziamento de reservatórios, sistemas de monitoramento de níveis de água em poços e muito mais. Esse sensor desempenha um papel fundamental na automação de processos que envolvem o uso e controle da água.

Sensor	**Conexão**
PH4502C	*Arduino Analog* 0
LDR	*Arduino Analog* 2
TDS	*Arduino Analog* 3
DS18B20	*Arduino Digital* 2
DHT22	*Arduino Digital* 3
Nível d'água	*Raspberry Pi* GPIO 24

Tabela 2 – Conexão dos sensores no circuito eletrônico

5.3.2.2 Atuadores

Todos os atuadores do sistema são controlados via *relés*. Dessa forma, do ponto de vista do sistema, um sinal lógico é enviado a estes componentes que ligam e desligam os atuadores que serão descritos em detalhes:

- Bombas peristáticas: O sistema possui 4 bombas peristáticas responsáveis pelas dosagens dos nutrientes A e B, além dos controladores de pH. As bombas são utilizadas no processo periódico de manutenção, no qual os valores de condutividade e pH são lidos e, através dessa informação, o sistema toma a decisão de quais bombas acionar e por quanto tempo. Para isso, os valores lidos são avaliados de acordo com os limites que a estufa está configurada para manter, e a quantidade de solução dispensada na água de cultivo é calculada proporcionalmente à essa variação.

- Exaustor e ventilador: São utilizados para controle de temperatura e umidade da estufa. Uma vez que estes valores extrapolem os limites definidos, estes atuadores são acionados, a fim de controlar estas propriedades do ar da estufa.

- Luz: Este último não responde a sensores, mas sim à cronogramas definidos nos perfis de cultivo. Na manutenção periódica, é avaliado o último horário válido do cronograma das luzes, que define se elas devem estar acessas ou apagadas. Reforçamos que, caso as luzes devam estar acessas e o sensor de luz detecte que a luminosidade está baixa, é enviada uma notificação para o usuário, uma vez que isto sinaliza um possível mau funcionamento do circuito de iluminação da estufa.

Atuador	**Conexão**
Bomba pH+	*Raspberry Pi* GPIO 5
Bomba pH-	*Raspberry Pi* GPIO 6
Bomba Nutriente B	*Raspberry Pi* GPIO 16
Painel de Luz	*Raspberry Pi* GPIO 17
Exaustor	*Raspberry Pi* GPIO 22
Bomba Nutriente A	*Raspberry Pi* GPIO 26
Ventilador	*Raspberry Pi* GPIO 27

Tabela 3 – Conexão dos atuadores no circuito eletrônico

5.3.2.3 *Arduino*

Os sensores de pH, condutividade elétrica e luminosidade fornecem os dados de suas medições em sinais analógicos, sendo assim não é possível fazê-los interagir diretamente com o *Raspberry Pi*, que não possui entradas diretas para este tipo de dado. Assim, foi utilizado um Arduino que é capaz de ler e interpretar estes dados, utilizando bibliotecas para abstrair e simplificar toda a complexidade deste processo. Através da comunicação serial, o *Raspberry Pi* envia o caracter 'R' para o *Arduino* sempre que deseja executar uma leitura desses sensores, e o *Arduino*, por sua vez, envia os dados desses leitores através da sua saída serial. A utilização do sinal 'R' para leitura dos dados se deu para que o envio de dados por parte do *Arduino* fosse efetuado somente quando necessário. Desta forma, não há necessidade da utilização de *threads*, que prejudicariam o desempenho do sistema, seja

para leitura constante da saída serial, ou com o enfileiramento de dados que ocasionaria o eventual descasamento dos dados lidos com a realidade presente na estufa.

5.3.2.4 Circuito eletrônico

O elemento central do nosso circuito eletrônico é o *Raspberry Pi*, onde são executadas nossas aplicações *back-end* e *front-end*. Conectado a ele, temos o sensor de nível d'água, a câmera, o *Arduino Uno* e os dois módulos *relé* serial.

Através dos módulos *relé*, o *Raspberry Pi* pode controlar diretamente a ativação de cada atuador do sistema. Para isso, um dos módulos é alimentado por uma fonte DC 12 V 5 A e se conecta às bombas peristálticas para ajuste do nível de pH e de nutrientes. Já o outro módulo, alimentado com uma tensão de 110 V ligado no filtro de linha, se conecta diretamente ao exaustor, ventilador e painel de iluminação.

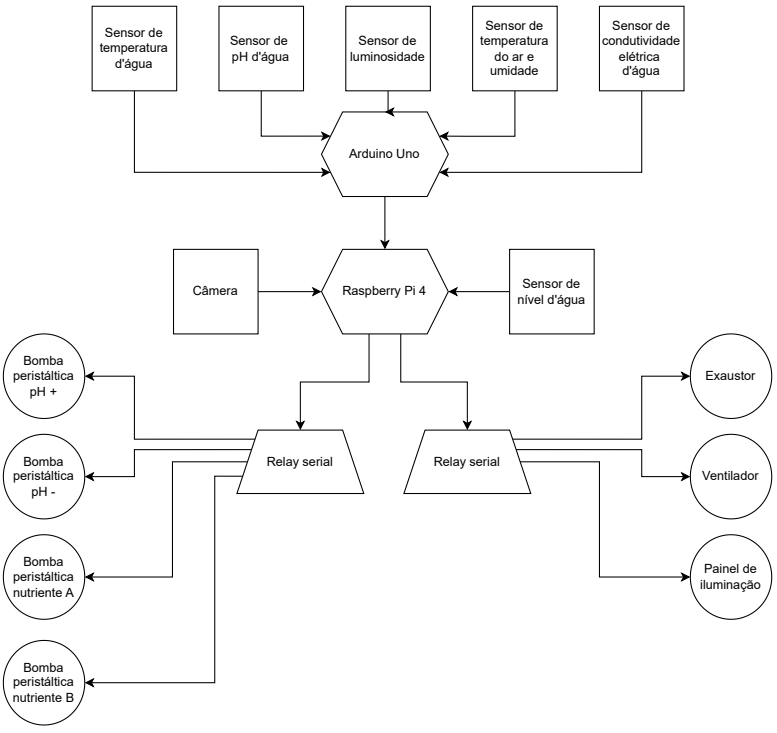

Figura 5 – Diagrama geral de organização dos componentes do sistema

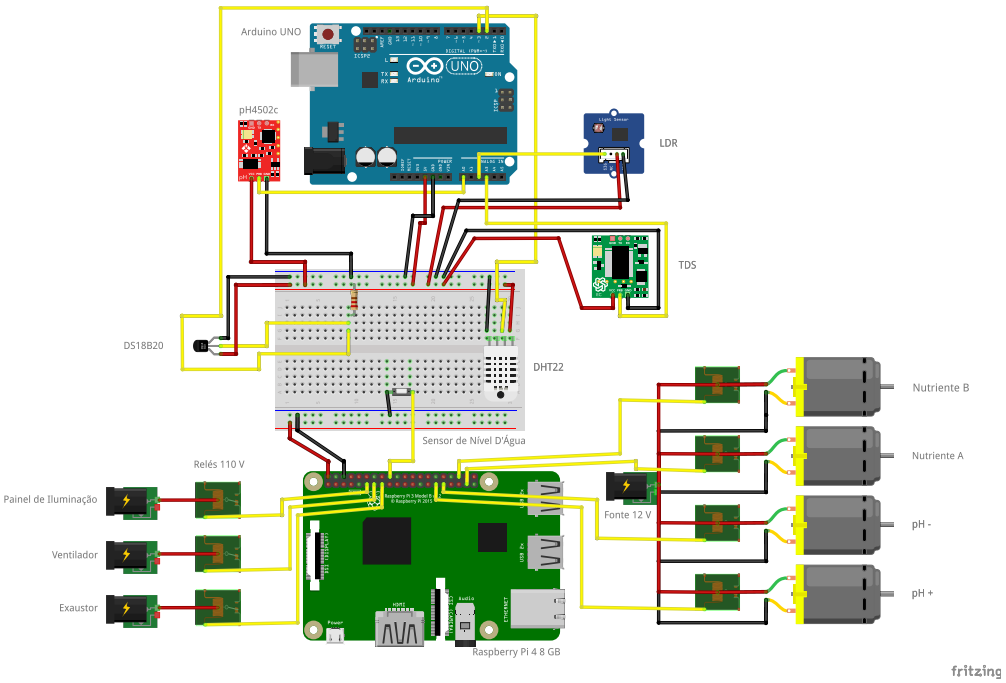

Figura 6 – Diagrama de montagem do circuito eletrônico

Para a montagem e posicionamento do circuito eletrônico dentro da estufa, fixamos os componentes na chapa de MDF com o uso de parafusos e porcas, além de colar a *protoboard* com adesivo auto colante. Utilizando uma cortadora *laser*, realizamos cortes para passagem dos cabos que interligam os módulos *relés* com as tomadas, além do posicionamento das bombas peristálticas, câmera e espelho das tomadas.

Utilizando abraçadeiras de *nylon* e corda plástica, prendemos a chapa na parte traseira da tenda, conforme citado na subseção anterior. Assim, garantimos um certo nível de segurança do circuito em relação a possíveis vazamentos do sistema hidráulico do nosso projeto.

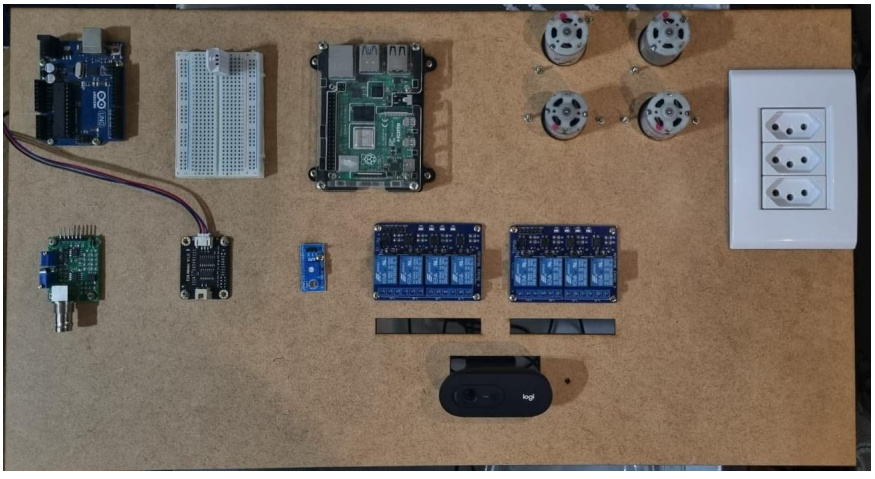

Figura 7 – Montagem do circuito eletrônico na chapa de MDF

5.3.3 Aplicação *Front-end*

O módulo *front-end* do nosso sistema foi desenvolvido em *TypeScript*, utilizando o *framework React*. Seguindo princípios da orientação a objetos e *design* de código (MARTIN, 2005), nosso código está escrito de maneira clara, legível e componentizada, facilitando assim a manutenção e reutilização do código como um todo.

5.3.3.1 Organização do código

Em cada etapa do desenvolvimento do nosso aplicativo *front-end*, o uso de boas práticas foi essencial para garantir maior eficiência do nosso trabalho. Além de envolver conceitos como componentização e legibilidade, já citados anteriormente, outro ponto fundamental foi a organização correta das pastas. Garantindo certas responsabilidades para cada diretório, torna-se muito mais fácil a colaboração da equipe no mesmo código. No nosso projeto, adotamos a seguinte organização:

- *components*: Pasta utilizada para implementar componentes que serão reutilizados em diferentes partes do código.
- *contexts*: Pasta utilizada para definir os *contexts* presentes no nosso programa, funcionalidade nativa do *React* que possibilita compartilhar dados entre diversos componentes sem depender da passagem de *props* pela árvore de componentes do aplicativo.
- *hooks*: Pasta utilizada para implementar os *custom hooks*, funções responsáveis pela comunicação com a aplicação *back-end* do nosso projeto.
- *pages*: Pasta utilizada para definir as páginas do nosso aplicativo. Neste caso, vale ressaltar que componentes que são utilizados apenas em uma página estão declarados dentro da própria pasta específica da página em questão, em um subdiretório chamado *_compose*.
- *routes*: Pasta utilizada para definir, com auxílio da biblioteca *react-router-dom*, as rotas de cada página do aplicativo.
- *types*: Pasta utilizada para definir as principais interfaces utilizados pelo projeto. Por utilizarmos o *TypeScript* para desenvolver nosso código, esta pasta possui grande importância para garantir a tipagem correta das funções e variáveis utilizadas em todo o projeto.
- *utils*: Pasta utilizada para definir funções e constantes que são reutilizadas em diferentes partes do código.

5.3.3.2 Telas do aplicativo *web*

Buscamos, ao longo do desenvolvimento do aplicativo, elaborar um fluxo com interfaces simples e intuitivas para o usuário, dispensando assim a necessidade de manuais complexos ensinando como utilizar o sistema. Além disso, uma das nossas maiores preocupações foi garantir a responsividade de cada tela, de forma a não comprometer a navegação, independente do dispositivo utilizado para acesso ao *site*.

Rota da página	**Título da página**
/login	Autenticação do usuário
/dashboard	*Dashboard* do sistema
/cam/now	Visualização da estufa ao vivo
/system	Informações sobre a estufa
/change-password	Alteração de senha
/profile	Listagem de perfis de cultivo cadastrados
/profile/details	Detalhes do perfil de cultivo
/profile/add	Criação de novo perfil de cultivo
/profile/edit	Edição de perfil de cultivo

Tabela 4 – Telas desenvolvidas no aplicativo *web*

- **Autenticação do usuário:** A tela inicial do aplicativo possui um formulário que solicita o nome de usuário e senha, campos necessários para a autenticação no sistema. Após ser autenticado com êxito, o usuário é direcionado para a tela de *dashboard*.

  Nesta etapa, o usuário recebe um *token* de autenticação que é armazenado localmente e utilizado em todas as requisições seguintes para o servidor *back-end*.

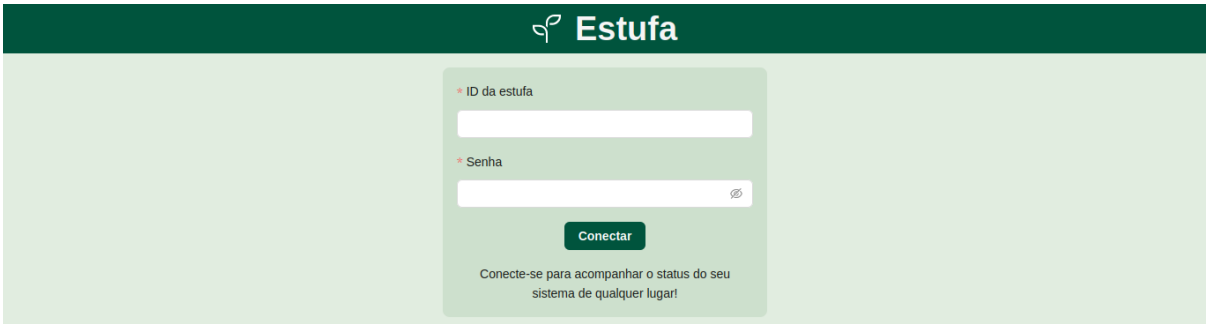

Figura 8 – Captura de tela da página de autenticação do usuário

- ***Dashboard* do sistema:** Nesta tela, considerada a página inicial do fluxo, o usuário consegue visualizar o estado atual das variáveis controladas pelo sistema, incluindo temperatura do ar e da água, umidade, pH, condutividade e luminosidade, podendo ainda ajustar individualmente a faixa desejada para cada parâmetro. Além disso, o usuário pode utilizar os botões de ação, localizados abaixo do cabeçalho da página,

para recarregar os dados do *dashboard*, visualizar a estufa no instante atual ou baixar um arquivo de vídeo *.mp4* mostrando um *time-lapse* do cultivo atual.

A partir desta tela, o cabeçalho da página (componentizado para ser reutilizado em todas as páginas nas quais o usuário já se autenticou) possui três ações:

- Seta para esquerda (canto esquerdo): Voltar para a página anterior;
- Logotipo do aplicativo (centro): Voltar para a página anterior;
- Ícone de usuário (canto direito): Ir para página de informações sobre a estufa.

Conforme pode ser observado abaixo, a criação de um componente genérico que exibe a faixa desejada de um parâmetro tornou a implementação dessa tela mais simples e legível.

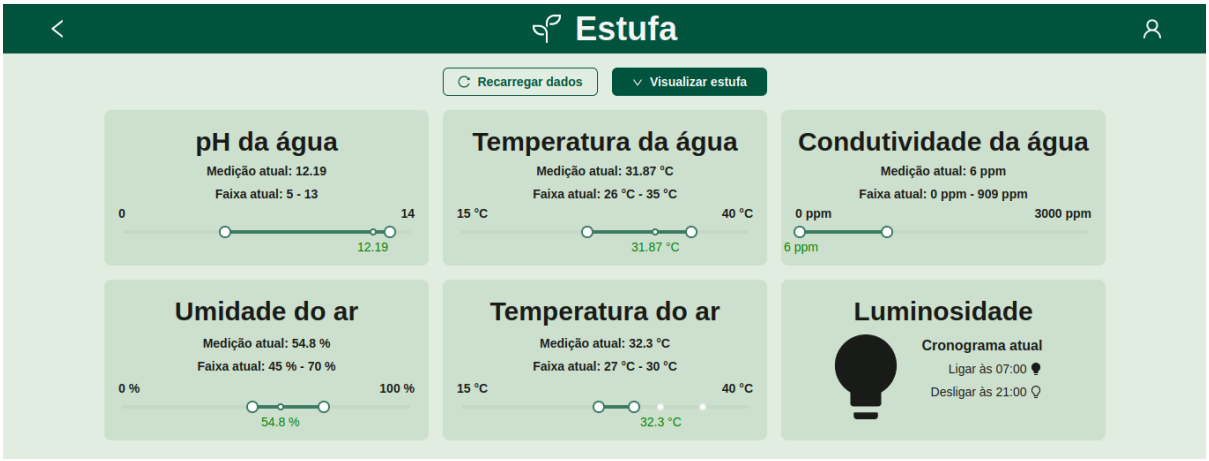

Figura 9 – Captura de tela da página de *dashboard* do sistema

- **Visualização da estufa ao vivo:** O sistema captura uma imagem da estufa, utilizando a câmera conectada ao *Raspberry Pi*, assim que o usuário entra nesta tela, enviando para o usuário um arquivo no formato *.jpeg*. A partir disso, é possível visualizar a foto no aplicativo *web* ou, se preferir, fazer o *download* através do botão indicado. Ao baixar a foto, o arquivo vem nomeado como '*DD-MM-AAAA-hh-mm.jpeg*', de forma a facilitar a organização cronológica por parte do usuário.
- **Informações sobre a estufa:** Nesta tela, o usuário consegue visualizar informações sobre o sistema, incluindo o nome de usuário, o perfil de cultivo selecionado e quantos dias o sistema está operando no respectivo perfil. Além disso, nesta tela o usuário pode navegar para a tela de alteração de senha do sistema ou para as telas de criação e de visualização de perfis.

  Caso o usuário opte por se desconectar do sistema, o *token* obtido durante a etapa de *login* é excluído, e a navegação é redirecionada para a tela de autenticação.

Figura 10 – Captura de tela da página de informações sobre a estufa

- **Alteração de senha:** Nesta tela, o usuário pode realizar a alteração de senha para acesso ao sistema, caso considere necessário. Para isso, é necessário digitar a senha atual e a nova senha, além de redigitar a senha para validação antes de enviar a solicitação para o servidor *back-end*. Se a alteração é realizada com sucesso, o usuário é redirecionado para o *dashboard*. Caso contrário, uma mensagem de erro é exibida e o usuário é mantido na tela atual para realização de uma nova tentativa de alteração da senha.

Figura 11 – Captura de tela da página de alteração de senha

- **Listagem de perfis de cultivo cadastrados:** Nesta tela, o usuário pode visualizar todos os perfis que estão cadastrados no sistema, de forma a poder selecionar, editar, consultar detalhes ou excluir algum perfil através dos respectivos botões de ação disponíveis no *card* de cada perfil. Além disso, o usuário ainda tem a possibilidade de recarregar a listagem dos perfis, criar um novo perfil ou filtrar os perfis existentes pelo nome.

Figura 12 – Captura de tela da página de listagem de perfis de cultivo cadastrados

- **Detalhes do perfil de cultivo:** Através desta tela, o usuário pode visualizar todas as configurações de parâmetros definidas para cada semana de um perfil específico. Além disso, o usuário consegue selecionar ou editar o perfil através dos botões de ação disponíveis no final do *card*.

  Com a criação de um componente genérico que detalha todos os dados de uma semana do perfil, a implementação dos detalhes referentes às N semanas foi mais simples de ser feita. Para fins de ilustração, a captura de tela desta página exibe apenas os detalhes da primeira semana.

Figura 13 – Captura de tela da página de detalhes do perfil de cultivo 'Perfil A'

- **Criação de novo perfil de cultivo:** Nesta tela, o usuário pode criar um novo perfil de cultivo do sistema, definindo o nome e a duração (em semanas) do perfil, e programando individualmente cada um dos parâmetros para cada semana do perfil, incluindo o cronograma de quando ligar/desligar a luz, a faixa de valores para pH, condutividade, temperatura do ar, temperatura da água e umidade, além da proporção entre os nutrientes A e B.

Após definir todos os parâmetros desejados para este novo perfil, o mesmo componente utilizado na tela de detalhes de um perfil foi reutilizado aqui para mostrar ao usuário todas as informações inseridas, permitindo assim uma verificação de todos os dados antes de confirmar a criação do perfil.

Após a criação ser finalizada, o usuário é redirecionado para a tela de listagem de perfis, que fará uma nova consulta no *endpoint* para buscar todos os perfis existentes, incluindo o que acabou de ser adicionado.

Figura 14 – Captura de tela do *input* de informações gerais no cadastro de um novo perfil

Figura 15 – Captura de tela do *input* de cronograma de luz no cadastro de um novo perfil

Os *inputs* que definem a faixa desejada para os parâmetros de cultivo do sistema foram feitos através da implementação de um componente genérico, semelhante ao que foi visto anteriormente na tela de *dashboard*. Para cada parâmetro instanciado (em ordem: pH, condutividade elétrica, temperatura d'água, temperatura do ar e umidade), o componente renderiza um *slider*, componente da biblioteca *Antd* utilizado para selecionar uma faixa de valores.

Figura 16 – Captura de tela do *input* de faixas desejadas para o pH no cadastro de um novo perfil

Figura 17 – Captura de tela do *input* de proporção dos nutrientes no cadastro de um novo perfil

- **Edição de perfil de cultivo:** Nesta tela, o usuário pode editar todos os parâmetros configurados previamente na tela de criação do perfil. Para isso, o usuário visualiza o mesmo fluxo visto na tela anterior, reaproveitando os componentes já mostrados. Entretanto, os valores atuais do perfil são utilizados como valores iniciais de forma a facilitar a jornada de editar e manter os parâmetros conforme necessidade.

5.3.4 Aplicação *Back-end*

A API foi desenvolvida utilizando o *microframework Flask* em *Python*, e através da utilização de um servidor GSWI, com *Green Unicorn* foi implementado e disponibilizado atráves do *Raspberry Pi*. A utilização do servidor foi fundamental para garantir a disponibilidade do servidor, uma vez que atráves dele pudemos definir a utilização de mais de uma instância, garantindo assim a melhor disponibilidade dos dados.

O servidor *back-end* é apoiado por um sistema de gerenciamento de dados local utilizando o *SQLite*. Esta aplicação, por sua simplicidade, permite o armazenamento permanente dos dados sem comprometer o desempenho da aplicação que, por ser executada em uma placa simples, não possui grandes capacidades de processamento e fluxo de dados.

Por fim, vale ressaltar que, para fins de segurança, a API não é exposta publicamente, sendo apenas consumida de maneira local pelo aplicativo *front-end*, que também é mantido no *Raspberry Pi*. Com esse isolamento, juntamente à autenticação via *tokens* JWT, conseguimos garantir a segurança da API, que só pode ser consumida através de uma aplicação uma vez que o usuário esteja devidamente autenticado. É fundamental a segurança da API, uma vez que sua utilização de forma maliciosa pode não só destruir plantios em andamento como também resultar na invasão de privacidade através da câmera interna da estufa.

A divisão do sistema *back-end* pode ser pensada em duas grandes partes: a API que é consumida pela aplicação *front-end* e pode, através deste, ser consumida pelo usuário, e pelos *jobs*, códigos executados periódicamente a fim de efetuar a manutenção da estufa, bem como seu acompanhamento e atualização de parâmetros de cultivo. Esta parte do código, por sua vez, não possui interação direta com o usuário.

5.3.4.1 *Endpoints*

- ***/actuator (GET):*** Retorna o estado de todos os atuadores na estufa.
- ***/actuator/<value>/<action> (POST):*** Permite ligar ou desligar um atuador específico na estufa. O parâmetro *value* especifica o atuador e o parâmetro *action* deve ser "*on*"para ligar ou "*off*"para desligar o atuador.
- ***/actuator/<value>/on_for/<seconds> (POST):*** Permite ligar um atuador por um número específico de segundos. O parâmetro *value* especifica o atuador e o parâmetro *seconds* define a duração em segundos para ligar o atuador.
- ***/sensor (GET):*** Retorna todos os valores medidos pelos sensores na estufa, incluindo temperatura do ar, umidade, temperatura da água, pH, condutividade, nível da água e intensidade de luz.
- ***/sensor/<sensor> (GET):*** Retorna o valor medido por um sensor específico. O parâmetro sensor especifica o sensor desejado, como "*air-temperature*", "*humidity*", "*conductivity*", "ph", "*light*", "*water-level*"ou "*water-temperature*".
- ***/limit (GET):*** Retorna os limites configurados para diferentes variáveis ambientais, como temperatura do ar, umidade, pH, condutividade, etc.

- ***/limit/<value> (GET e PUT):*** Permite obter ou atualizar o limite para uma variável ambiental específica. O parâmetro *value* especifica a variável ambiental desejada, como "*air-temperature*", "*humidity*", "*conductivity*", "ph"ou "*water-temperature*".
- ***/light/schedule (GET e POST):*** Permite visualizar os horários programados para o controle de luz na estufa (liga/desliga). Você pode adicionar novos horários de programação através do método *POST*.
- ***/light/schedule/<id> (PUT e DELETE):*** Permite atualizar ou excluir um horário de programação de luz específico com base no ID do horário.
- ***/nutrient/proportion (GET e PUT):*** Retorna ou atualiza a proporção de nutrientes usada no sistema.
- ***/cam/<action> (GET):*** Permite capturar uma foto da estufa (*action=photo*) ou um vídeo em *time-lapse* (*action=timelapse*).
- ***/change-password (POST):*** Permite ao usuário alterar sua senha.
- ***/login (POST):*** Permite ao usuário fazer *login* no sistema.
- ***/logout (POST):*** Permite ao usuário fazer *logout* do sistema.
- ***/profile (GET e POST):*** Permite visualizar todos os perfis de cultivo disponíveis na estufa (GET) e adicionar um novo perfil de cultivo (POST).
- ***/profile/<id> (GET, PUT e DELETE):*** Permite visualizar, atualizar ou excluir um perfil de cultivo específico com base no ID do perfil.
- ***/profile/current (GET e POST):*** Retorna ou define o perfil de cultivo atualmente em uso na estufa. O perfil atual afeta o comportamento dos atuadores e sensores.

5.3.4.2 *Jobs*

Os *jobs* fazem uso da biblioteca *APScheduler* para agendar tarefas periódicas, que são executadas a intervalos regulares no contexto de um aplicativo *Flask*:

- ***maintain_greenhouse:***
  - Este é um *job* agendado para ser executado a cada *MONITORING_INTERVAL* segundos (definido como 1800 segundos, ou seja, a cada 30 minutos).
  - A função *maintain_greenhouse* executa uma série de operações relacionadas à manutenção da estufa. Ela monitora vários sensores, como temperatura, umidade, condutividade da água, pH, nível da água e luz.

  - Com base nas leituras dos sensores, a função toma decisões e realiza ações, como ligar ou desligar atuadores (como bombas e ventiladores) para manter as condições da estufa dentro dos limites desejados.
  - Além disso, a função também envia notificações se as condições estiverem fora dos limites especificados.
  - Por fim, a função registra a data e hora da última execução no banco de dados.

- ***daily_update:***
  - Este é um *job* agendado para ser executado a cada *UPDATING_INTERVAL* segundos (definido como 86400 segundos, ou seja, a cada 24 horas).
  - A função *daily_update* é responsável por realizar atualizações diárias na estufa. Isso inclui capturar uma foto da estufa usando uma *webcam*, atualizar os registros de perfil de cultivo e atualizar os limites com base no perfil de cultivo atual.
  - A função também realiza a contagem de dias e, se a planta já atingiu o fim do ciclo de crescimento, envia notificações e encerra o ciclo de plantio.
  - Além disso, a função registra a data e hora da última execução no banco de dados.

Esses *jobs* são essenciais para a automação e monitoramento contínuo da estufa, garantindo que as condições ambientais sejam mantidas dentro dos limites desejados e que a estufa seja gerenciada de acordo com o perfil de cultivo atual. Através do agendamento periódico, o sistema automatiza muitas das tarefas de manutenção e ajuste necessárias para o cultivo de plantas na estufa. Além disso, é importante notar que a preservação dos horários de execução é fundamental para reforçar a tolerância a falhas do sistema, uma vez que ao ser desligado por qualquer motivo, no momento em que é reiniciado é possível retomar suas atividades, mantendo os cronogramas o mais próximos do ideal.

5.3.4.3 Banco de Dados

Para a aplicação, é fundamental a persistência de dados de maneira leve e estruturada. Desta forma, o banco de dados *SQLite* foi utilizado pois oferece uma solução eficiente e escalável. Sua modelagem foi feita de modo que permita a adaptação a diferentes tipos de plantas e condições de cultivo.

O banco de dados é composto por sete tabelas:

- ***limit_value:*** Armazena valores limites para parâmetros como temperatura do ar, umidade, pH, condutividade elétrica e temperatura da água.

- ***nutrient_proportion:*** Registra as proporções de diferentes nutrientes a serem aplicados no cultivo.
- ***profile:*** Contém perfis de cultivo que englobam faixas ideais para os parâmetros ambientais ao longo do tempo, incluindo também informações sobre programação de luz e proporções de nutrientes.
- ***current_profile:*** Mantém informações sobre o perfil de cultivo atualmente em execução, incluindo o número de dias passados e o status de conclusão.
- ***light_schedule:*** Armazena horários de iluminação para cada perfil de cultivo.
- **user:** Registra informações de usuários, como nome de usuário e senha.
- ***job_runtime:*** Contém informações sobre o tempo de execução de diferentes tarefas, como monitoramento e atualização do sistema, para que o sistema recupere seu fluxo de manutenção de forma eficiente em caso de queda de energia, por exemplo.

Por fim, o *script* conta com perfis de cultivo pré-definidos, a fim de facilitar a inicialização do sistema.

5.3.4.4 *Ntfy*

O Ntfy é uma ferramenta de notificação e gerenciamento de notificações que permite que um servidor envie informações, alertas e mensagens para várias fontes, como dispositivos móveis, *sites* e outros sistemas. Utilizamos esta ferramenta para garantir a comunicação eficiente com o usuário em diversos canais com o nível de detalhes e urgência customizaveis. Para implementar um servidor de notificações, utilizamos o próprio *Raspberry Pi* e o integramos à API *Flask*. Esse servidor pode receber, armazenar e gerenciar notificações, associando-as à identificadores exclusivos e detalhes específicos. Os dispositivos ou sistemas que desejam enviar notificações podem fazer solicitações HTTP à API, incluindo os dados da notificação. Por outro lado, os clientes que consomem notificações, como aplicativos móveis ou *sites*, precisam autenticar-se na API e buscar notificações pendentes em intervalos regulares. Além disso, o servidor de notificações pode oferecer suporte a diversos canais de entrega para atender às preferências dos usuários, como notificações *push*, e-mails e mensagens de texto. Esse sistema de notificação multicanal torna-se uma ferramenta eficaz para disseminar informações e alertas.

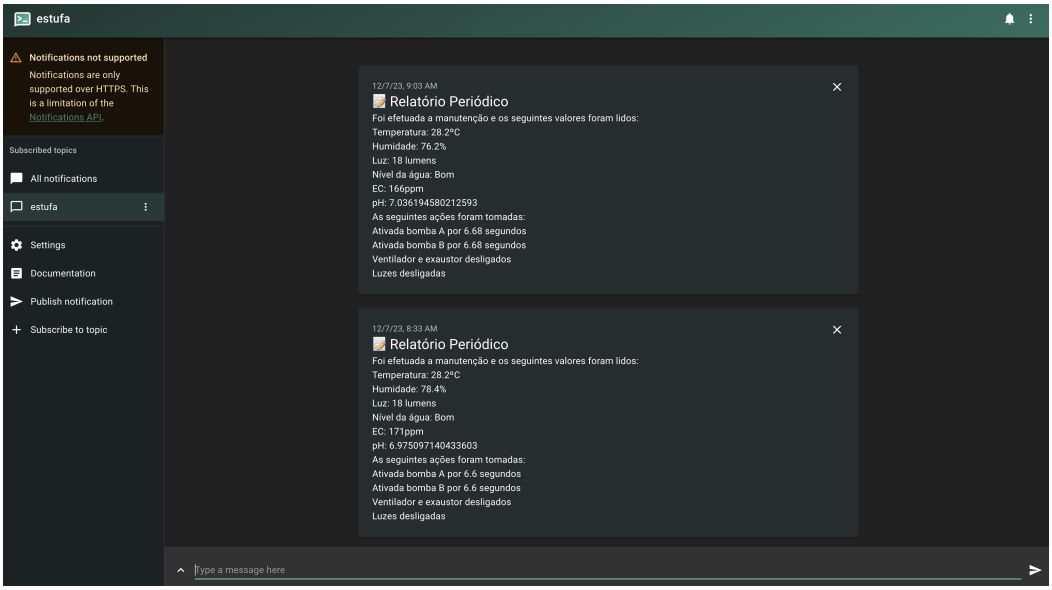

Figura 18 – Captura de tela do *Ntfy*

5.3.4.5 Infraestrutura

A infraestrutura foi exclusivamente implementada no *Raspberry Pi*. A ideia por trás dessa decisão é unificar o sistema, já que dessa forma o produto é visto e produzido como um todo, e uma vez adquirido não é necessário pagar taxas de manutenção para serviços de *cloud*. Sendo assim, dentro do *Raspberry Pi* temos três servidores em funcionamento: O GWSI para a aplicação *back-end*, o *Node.js* para a aplicação *front-end* e o servidor de notificações do Ntfy. Esses servidores estão fechados no ecossistema da aplicação, e a única exposição é feita para o sistema *front-end*, onde o usuário pode visualizar e alterar dados. Para garantir o acesso remoto, foi exposta a porta do servidor no roteador, e além disso, utilizamos o serviço No-IP para utilização de um *Dynamic Domain Naming Service* (DDNS). Desta forma, através de um *link* legível e facilmente compreensível para humanos, é possivel acessar o sistema de qualquer lugar e receber as notificações no dispositivo que melhor atender às necessidades do usuário, lembrando que tanto o *Raspberry Pi* quanto cada um dos servidores descritos tem acesso protegido e seu consumo limitado a acessos específicos a fim de garantir a segurança dos dados e privacidade do usuário.

5.3.5 Integração com Assistente Virtual

A fim de melhorar a interface humano-computador e aumentar as possibilidades de conexão e interação com a estufa, foi desenvolvida uma *skill* para a assistente virtual da *Amazon*, a *Alexa*. Quando falamos do desenvolvimento de aplicações para assistentes virtuais como a *Alexa* ou similares, devemos ter em mente um paradigma específico: estes assistentes não possuem inteligência artificial generativa, portanto, não são capazes de desenvolver uma conversa natural de forma autônoma com o usuário. Sendo assim, fica a

cargo do desenvolvedor arquitetar a aplicação de modo flúido e intuitivo para o usuário. Para isso, utilizamos alguns conceitos que serão descritos a seguir juntamente com seus respectivos processos de desenvolvimento.

5.3.5.1 Invocação

O primeiro passo na criação de uma *skill* é a definição de um nome, que deve possuir no mínimo duas palavras, para ser utilizado para invocação da aplicação desenvolvida. Em nosso caso utilizamos 'estufa inteligente'. Este nome deve ser pensado de modo a se encaixar em diversos cenários de utilização da *skill* como o simples cenário de inicialização: "Alexa, abra estufa inteligente.". Neste exemplo, temos '*Alexa*' como a palavra de ativação da assistente, em seguida, 'abra', palavra que é associada a iniciar alguma aplicação ou interação com dispositivo *smart* e, por fim, o nome de invocação de nossa *skill*. É importante notar que para uma utilização fluída e eficaz o nome deve ser adequado para utilização direta com os *intents* que serão abordados em seguida.

5.3.5.2 *Intents*

Os *intents*, ou simplesmente 'intenções', são frases de ativação para funções específicas de nossa aplicação, para cada intenção podemos configurar diversas frases que se refiram a ela de modo a flexibilizar a utilização da mesma. Em nossa *skill*, desenvolvemos os seguintes *intents*, além dos essenciais:

- **Status:** Se responsabiliza por trazer para o usuário informações medidas pelos sensores. É possível invocá-la falando 'Alexa, qual o status da estufa inteligente?' ou 'Alexa, como está a estufa?', por exemplo. Ao citar 'estufa inteligente', a assistente já irá utilizar a *skill* diretamente. Caso contrário, é necessário primeiro abri-la e, em seguida, perguntar sem especificar o contexto que já foi explicitado com a abertura da aplicação. Este *intent* consulta o *endpoint /sensor* e traduz os dados retornados em JSON pela API para uma linguagem natural.
- **PerfilAtual:** Faz uma consulta no *endpoint /profile/current* e traz, em linguagem natural, informações sobre o cultivo atual, bem como o andamento do mesmo. Pode ser chamado com 'Qual o perfil atual?' ou 'Qual é este cultivo?', por exemplo.
- **Limites:** Se responsabiliza por consultar o *endpoint /limit* e informa ao usuário os limites atualmente definidos para a estufa. Pode ser chamado atráves das perguntas 'Quais são os limites atuais?', 'Quais são os limites?' ou 'Quais são os parâmetros atuais?', por exemplo.
- **HorarioLuz:** Consulta o *endpoint /limit* e informa ao usuário os limites atualmente definidos para a estufa. Pode ser chamado com 'Quais os horários das luzes?' ou 'Qual o cronograma das luzes?', por exemplo.

5.3.5.3 Código

O código da *skill* foi desenvolvido em *Python*. Para cada *intent*, é feita uma classe com uma função *handle*, que lida com a intenção invocada. Em nosso projeto, todo o processamento e regras de negócio foram alocados para o servidor *back-end*. Dessa forma, para a integração com a assistente, foi possível focar na fluída e flexível interação com o usuário, uma vez que os *intents* se resumem a efetuar chamadas HTTP para o servidor dentro do *Raspberry Pi* e converter os dados retornados em JSON para um texto amigável ao usuário que será falado pela *Alexa*.

5.3.5.4 Casos comuns

As aplicações para *Alexa* possuem códigos para alguns *intents* padrões e essenciais para funcionamento de uma aplicação. São elas: *Help, Welcome, Fallback* e *ErrorHandler*. Para estas, apenas deixamos o texto em português, mantendo as características amigáveis da assistente. *Help* fornece instruções de uso da *skill*. *Welcome* é executado ao abrir a aplicação, e apenas fornece uma breve descrição de uma linha para o usuário sobre a *skill*. *Fallback* é utilizado em casos de erro, quando não há ação configurada para o *intent* chamado. Por fim, *ErrorHandler* é utilizado pela *Alexa* quando não foi possível compreender o que o usuário comunicou, e é responsável por pedir novamente as instruções para o usuário por até duas vezes. Caso em alguma delas seja possível processar o comando, o *intent* é executado, caso contrário, é reproduzida uma mensagem de desculpas e encerrada a interação com a *skill*.

5.3.5.5 *Deploy* e Infraestrutura

A infraestrutura que utilizamos é fornecida pelo próprio serviço de *Cloud* da *Amazon*, a *Amazon Web Services* (AWS). Utilizamos o serviço de *Lambda Functions* para executar o processamento da *skill*. Utilizando o ambiente de desenvolvimento da própria *Amazon*, o *deploy* é feito diretamente para o servidor da empresa, e o custo para esse processo é gratuito para até 3 milhões de utilizações mensais, o que é muito mais do que o suficiente para nossa aplicação, uma vez que isso representa 100.000 chamadas por dia, um número muito superior do que de fato o usuário virá a interagir com a *skill* diariamente.

5.4 Testes e Avaliação

Para garantir o pleno funcionamento de cada parte do nosso projeto, dividimos os testes em três grandes etapas:

Na primeira, testamos individualmente cada funcionalidade do sistema, garantindo assim o bom funcionamento dela por si só e evitando falhas mais difíceis de se rastrear quando conectada a outras partes do sistema. Isso inclui:

- Sensores: Cada sensor foi testado individualmente, de forma a garantir o bom funcionamento, evitando falhas futuras devido a componente defeituoso. Nestes testes, inclui-se também o código necessário para leitura, calibração e interpretação do seu dado.

- Atuadores: Cada atuador do sistema foi testado individualmente de forma a garantir seu pleno funcionamento conforme o uso que esperamos dele no sistema. Com isso, podemos garantir que qualquer falha após a integração destes componentes ao sistema não estará sendo causado por ele.

- *Endpoints*: Cada *endpoint* implementado foi previamente testado através do *software Postman*, garantindo assim que a lógica implementada estava de acordo com o esperado, tal como o objeto esperado na requisição e o objeto retornado como resposta. Com isso, a integração na aplicação *front-end* se tornou mais simples.

- Telas (aplicativo *web*): Cada nova tela desenvolvida foi testada tanto em um dispositivo móvel (1080 px x 2340 px) quanto em um monitor comum (1366 px x 768 px), de forma a garantir a responsividade e boa usabilidade, fatores essenciais para garantir o uso do aplicativo *web* em qualquer dispositivo.

Na segunda etapa, iniciamos a integração parcial entre componentes que estão diretamente relacionados no funcionamento do nosso sistema. Nesta etapa, vale salientar que alteramos o parâmetro de tempo para realização de algumas rotinas implementadas, de forma a garantir maior facilidade nos testes. As principais integrações feitas nesta etapa incluem:

- Integração dos *endpoints* em cada tela do aplicativo;

- Integração dos sensores com os *endpoints* que realizam leitura dos parâmetros do cultivo;

- Integração dos atuadores com os *endpoints* que realizam o controle do sistema;

Por fim, a etapa final de testes foi realizada após a montagem completa do sistema, na qual deixamos o sistema operando totalmente integrado e acompanhamos as leituras, os atuadores e os *logs* do sistema para buscar eventuais falhas e oportunidades de melhoria.

5.5 Problemas enfrentados

É importante ressaltar que diversos problemas ocorreram ao longo do desenvolvimento e implementação do nosso trabalho, dado o caráter de prototipação e a sensibilidade do sistema. Com isso, aspectos como a montagem e a qualidade dos sensores geraram

pontos de dificuldade. A fim de reduzir possíveis danos ao resultado final, foi necessário um planejamento minucioso, no qual consideramos todos os aspectos do projeto com cuidado, com margens para garantir a completude dos testes e eventuais ajustes que naturalmente iriam surgir. Portanto, nesta seção iremos abordar os principais problemas que enfrentamos no projeto, aqueles que de fato ameaçaram datas de entrega e qualidade do produto final. São eles:

5.5.1 Vazamento da caixa d'água

O primeiro problema enfrentado foi que a caixa organizadora, utilizada como caixa d'água da estufa, embora aparentemente fosse impermeável, possuía microfissuras invisíveis a olho nú que geravam vazamentos significativos de água, ao ponto que tornaria o armazenamento de líquidos inviável. Na busca por uma solução, focamos em dois pontos: redução de custos e a não contaminação da água. Por fim, optamos por utilizar uma manta impermeabilizadora líquida. Passamos em torno de 4 camadas diluídas em água com pelo menos 4 horas de secagem entre cada uma, a fim de que a manta penetrasse nas microfissuras, além de mais 6 camadas da manta pura, também com um intervalo de 4 horas entre cada camada, para garantir a completa impermeabilização da caixa. A solução se mostrou satisfatória e não houveram mais vazamentos.

5.5.2 Alimentação das bombas peristáticas

Ao montarmos a parte elétrica responsável pelo acionamento das bombas peristáticas, analisamos a necessidade de cada bomba para garantir que a alimentação de tensão e corrente fosse devidamente fornecida. Cada bomba necessita de 350 mA com 12 V de tensão para seu funcionamento, sendo ligada apenas uma por vez na lógica implementada no nosso sistema. Portanto, foi utilizada uma fonte de 12 V 1 A.

O cabeamento que fizemos foi testado parte a parte e como um todo com o auxílio de um multímetro, para garantir que não houvesse mau contato. Todo este desenvolvimento ocorreu sem problemas, entretanto, no teste final as bombas não eram acionadas, e a fonte apresentava *resets* por não suportar a corrente demandada para dar partida em uma bomba. Devido aos valores nominais serem atendidos com margens de segurança, tivemos uma certa dificuldade para diagnosticar a origem do problema, mas ao testar não somente a tensão, mas também a corrente com o multímetro, pudemos identificar o problema na fonte e substituí-la por uma com corrente nominal de 5 A, o que resolveu o problema.

5.5.3 Falha no sensor DHT22

A fim de garantir o funcionamento do sistema embarcado dentro da estufa, passamos por meses de testes e validações extensivas com o sistema montado fora da estrutura final.

Ao longo deste tempo, o sistema rodou 24 horas por dia sendo acompanhado diariamente através dos *logs* e do sistema de notificações. Poucos problemas foram apresentados, em geral, na lógica do servidor *back-end*, nos quais soluções simples foram suficientes. Apesar dos testes e meses de validação que apontavam que nosso sistema estava robusto e funcional, identificamos uma falha do sensor de temperatura do ar e umidade (DHT22) ao montá-lo dentro da estufa: O sistema era ligado e tudo funcionava normalmente por alguns minutos, mas eventualmente o DHT22 cessava o envio das medições, interrompendo assim o fluxo de monitoramento e controle dos parâmetros.

A parte interessante deste cenário foi validar a resposta da estufa sobre um problema com os sensores, na qual éramos notificados e os atuadores eram desligados para evitar atuações que desbalanceassem o ambiente interno da estufa. Entretanto, sem as informações do sensor era inviável manter o cultivo conforme o esperado. Testamos diversos cenários e não encontramos um padrão para o mau funcionamento. Foram testados diferentes *softwares*, montagens e ligações dos cabos e nada aparentava melhorar o comportamento observado.

Após longas pesquisas e diversos experimentos, nossa principal hipótese foi a de que o caráter assíncrono do sensor, juntamente com as bibliotecas desatualizadas, faziam com que a comunicação com o sensor falhasse eventualmente e não voltasse, sem um padrão bem definido. Migramos então o sensor para que se conectasse diretamente ao Arduino e se comunicasse com o *Raspberry Pi* através da comunicação serial, como outros sensores do nosso sistema já faziam. Infelizmente, esta solução não se mostrou eficaz, e o problema seguiu.

Assim, consideramos trocar o sensor por um mais caro e mais confiável, como o BME280, mas uma solução mais simples e sem custo foi encontrada: *resetar* o sensor. Como não havia uma entrada de *reset* própria, estabelecemos uma da seguinte maneira: utilizamos uma porta digital do *Arduino*, que apresenta uma corrente de 80 mA e tensão de 5 V quando ligada, para alimentar o sensor. No momento que o problema descrito anteriormente ocorre, desativamos a saída digital e, após 500 ms, a reativamos e refazemos a leitura. Esta solução se mostrou eficaz e não foram apresentados problemas com o sensor, que seguiu funcionando incessantemente por semanas.

Em suma, diversos problemas de pequeno porte, majoritariamente relacionados à lógica, cabeamento e montagem foram encontrados, mas os principais, tal como suas respectivas soluções, foram descritos acima. Por fim, a montagem pôde ser concluída a tempo, e a estufa se apresentou funcional para tudo aquilo que foi proposta.

6 Considerações Finais

6.1 Conclusões do Projeto de Formatura

O objetivo do nosso projeto foi desenvolver um sistema de cultivo hidropônico com uso de tecnologia, a fim de automatizar todo o acompanhamento necessário para o desenvolvimento de um cultivo. Nosso foco foi manter o pequeno porte, de forma a possibilitar que essa estufa seja construída em apartamentos e casas com pouco espaço para a construção de um jardim.

Em relação à estrutura física da estufa, incluindo neste cenário todo o material necessário para implementação do sistema embarcado, conseguimos atingir o objetivo principal de um sistema bem compacto, onde nossa estrutura completa coube no espaço delimitado pela tenda de cultivo (60 cm x 60 cm x 140 cm), deixando espaço suficiente para o crescimento simultâneo de até 11 plantas de pequeno e médio porte. Vale ressaltar que a estrutura montada para o cultivo hidropônico pode ser adaptada para comportar diferentes necessidades, como plantas maiores ou uma quantidade maior de espaços para cultivo simultâneo.

O desenvolvimento completo do *software*, incluindo as aplicações *back-end* e *front-end* disponíveis na página do projeto no *GitHub* (NUNES; PRADO; SILVA, 2023), conseguiu implementar toda a lógica necessária para controle do sistema de cultivo de forma automatizada, permitindo ao usuário ajustar as faixas desejadas e acompanhar o status dos principais parâmetros de cultivo.

6.2 Perspectivas de Continuidade

Como continuação para o nosso projeto de formatura, enxergamos a oportunidade do uso de técnicas de inteligência artificial, como redes neurais, para identificar estágios de desenvolvimento da planta através das imagens capturadas diariamente, de forma a identificar parâmetros específicos que possam aumentar o desempenho do cultivo em determinada fase.

Além disso, a utilização de sensores mais precisos e atuadores mais potentes pode garantir um controle maior sobre os parâmetros de cultivo. Essa mudança, apesar de aumentar o custo do projeto, pode influenciar diretamente em questões de performance e eficácia, já que com uma precisão maior no controle de cada atuador, uma resposta mais rápida no reajuste das variáveis seria possível.

Ainda em relação aos atuadores, pode-se também adicionar maior complexidade

ao sistema para que seja possível atuar automaticamente em cenários adicionais, como nível d'água abaixo do esperado, temperatura do ar e umidade abaixo da faixa desejada ou temperatura d'água fora do intervalo definido. Atualmente, nosso sistema conta com alertas que são enviados caso haja alguma inconsistência nessas variáveis, mas ainda é necessária a ação direta do usuário para sua correção.

Em relação ao aplicativo *web*, pode-se mapear algumas oportunidades de aprimoramento, que incluem: adicionar filtros mais específicos, além do nome, para se buscar por um perfil de cultivo; requisitos mínimos para alteração de senha, de forma a aumentar a segurança para o usuário; desenvolvimento de um aplicativo *mobile*, tornando a utilização ainda mais cômoda para dispositivos móveis, apesar da responsividade do aplicativo *web* já garantir uma boa experiência ao usuário.

Referências

ARORA, U. et al. Automated dosing system in hydroponics with machine learning. In: IEEE. *2021 International Conference on Communication information and Computing Technology (ICCICT)*. [S.l.], 2021. p. 1–6. Citado na página 11.

LEE, S.; LEE, J. Beneficial bacteria and fungi in hydroponic systems: Types and characteristics of hydroponic food production methods. *Scientia Horticulturae*, Elsevier, v. 195, p. 206–215, 2015. Citado na página 10.

LUKITO, R. B.; LUKITO, C. Development of iot at hydroponic system using raspberry pi. *TELKOMNIKA (Telecommunication Computing Electronics and Control)*, v. 17, n. 2, p. 897–906, 2019. Citado na página 11.

MARTIN, R. C. *The principles of OOD*. 2005. Disponível em: <http://butunclebob.com/ArticleS.UncleBob.PrinciplesOfOod>. Acesso em: 12 nov 2023. Citado na página 28.

NUNES, B.; PRADO, M. A.; SILVA, S. *Estufa-Hidroponica-Automatizada*. 2023. Disponível em: <https://github.com/Estufa-Hidroponica-Automatizada/hidroponic-automation>. Acesso em: 01 dez. 2023. Citado na página 45.

SILVA, M. G. de C. et al. A global overview of hydroponics: nutrient film technique. *Revista Engenharia na Agricultura-REVENG*, v. 29, p. 138–145, 2021. Citado na página 10.

SRINIDHI, H.; SHREENIDHI, H.; VISHNU, G. Smart hydroponics system integrating with iot and machine learning algorithm. In: IEEE. *2020 International Conference on Recent Trends on Electronics, Information, Communication & Technology (RTEICT)*. [S.l.], 2020. p. 261–264. Citado na página 11.