Sistema de monitoramento IoT para cultivos de plantas

Hian de Almeida Santiago

CENTRO DE INFORMÁTICA
UNIVERSIDADE FEDERAL DA PARAÍBA

João Pessoa, 2024

Hian de Almeida Santiago

Sistema de monitoramento IoT para cultivos de plantas

Trabalho técnico apresentado ao curso Engenharia da Computação
do Centro de Informática, da Universidade Federal da Paraíba,
como requisito para a obtenção do grau de Bacharel em Engenharia da Computação

Orientador: Verônica Maria Lima Silva

Novembro de 2024

Catalogação na publicação
Seção de Catalogação e Classificação

S235s Santiago, Hian de Almeida.
Sistema de monitoramento IoT para cultivos de plantas / Hian de Almeida Santiago. - João Pessoa, 2024.
43 f.

Orientação: Verônica Silva.
TCC (Graduação) - UFPB/Informática.

1. Internet das coisas. 2. Jardim inteligente. 3. Esp32. I. Silva, Verônica. II. Título.

UFPB/CI CDU 004:633

Elaborado por Michelle de Kássia Fonseca Barbosa - CRB-738

CENTRO DE INFORMÁTICA
UNIVERSIDADE FEDERAL DA PARAÍBA

Trabalho de Conclusão de Curso de Engenharia da Computaçãointitulado ***Sistema de monitoramento IoT para cultivos de plantas*** de autoria de Hian de Almeida Santiago, aprovada pela banca examinadora constituída pelos seguintes professores:

Prof. Dr. Verônica Maria Lima Silva
Universidade Federal da Paraíba

Prof. Dr. Ewerton Monteiro Salvador
Universidade Federal da Paraíba

Prof. Dr. Raoni Kulesza
Universidade Federal da Paraíba

Coordenador(a) do Departamento CI - Departamento de Sistemas de Computação
Josilene Aires Moreira
CI/UFPB

João Pessoa, 6 de novembro de 2024

Centro de Informática, Universidade Federal da Paraíba
Rua dos Escoteiros, Mangabeira VII, João Pessoa, Paraíba, Brasil CEP: 58058-600
Fone: +55 (83) 3216 7093 / Fax: +55 (83) 3216 7117

AGRADECIMENTOS

Gratidão a meus pais, Francisco e Ivoneide que foram incansáveis em dedicarem seus esforços e tempo para que minha jornada até aqui fosse possível. À minha irmã Ianny, agradeço seu apoio e presença constante.

Falar de gratidão é lembrar da família. Por isso, muito obrigado, a todos meus tios, avós e demais familiares, que de alguma forma participaram da minha jornada.

Durante a jornada, conheci amigos que foram fundamentais para que a caminhada fosse mais leve e gratificante, obrigado pela amizade e por todo o suporte. Por fim, agradecimento especial à José Venâncio por ser o principal incentivador de minha carreira acadêmica.

RESUMO

O cultivo de plantas em ambientes domésticos é bastante praticado como um hobby por seus benefícios à saúde de quem pratica ou pelo apreço à estética agradável ao ambiente que o contato com a natureza proporciona. Embora tenha muitos adeptos, essa prática pode ser dificultada pela falta de tempo em rotinas agitadas ou por ser difícil mensurar, sem ferramentas apropriadas, as condições em que a planta se encontra. O presente trabalho descreve o processo de idealização, desenvolvimento e validação de um sistema IoT de código aberto de cuidados de plantas domésticas. O objetivo principal é facilitar o cultivo por fornecer uma plataforma IoT de fácil uso, com informações do ambiente em que a planta se encontra, como umidade do solo e do ar, luminosidade e temperatura, além de disponibilizar cuidados automatizados para a mesma, como irrigação e adubação. Os testes de validação do sistema mostraram uma automação eficiente e de fácil uso, tendo potencial para incentivar o cultivo.

Palavras-chave: Internet das coisas, jardim inteligente, Esp32, sensores, open source

ABSTRACT

Growing plants in domestic environments is widely practiced as a hobby due to its health benefits for those who practice it, or because of the pleasant aesthetics that contact with nature provides. Although it has many followers, this practice can be hindered by the lack of time in busy routines or because it is difficult to measure the conditions in which the plant is found without tools. This technical report describes the process of idealization, development and validation of the open source system for caring for domestic plants. The main objective is to facilitate cultivation by providing an easy-to-use IoT platform with information about the environment in which a plant is found, such as soil and air humidity, light and temperature, in addition to providing automated care for the plant, such as watering and fertilization. The validation tests of the system showed efficient and easy-to-use automation, with the potential to encourage cultivation.

Key-words: Internet of things, smart garden, Esp32, sensors, open source

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS

IoT – Internet of things

TCP – Transmission Control Protocol

HTTP – Hypertext Transfer Protocol

SGBD - Sistema Gerenciador de Banco de Dados

XML - Extensible Markup Language

SQL - Structured Query Language

HTML - Hyper Text Markup Language

CSS - Cascading Style Sheets

JS - Javascript

JSON - JavaScript Object Notation

API - Application Programming Interface

Sumário

1 INTRODUÇÃO

O ato de cultivar e manter um jardim oferece uma variedade de benefícios para a saúde, tendo efeitos positivos no humor, reduzindo o estresse e contribuindo para melhorias na saúde mental. Essa atividade combina o encontro com a natureza com um exercício leve favorecendo o bem-estar físico[1].

Botelho et al. (2014) [2], em sua pesquisa realizada em três estados brasileiros, revela uma tendência de cultivo doméstico mais acentuada nas áreas interioranas do que nas capitais, indicando que, apesar da intensa urbanização, o hábito de cultivar plantas ainda é amplamente praticado. A pesquisa aponta que a maioria das cidades utiliza predominantemente plantas alimentícias, possivelmente como forma de complementar a dieta, seguidas pelo cultivo de plantas medicinais.

Diante do contexto do cultivo em zona urbana, surge o conceito de agricultura urbana, que se refere ao ato de cultivar em pequenos espaços, sejam eles privados ou coletivos. Essa prática oferece benefícios ecológicos significativos, como a redução do lixo urbano ao aproveitar áreas ociosas para cultivo, a reciclagem de nutrientes por meio da compostagem, a melhoria na gestão da água e o aumento da segurança alimentar [3]. As principais características da agricultura urbana incluem a utilização de pequenas áreas para cultivo, o baixo conhecimento técnico dos produtores e a falta de dedicação exclusiva a essa atividade [4].

Nesse contexto, a tecnologia desempenha um papel fundamental na otimização do cultivo de plantas. A integração com sistemas inteligentes, que utilizam sensores e dispositivos conectados, possibilita o monitoramento em tempo real das condições ambientais, permitindo que os produtores acompanhem as necessidades das plantas e economizem recursos como água e adubação, promovendo assim um desenvolvimento mais sustentável. Além disso, a automação dessas práticas é especialmente benéfica para aqueles que têm pouco tempo para se dedicar ao cultivo.

Desse modo, a ideia do trabalho proposto, como mostra a Figura 1, é criar um ambiente integrado de atenção em que o usuário, por meio da plataforma, acompanha e gerencia ações de cuidado e leituras do ambiente em que a planta se encontra. O microcontrolador ESP32 é responsável pela lógica de transmitir os comandos — sendo eles irrigação, adubação e iluminação — do usuário para os atuadores bem como de gerenciar o salvamento dos dados obtidos pelos sensores, que acompanham a umidade do ar e do solo, o índice de luminosidade do ambiente e o nível de água do reservatório.

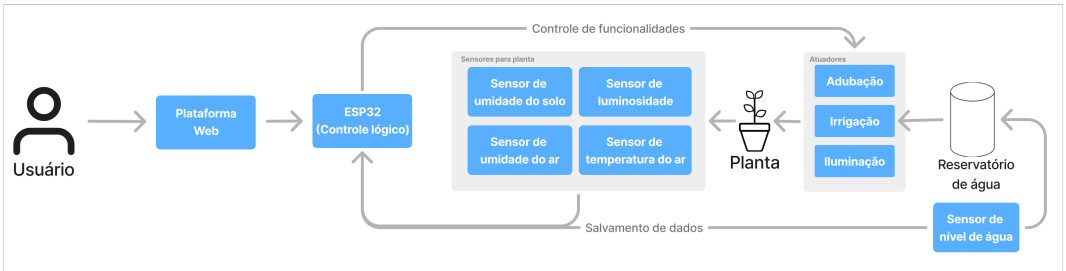


Figura 1: Visão geral do sistema
Fonte: Imagem do autor.

1.1 Premissas e Hipóteses

Levando em conta o contexto da agricultura urbana e a dificuldade de cultivar em pequenos espaços, sem o conhecimento técnico e as ferramentas que facilitem o processo, bem como os desafios enfrentados ao cultivar em ambientes internos, é evidente a necessidade de soluções inovadoras que possam apoiar os habitantes na prática do cultivo. Sendo assim, surge a necessidade de combinar automações para otimizar o uso do espaço e os recursos que facilitem o cultivo para aqueles com pouco tempo ou pouca experiência.

A hipótese central desse desenvolvimento, baseada no contexto de trabalhos de automação de cultivo inteligente, é que um sistema integrado com automação de cuidados e com a integração de sensores para acompanhamento do ambiente traga melhorias significativos para a manutenção de plantas domésticas, evitando o uso desproporcionado de recursos hídricos, diminuindo o tempo gasto com a prática de cultivo e favorecendo uma maior popularização entre aqueles que encontram dificuldades em executar com êxito essa prática.

1.2 Objetivo geral

Esse estudo visa projetar e implementar um sistema integrado de monitoramento e cuidados de plantas domésticas, disponibilizado online para acesso via computador e dispositivos móveis, que por meio de sensores colete dados do ambiente em que a planta se encontra e forneça atuadores automáticos de cuidados, promovendo assim um ambiente favorável ao cultivo saudável de plantas.

1.3 Objetivos específicos

- Desenvolver um protótipo de sistema de monitoramento com sensores de umidade temperatura e luminosidade
- Implementar um protótipo com atuadores que realizem ações de rega, adubação e iluminação automática
- Desenvolver uma interface para monitoramento das leituras dos sensores, definir padrões de cuidados da planta e acompanhar as atuações.
- Desenvolver o sistema de gerenciamento com integração a banco de dados de modo que seja possível ter mais de uma planta acompanhada e controlada na plataforma.
- Disponibilizar o código de todo o projeto em repositório GitHub.

1.4 Estrutura do Trabalho

A estrutura deste trabalho de conclusão de curso tem início no capítulo de introdução com a contextualização do problema abordado e com os principais objetivos dessa pesquisa. O segundo capítulo descreve os principais conceitos, técnicas e ferramentas usadas no desenvolvimento. O terceiro capítulo descreve a metodologia utilizada, os requisitos da aplicação e as estruturas usadas na arquitetura. No quarto capítulo são apresentados os casos de teste abordados. Por fim, o quinto capítulo traz uma visão das conclusões obtidas com esse desenvolvimento bem como uma análise de possíveis melhorias e trabalhos futuros.

2 FUNDAMENTAÇÃO TEÓRICA

Nesse capítulo, serão apresentadas as principais tecnologias escolhidas para a construção dessa aplicação, além de alguns conceitos que envolvem essa pesquisa.

2.1 Conceitos Gerais

Nessa seção serão definidos os principais conceitos utilizados no desenvolvimento da solução proposta.

2.1.1 Internet of Things (IoT)

"Internet das Coisas"(IoT) é um conceito apresentado por Kevin Ashton, um dos fundadores do Laboratório de Auto-ID do Instituto de Tecnologia de Massachusetts, em 1999 [5]. Embora não exista uma definição única, a IoT pode ser sintetizada pela integração de tecnologia incorporada em objetos cotidianos, permitindo que eles coletem dados, por meio de sensores, transmitam-os pela internet e se comuniquem com aplicativos na nuvem. Esta tecnologia está presente em diversos setores, como casas inteligentes, na indústria com automação de processos e até mesmo na área de assistência médica [6].

2.1.2 Websockets

Websockets é um protocolo de comunicação que tem por objetivo fornecer um mecanismo de conexão bidirecional entre o cliente e servidor, para aplicativos web, sem que eles dependam da abertura de várias conexões HTTP para a troca de mensagens [7]. O protocolo consiste em um handshake inicial de abertura da conexão por meio de solicitação HTTP entre o cliente e servidor seguido da troca de mensagens por meio de uma camada de abstração baseada em TCP.

Outros protocolos de comunicação IoT também foram estudados, como exemplo o MQTT (Message Queuing Telemetry Transport), que pode atender também todas as funcionalidades de comunicação necessárias nesse trabalho, porém a escolha do Websockets foi feita pelo conhecimento prévio do desenvolvedor em trabalhar com essa tecnologia e sua facilidade de implementação e integração numa Api Python.

2.1.3 Banco de dados

Date (2004) define banco de dados como um sistema computadorizado de manutenção de registros, que tem por finalidade armazenar informações e que quando solicitado seja permitido ao usuário criar, buscar e atualizar essas informações. Essas funções de

utilização do banco de dados são controladas por um sistema gerenciador de banco de dados (SGBD).

Dentre os modelos de banco de dados mais conhecidos se destacam os bancos de dados relacionais. Nesse tipo de banco os dados são organizados em tabelas inter-relacionadas, com a divisão de dados organizados em colunas semelhantes a planilhas [9]. Em sua maioria usam em sua estrutura a linguagem de consulta Structured Query Language (SQL) para manipular os dados armazenados.

2.1.4 Back-end

O back-end é a parte do servidor do sistema responsável por armazenar, organizar e disponibilizar os dados. Esses dados são criados e acessados pelos usuários por meio da aplicação front-end [10]. O back-end é então responsável pela comunicação entre a interface e o banco de dados, gerenciando suas operações de acesso e assegurando as funcionalidades do sistema.

2.1.5 Front-end

Front-end diz respeito a parte visual que o usuário interage diretamente ao acessar uma aplicação, englobando as cores, designs de texto e elementos gráficos como imagens e tabelas[10].

As linguagens *Hypertext Markup Language* (HTML) e Javascript são comumente usadas para construir a parte visual e trazer lógica para essa estrutura, sendo comum também o uso de *Cascading Style Sheets* (CSS) para a estilização visual.

2.2 Tecnologias utilizadas

Nessa seção serão especificadas as tecnologias utilizadas para o desenvolvimento do sistema.

2.2.1 Desenvolvimento Web

- **JavaScript:**[1] É uma linguagem interpretada, multi-paradigma, baseada em objetos e de tipagem leve. É a linguagem mais utilizada para páginas Web, mas também pode ser usada em ambientes sem browser como o Node.js. Nessa aplicação, foi

[1]Documentação disponível em: https://developer.mozilla.org/pt-BR/docs/Web/JavaScript. Acesso em 17 de Setembro de 2024

utilizado o TypeScript para o desenvolvimento da aplicação web front-end. O TypeScript é uma linguagem de programação que estende o JavaScript, adicionando tipagem aos dados[2].

- **Node.js:**[3] É um ambiente de execução JavaScript, multiplataforma e open-source. O Node.js permite executar javascript fora de um browser, o que o permite ser performático para diversas aplicações. Para o projeto desenvolvido ele será utilizado na construção do front-end.

- **React:**[4] É uma biblioteca JavaScript, criada pela Facebook, para construção de interface de usuário de aplicações web ou aplicações nativas em JS. A construção de interfaces em React é feita baseada em componentes que podem ser reutilizados em diferentes partes da aplicação, promovendo assim reutilização do código e facilidade no desenvolvimento.

- **MaterialUi:**[5] É um biblioteca de componentes React de interface de usuário, código aberto, desenvolvida pela Google. Ela é composta por componentes prontos, além de opções de customização que facilitam o desenvolvimento.

- **Python:**[6] É uma linguagem de programação de código aberto, alto nível, interpretada e multiparadigma. Possui sintaxe simples com ênfase em legibilidade do código. Por ser uma linguagem de código aberto possui uma vasta biblioteca padrão que simplifica e acelera o desenvolvimento.

- **Django Rest Framework:**[7] - É um framework de código aberto para desenvolvimento web server baseado em Python e que utiliza a estrutura model-template-view. Nessa aplicação esse framework foi utilizado para a contrução da API.

2.2.2 Banco de dados

- **PostgreSQL:**[8] É um sistema de banco de dados relacional de código aberto que usa os recursos da linguagem SQL e soma a ela recursos para administrar e gerenciar dados.

[2]Website oficial acessível em: https://www.typescriptlang.org/. Acesso em 17 de Setembro de 2024
[3]Website oficial acessível em: https://nodejs.org/pt. Acesso em 17 de Setembro de 2024
[4]Website oficial acessível em: https://react.dev. Acesso em 17 de Setembro de 2024
[5]Website oficial acessível em: https://mui.com/material-ui/getting-started/. Acesso em 17 de Setembro de 2024
[6]Documentação disponível em: https://www.python.org/about/. Acesso em 17 de Setembro de 2024
[7]Documentação oficial acessível em: https://www.django-rest-framework.org/. Acesso em 17 de Setembro de 2024
[8]Documentação oficial acessível em: https://www.postgresql.org/about/. Acesso em 17 de Setembro de 2024

2.2.3 Microcontrolador

As ferramentas definidas nessa seção descrevem a comunicação entre o módulo microcontrolador e os sensores usados na aplicação.

- **Arduino Programming Language:**[9] é uma adaptação da linguagem de programação C++ de sintaxe simples, open souce e com uma vasta coleção de bibliotecas para facilitar a comunicação com sensores e outros componentes de hardware.
- **PWM:**[10] Modulação por Largura de Pulso é uma técnica usada para obter valores analógicos por meio de pulsos digitais, criando uma onda quadrada entre a tensão de saída do controlador e a tensão zero, permitindo assim uma variação liga-desliga por meio de controle digital. Esse protocolo foi utilizado no controle do motor.
- **DHT Sensor Library:**[11] Biblioteca Arduino construída em C++ para a leitura de dados de sensores de temperatura sendo eles DHT1 e DHT22.

2.2.4 Sistema de controle de versões

O sistema de controle de versões é uma ferramenta de gerenciamento de código que permite o acompanhamento de todas as modificações feitas ao longo do tempo [11]. Ele facilita o trabalho colaborativo entre desenvolvedores, permitindo que trabalhem em diferentes partes do projeto ao mesmo tempo. Conta com funções de correção de conflitos entre mescla de códigos, recuperação de versões anteriores, e rastreabilidade da autoria de cada modificação feita. Para esse trabalho, o sistema usado foi o **Git**[12].

2.3 Trabalhos relacionados

Nesta seção, serão abordados trabalhos semelhantes que oferecem soluções para o mesmo problema. Será realizado um comparativo entre os componentes e softwares utilizados, destacando as similaridades e diferenças em relação ao que foi desenvolvido nesta pesquisa.

[9]Documentação disponível em: https://www.arduino.cc/reference/en/. Acesso em 18 de Setembro de 2024

[10]Documentação disponível em: https://docs.arduino.cc/learn/microcontrollers/analog-output/. Acesso em 18 de Setembro de 2024

[11]Documentação oficial acessível em: https://github.com/adafruit/DHT-sensor-library?tab=readme-ov-file. Acesso em 18 de Setembro de 2024

[12]Documentação oficial acessível em: https://git-scm.com/. Acesso em 19 de Setembro de 2024

2.3.1 IoT and artificial intelligence based smart gardening and irrigation system

Tumpa et al. (2023) propõem o desenvolvimento de um sistema de irrigação inteligente controlado remotamente por um aplicativo móvel. Esse sistema é composto por dois sensores de umidade do solo, o Soil Moisture Sensor YL69 e o V1.2, além de um sensor de chuva, um sensor de umidade e temperatura do ar DHT11, um motor hidráulico, um módulo relé para controle do motor e uma bateria de 9V para alimentação [12].

Para o armazenamento dos dados, foi utilizado o Real Time Database da Firebase, onde são salvos os últimos dados de leitura do ambiente. A irrigação pode ser realizada de forma remota ou automaticamente, caso a umidade do solo seja inferior a um valor pré-determinado[12].

2.3.2 IoT-Enabled Smart Drip Irrigation System Using ESP32

Pereira, Chaari e Daroge (2023) desenvolveram um sistema que combina tecnologia IoT com irrigação por gotejamento, controlado por um microcontrolador ESP32. O sistema utiliza um sensor de umidade do solo (SEN0308 DFRobot), um sensor de temperatura do solo (DS18B20), um sensor de umidade e temperatura do ar (DHT22), um sensor de fluxo de água (FS300A) e uma válvula para controle do fluxo de água, além de um relé e um regulador de tensão.

Esse sistema oferece uma automação complexa da irrigação, baseada na umidade e temperatura, com horários fixos para realizar a irrigação. Contudo, também permite o controle manual do usuário, que pode acionar a válvula para o gotejamento ou interromper uma rega em andamento. Para gerenciar toda essa lógica, utiliza-se a ESP32 conectada à aplicação Blynk[13] , uma plataforma para construção de aplicações IoT. Nessa plataforma, o usuário pode acompanhar as leituras dos sensores e executar as ações mencionadas anteriormente. [13]

2.3.3 A Study on IoT based Real-Time Plants Growth Monitoring for Smart Garden

Song (2020) apresenta uma nova perspectiva sobre os componentes de automação para o cuidado das plantas, incluindo uma hélice para ventilação (Fan Blade Motor) para situações em que o ambiente está muito seco e uma luz LED para casos de alta umidade. Além disso, o sistema proposto incorpora um sensor de gás (MQ135), sensível à fumaça e a gases como amônia e sulfeto, que são prejudiciais ao desenvolvimento das plantas. Por fim, inclui uma bomba de água para irrigação [14].

[13]Website oficial acessível em: https://blynk.io/. Acesso em 18 de Setembro de 2024

O sistema é controlado por valores preestabelecidos das condições ambientais, que determinam quando as ações devem ser executadas. Para o armazenamento dos dados, foi utilizada a Real-time Database do Firebase, além do Android Studio para o acompanhamento das informações por meio de uma aplicação móvel e um smartwatch [14].

2.3.4 IoT-based Smart Gardening System

Choudhari et al. (2023) desenvolve um sistema que se assemelha bastante aos demais descritos anteriormente, incorporando componentes já conhecidos, como o módulo microcontrolador ESP32, o sensor de umidade e temperatura do ar DHT11, a bomba de água e o sensor de umidade do solo. No entanto, este sistema apresenta uma nova funcionalidade de adubação, contando com a assistência de um servo motor e o uso de um novo sensor de luminosidade (LDR)[15].

Para o controle dos componentes e acompanhamento do usuário, utiliza a plataforma Blynk. O sistema oferece a opção de controle manual remoto da adubação e dos demais atuadores, além de um controle automático das regas, que é acionado de acordo com valores pré-estabelecidos de umidade do solo [15] .

2.3.5 Sistema de cultivo inteligente

Anastácio et al. (2023) propõem um sistema de irrigação inteligente, resultado de uma análise de diferentes amostras de solo em variados níveis de umidificação, além de uma pesquisa para entender o perfil de cultivo dos entrevistados, incluindo os tipos de plantas mais cultivadas e seus hábitos. O sistema utiliza um sensor de nível de umidade do solo, um motor para irrigação e um sensor de nível de água para monitoramento do reservatório. Como controlador, emprega um Arduino, juntamente com um módulo Bluetooth para conexão [16].

Além disso, conta com um sistema de controle móvel desenvolvido na plataforma Kodular[14], que permite a conexão ao Arduino via Bluetooth e a personalização da atuação do modelo com opções pré-configuradas para as plantas mais comuns, obtidas através da pesquisa de campo. O diferencial deste projeto é o alto nível de personalização, permitindo que o usuário final adapte o sistema às necessidades específicas da planta que cultiva [16].

2.3.6 Smart Home Gardening Management System: A Cloud-Based Internet-of-Things (IoT) Application in VANET

Sharma et al. (2020) desenvolvem um sistema tendo por base o controlador Arduino conectado a um módulo Wifi e a dois sensores sendo eles o de umidade do solo e

[14]Website oficial acessível em: https://www.kodular.io/. Acesso em 18 de Setembro de 2024

do umidade do ar DHT11. O sistema é então conectado à plataforma ThingSpeak que gera gráficos de monitoramento dos dados obtidos pelos sensores. O sistema conta ainda com um motor de irrigação para atuação, sendo controlado pelo nível crítico de umidade pré estabelecido. O sistema propõe a integração dos dados obtidos num modelo de rede VANET [17].

Por fim, dados os trabalhos analisados, é possível concluir que a principal contribuição do projeto desenvolvido nesse trabalho é a integração de diferentes sensores usados em alguns dos trabalhos mencionados, com uma plataforma que permite não só a personalização dos padrões de cuidados como também uma visualização das leituras dos sensores ao longo do tempo. Além de propor uma plataforma capaz de controlar mais de um dispositivo de cuidado de plantas. podendo assim abrir a possibilidade de escalar o projeto para plantações maiores.

A Figura 2 traz um comparativo geral entre os componentes usados nos trabalhos e as funcionalidade de controle dos sistemas analisados.

	IoT and artificial intelligence based smart gardening and irrigation system [12]	IoT-Enabled Smart Drip Irrigation System Using ESP32 [13]	A Study on IoT based Real-Time Plants Growth Monitoring for Smart Garden	IoT-based Smart Gardening System	Sistema de cultivo inteligente	Smart Home Gardening Management System: A Cloud-Based Internet-of-Things (IoT) Application in VANET	Sistema proposto
Sensores							
Umidade do ar	X	X	X	X		X	X
Umidade do solo	X	X	X	X	X	X	X
Temperatura do ar	X		X	X		X	X
Temperatura do solo		X					
Chuva	X						
Fluxo de água		X					
Gás			X				
Luminosidade				X			X
Nível de água					X		X
Atuadores							
Motor para irrigação	X		X	X	X	X	X
Motor para adubação				X			X
Irrigação por gotejamento		X					
Hélice para ventilação			X				
LED			X				X
Umidificador do ar						X	
Microcontrolador	ESP32	ESP32	ESP8266	ESP8266	Arduino UNO + módulo bluetooth	Arduino UNO + módulo WiFi	ESP32
Controle	Automático e manual. Visualização dos últimos dados de leitura por meio de Aplicativo Android.	Automático e manual. Plataforma Blynk para acompanhamento de gráficos de leitura.	Automático e manual. Aplicativo construído no Android Studio para controle e visualização. Dados salvos no Firebase.	Automático e manual para fertilização. Dados enviados para acompanhamento na plataforma Blynk.	Personalização do controle automático. Aplicação mobile para visualização de dados de mais de um dispositivo de automação.	Automático. Dados enviados para plataforma ThingSpeak para visualização em gráficos	Personalização do controle automático. Controle de mais de um dispositivo. Personalização do intervalo de leituras. Plataforma com histórico de atuações e gráficos de acompanhamento das leituras ao longo do tempo

Figura 2: Tabela comparativa trabalhos relacionados
Fonte: Imagem do autor.

3 METODOLOGIA

O desenvolvimento desse projeto começou com a projeção da arquitetura, partindo das funcionalidades a serem incluídas e os componentes disponíveis. Em seguida, foram levantados os requisitos funcionais para guiar a implementação da plataforma de controle do usuário e as operações que o microcontrolador deve executar. Essa abordagem, que priorizou a arquitetura antes do levantamento de requisitos, ocorreu pelo fato da construção do protótipo ter iniciado antes do planejamento detalhado do roteiro de execução do trabalho.

3.1 Visão Geral

O usuário do sistema terá como principal ferramenta a plataforma disponibilizada para acompanhamento das leituras do ambiente e atuação na planta monitorada, além do protótipo físico no qual a planta deve estar inserida.

Desse modo, a metodologia de desenvolvimento estará dividida em dois grandes blocos funcionais, sendo o primeiro a implementação da estrutura física do projeto, com a descrição da montagem de todos os componentes com o módulo microcontrolador e toda a lógica envolvida nesse processo. Em seguida será abordado o desenvolvimento do conjunto web do sistema, com a construção da interface do usuário, a API de gerenciamento e a estrutura do banco de dados.

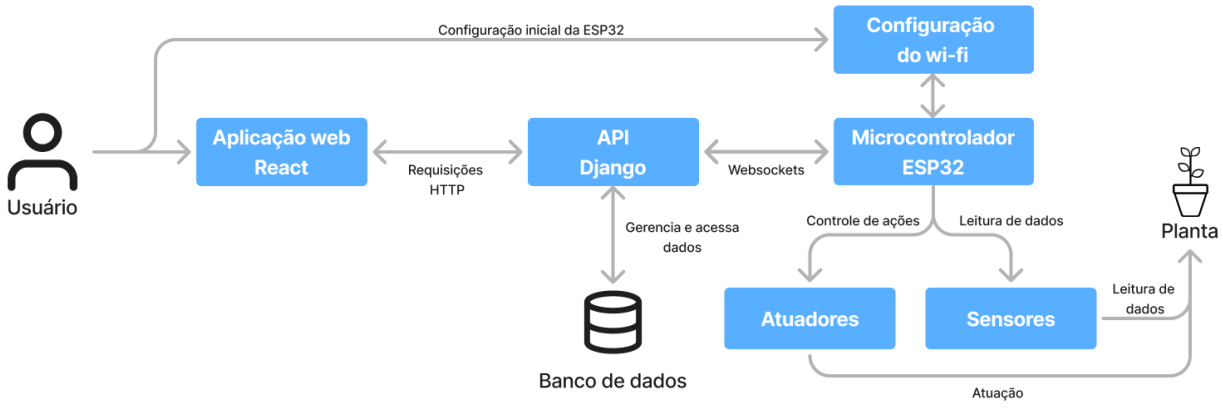


Figura 3: Arquitetura geral do sistema
Fonte: Imagem do autor.

A Figura 3 descreve a arquitetura do sistema com o fluxo completo de informações gerado pela interação do usuário com o sistema. Esse processo inicia com a configuração inicial da ESP32 que precisa dos dados da rede Wi-fi. Após isso, todas as demais interações são feitas por meio da Aplicação Web, desenvolvida em React, que se comunica com a API por meio de requisições HTTP. Já a API, construída em Django, tem conexão direta com o banco de dados fazendo o gerenciamento das informações criadas pelo usuário e

pelo microcontrolador. A API também possui uma conexão, do tipo Websockets, com o microcontrolador ESP32 para envio dos comandos do usuário e também a obtenção dos dados dos sensores. Por fim, o microcontrolador interage com a planta, sendo o último ponto do fluxo de informações, onde os sensores captam os dados do ambiente e os atuadores realizam as ações de cuidados, sendo elas rega, adubação e iluminação.

3.2 Modelo em Hardware

Esse segmento se destina a especificar os requisitos que envolvem a aplicação em Hardware do sistema, descrevendo as funções implementadas no microcontrolador como também as conexões feitas em todos os componentes. Por fim, será apresentado o modelo final desenvolvido.

Tabela 1: Componentes utilizados e suas descrições

Componente	Descrição
ESP-WROOM-32	Microcontrolador
Micro Servo 9g SG90	Servo motor usado no dispenser de fertilizante
Mini Water Pump 12V DC Motor RS-385	Bomba peristáltica usada na função de rega
Reed Switch Water Level Sensor	Sensor de nível de água
UV LED	Luz UV
12v power supply	Fonte usada para alimentação do sistema
XL6009 Step Up Voltage Regulator Module	Módulo regulador de tensão
L298N Double H Bridge	adicionar descrição
DHT22 sensor	Sensor de temperatura e umidade do ar
GC-58 Soil Moisture Sensor	Sensor de umidade do solo
LDR	Sensor de luminosidade
Resitor	1kΩ, 470Ω

Os componentes descritos na Tabela 1 foram utilizados para a construção do protótipo. Além desses componentes também foram utilizados fios e uma protoboard.

3.2.1 Requisitos Funcionais

Descrevem as funcionalidades do sistema a ser desenvolvido, quais os serviços que deve abranger e quais as condições em que cada um deles se encontra. O objetivo desse mapeamento é delimitar e mapear o escopo do trabalho.

- **[RF01] Fornecer página web para conexão com Wi-fi:** Ao iniciar o microcontrolador pela primeira vez deve ser disponibilizada uma página web para que o usuário consiga cadastrar as informações do Wi-fi, uma vez que esses dados podem mudar e não devem ser fixos no código da ESP32.
- **[RF02] Estabelecer conexão com a API:** O microcontrolador deve ser capaz de se conectar a internet com as informações fornecidas no RF001 e conseguir se conectar a API para troca de informações.
- **[RF03] Manter a conectividade após retorno de momentos inativos:** O microcontrolador deve ser capaz de reestabelecer conectividade com a internet e com a API (RF002) quando houver problemas no Wi-fi ou quando o microcontrolador for reiniciado.
- **[RF04] Ler os dados do sensor GC-58:** O microcontrolador deve conseguir ler os dados do sensor de umidade do solo.
- **[RF05] Ler os dados do sensor DHT22:** O microcontrolador deve conseguir ler os dados do sensor de umidade e temperatura do ar.
- **[RF06] Ler os dados do sensor Reed Switch Water Level:** O microcontrolador deve conseguir ler os dados do sensor de nível de água do recipiente de abastecimento.
- **[RF07] Ler os dados do sensor LDR:** O microcontrolador deve conseguir ler os dados do sensor de luminosidade do ambiente.
- **[RF08] Deve executar ação de rega quando acionado:** Ao receber um comando de rega a bomba peristáltica de irrigação deve ser acionada por apenas um período de tempo e depois deve retornar ao estado inativo.
- **[RF09] Deve executar ação de adubação quando acionado:** Ao receber um comando de adubação o motor de adubação deve ser acionado por apenas um período de tempo e depois deve retornar ao estado inativo.
- **[RF10] Executar ação de iluminação:** Ao receber um comando de acionamento da luz o LED deve ser ligado até que o tempo determinado finalize.

- **[RF11] Deve receber da API os parâmetros para controle:** O microcontrolador deve receber da API configurações do usuário com os parâmetros do ambiente necessários para cada ação, sendo elas o tempo de iluminação diária, nível de umidade aceitável e crítico para as regas automáticas e intervalo de tempo para adubações.
- **[RF12] Deve executar as leituras no intervalo estabelecido:** Dadas as condições descritas no RF11 deve apenas executar as leituras no intervalo estabelecido.
- **[RF13] Deve executar as ações no intervalo estabelecido** Dado o mesmo contexto do RF11 o microcontrolador deve apenas executar ações com os atuadores no tempo estabelecido.
- **[RF14] Executar ações apenas nas condições estabelecidas** O microcontrolador deve apenas executar ações com os atuadores se as condições de ambiente estiverem de acordo com as fornecidas pelo usuário, mesmo que esteja no intervalo de tempo (RF012) de execução de atuações.
- **[RF15] Enviar para a API a leitura dos sensores** No intervalo estabelecido (RF011) o microcontrolador deve enviar a leitura dos sensores para a API.

3.3 Protótipo físico

A arquitetura do projeto desenvolvido pode ser vista na Figura 4, onde o diagrama esquemático de todas as conexões foi apresentado. Como entrada do sistema temos uma fonte de 12V, que é convertida pelo regulador de tensão para 5V para que seja possível alimentar a ESP32 e todos os demais componentes. A ponte H (L298 Double H Bridge) foi utilizada para controlar a bomba peristáltica que também é alimentada por 12V.

Na Figura 5 é possível ver a maioria dos componentes usados na construção física do projeto. Ao centro da imagem temos a ESP32 conectada aos demais elementos na protoboard, é possível também ver o sensor DHT22, sensor de luminosidade LDR, e também a bomba peristáltica acoplada ao reservatório de água. Os demais fios conectados à protoboard e à ESP32 são usados para a conexão com o Servo Motor de adubação e o sensor de leitura do solo, como é possível ver na Figura 6.

A Figura 9 mostra em mais detalhes como o Servo Motor foi utilizado para criar o dispenser de fertilizante sólido. Já a Figura 10 traz em detalhes como o sensor de nível de água foi conectado ao reservatório de água do sistema.

A ESP32 opera em dois modos de conectividade, alternando entre ponto de acesso e cliente, dependendo da conectividade com a internet. No modo de ponto de acesso, o microcontrolador permite que o usuário se conecte a um endpoint fixo, acesse uma

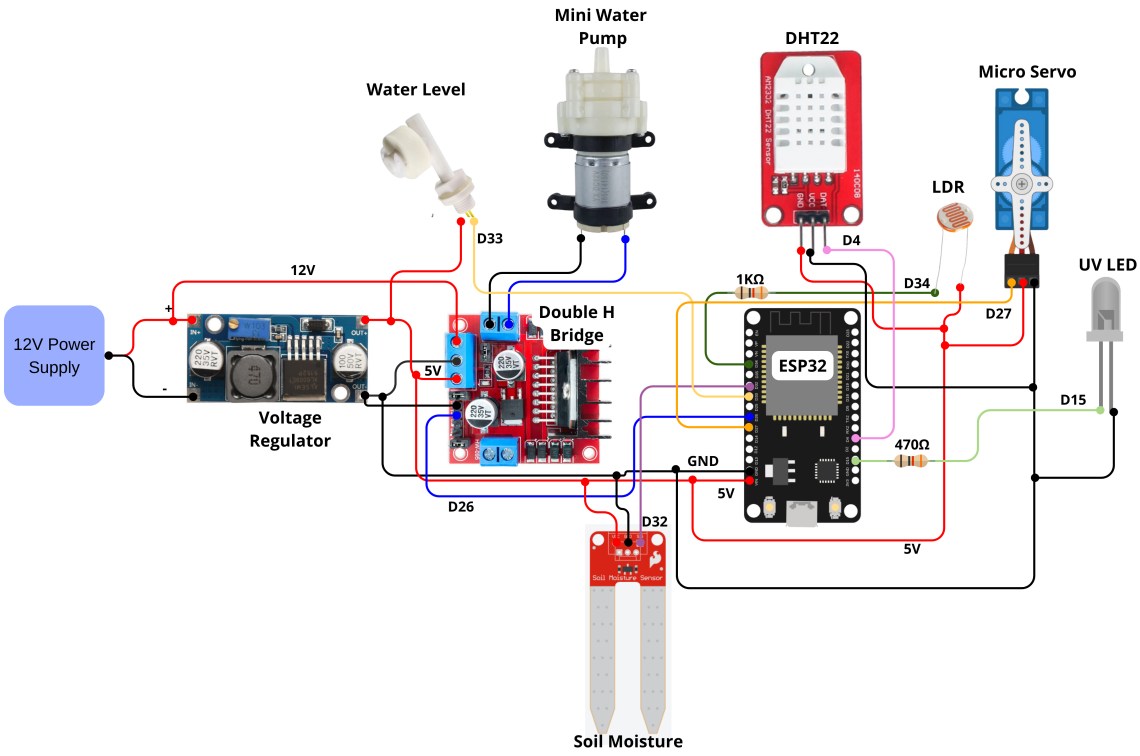


Figura 4: Diagrama de conexões do protótipo
Fonte: Imagem do autor.

plataforma Web e insira os dados de nome e senha da rede na qual o microcontrolador deve ser conectado conforme mostrado na Figura 7. Após a conexão bem sucedida, a ESP32 desativa o ponto de acesso e funciona apenas como um cliente conectado ao backend. Quando conectada a rede, o microcontrolador fornece ao usuário um código (Figura 8) que deve ser inserido na plataforma no momento de cadastro do dispositivo (Figura 16). Caso a conexão não seja bem sucedida com os dados fornecidos pelo usuário, a ESP32 alterna novamente para o modo de ponto de acesso aguardando novos dados para conexão.

Após ser conectada à rede, a Esp32 inicia sua conexão com o back-end por meio de websockets. Para identificar os canais no back-end, é utilizado o valor do endereço MAC (Media Access Control) do microcontrolador como um identificador exclusivo, referenciando o canal ao qual está conectado. Desse modo, quando o usuário requisita uma ação na plataforma, apenas o dispositivo acessado recebe a mensagem via websockets. É importante ressaltar que o endereço MAC apenas é utilizado como um valor de identificação e não como um protocolo de conexão sob o websockets.

3.4 Sistema Web

No presente tópico será descrito os requisitos voltados ao sistema Web, sendo sua implementação dividida entre a interface de usuário e o back-end.

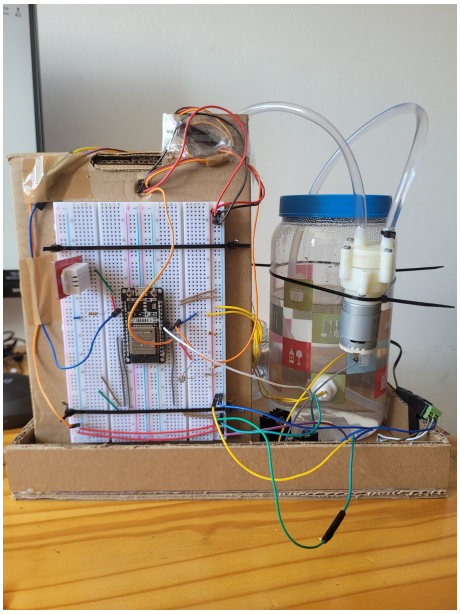


Figura 5: Visão traseira do protótipo
Fonte: Imagem do autor.

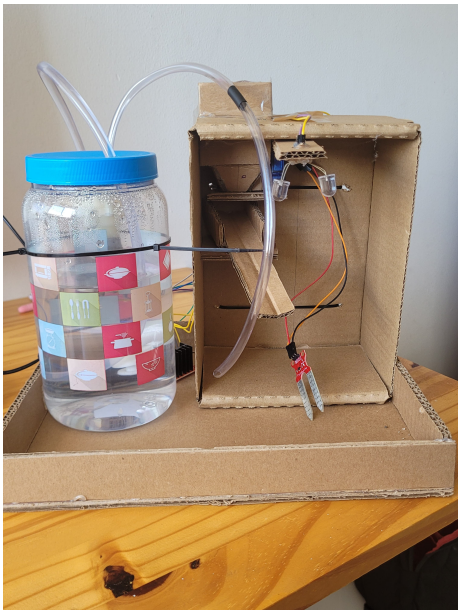


Figura 6: Visão frontal do protótipo
Fonte: Imagem do autor.

Figura 7: Tela de configuração do Wi-fi
Fonte: Imagem do autor.

3.4.1 Requisitos Funcionais

- **[RF16] Cadastro de usuário:** O usuário deve ser capaz de criar um cadastro para uso da plataforma.
- **[RF17] Atualização de cadastro de usuário:** Na plataforma deve ser possível atualizar informações cadastrais do usuário e isso inclui recuperação de acesso.
- **[RF18] Login de usuário:** O usuário deve conseguir se logar na plataforma para acesso de seus recursos.
- **[RF19] Cadastro de dispositivos:** O usuário deve conseguir cadastrar um novo dispositivo na plataforma, isso inclui especificar como esse dispositivo deve atuar e incluir informações de identificação desse dispositivo.

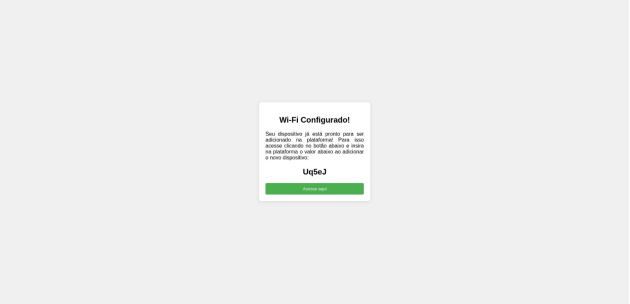


Figura 8: Tela pós configuração do Wi-fi
Fonte: Imagem do autor.

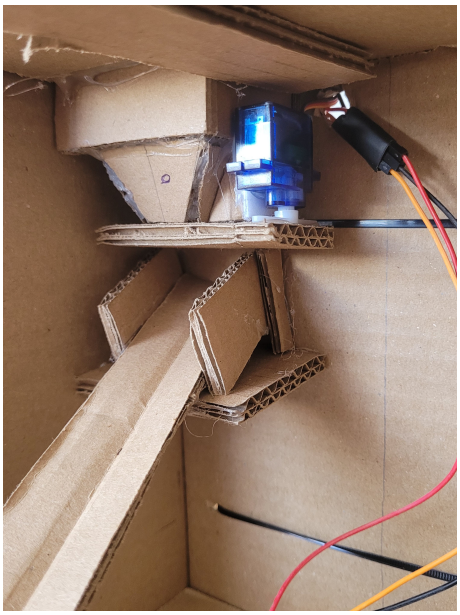


Figura 9: Dispenser de fertilizante
Fonte: Imagem do autor.

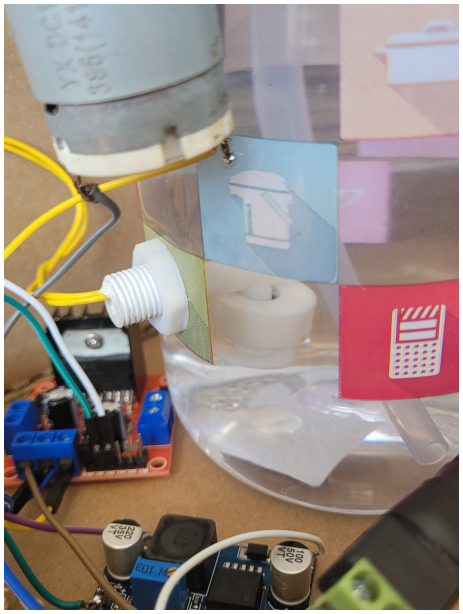


Figura 10: Sensor de nível no reservatório
Fonte: Imagem do autor.

- **[RF20] Edição de dispositivos:** O usuário deve ser capaz de editar as informações do dispositivo cadastrado.
- **[RF21] Visualizar leitura atual do dispositivo:** Deve estar disponível para o usuário a última leitura das condições da planta por sensor na plataforma.
- **[RF22] Visualizar histórico de leituras do dispositivo:** Deve ser disponibilizado para o usuário o histórico de leituras do dispositivo ao longo do tempo.
- **[RF23] Visualizar histórico de ações do dispositivo:** Deve ser disponibilizado para o usuário o histórico de ações do dispositivo ao longo do tempo.
- **[RF24] O usuário deve ser capaz de realizar uma atuação direta para um dispositivo:** O usuário deve ser capaz de realizar uma ação direta no dispositivo seja ela de rega, adubação ou iluminação.

3.5 Desenvolvimento Web

O desenvolvimento tem início pela construção do banco de dados como mostrado no diagrama de entidade relacionamento da Figura 11, a entidade User com os campos de nome e sobrenome, username, email e password. A relação entre um usuário e um dispositivo é representada pela entidade UserDevice, sua importância se dá pelo fato de um dispositivo poder pertencer a mais de um usuário, como também um usuário poder ter mais de um dispositivo. A entidade Action registra comandos de atuação diretos do usuário para o dispositivo, também é utilizada pelo microcontrolador para registrar todas as ações executadas. E por fim temos a classe Reading que registra todas as leituras dos sensores ao longo do tempo por dispositivo.

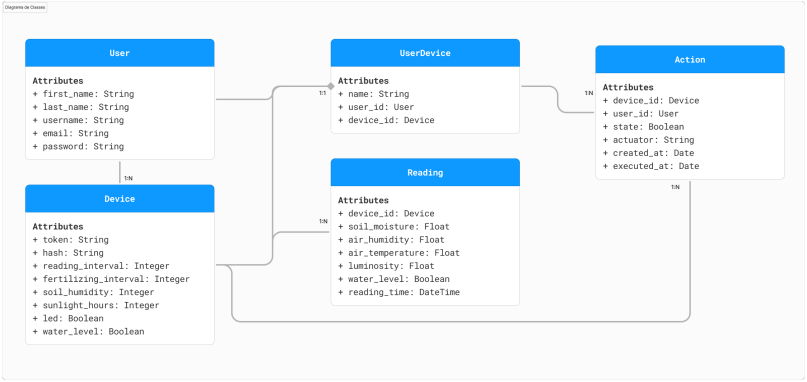


Figura 11: Diagrama de entidade relacionamento
Fonte: Imagem do autor.

A tela inicial do sistema (Figura 12) mostra as opções do usuário para cadastro, login e recuperação de acesso. Caso a opção de cadastro seja escolhida ele será direcionado para a tela exibida na Figura 13. Ao finalizar um dos dois processos ele será direcionado para a tela inicial (Figura 14) que mostra os dispositivos cadastrados e também a opção de cadastrar um novo dispositivo.

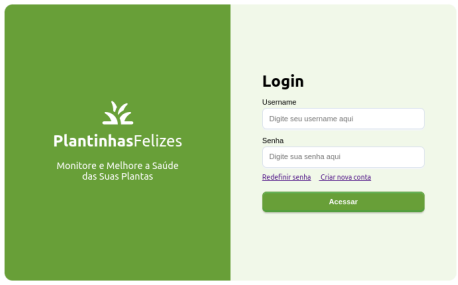


Figura 12: Tela inicial de login
Fonte: Imagem do autor.

Figura 13: Tela de cadastro do usuário
Fonte: Imagem do autor.

Na Figura 14 é possível ver o dispositivo cadastrado com suas principais informações de uso, como intervalo entre leituras, intervalo de adubação e tempo de exposição a luminosidade por dia. Nessa página também é possível ver a opção de edição e de visualização. Caso escolhida a opção de edição ele será direcionado para a página da Figura 15 com o formulário com os campos a serem editados. Ainda nessa tela o usuário consegue acompanhar o alerta caso o reservatório de água esteja com um nível critico.

Figura 14: Tela inicial com dispositivos cadastrados
Fonte: Imagem do autor.

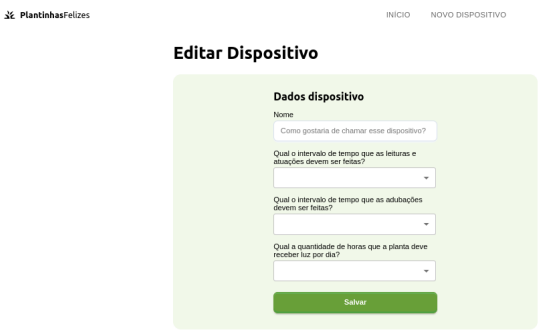


Figura 15: Tela de edição de dispositivo
Fonte: Imagem do autor.

A Figura 16 será acessada quando a opção "Novo dispositivo", presente na tela inicial, for escolhida podendo assim adicionar e configurar um novo dispositivo.

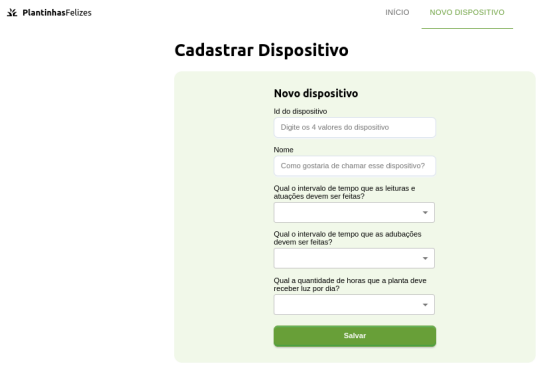


Figura 16: Tela de cadastro do dispositivo
Fonte: Imagem do autor.

Por fim, temos a página de gerenciamento de todas as funções de atuação como também todo o histórico de acompanhamento da planta. No "Anexo A - Tela de histórico do dispositivo"é possível ver uma imagem completa dessa página, tendo no início os dados da ultima leitura dos sensores, seguida de três botões para atuação sendo eles, "REGAR", "LIGAR LUZ"e "ADUBAR". Em seguida é apresentado o "Histórico de Leituras"com todos os dados obtidos nas 10 ultimas leituras dos sensores, com um gráfico para temperatura e umidade do ambiente, um gráfico para umidade do solo e um gráfico para os níveis de luminosidade do ambiente. Por fim temos o histórico de atuações, com os horários que cada um foi executado.

Ainda na tela de visualização do dispositivo o usuário também é alertado caso o reservatório de água esteja em nível crítico, como mostra a Figura 17

Figura 17: Tela de acompanhamento do dispositivo
Fonte: Imagem do autor.

4 APRESENTAÇÃO E ANÁLISE DOS RESULTADOS

Essa seção será destinada a expor os resultados obtidos do sistema desenvolvido.

4.1 Casos de testes

Para validação do principal fluxo de atividades descritas nesse trabalho foi desenvolvido um roteiro de testes, apresentado a seguir. Para o uso dos testes um servidor local foi montado numa máquina com Ubuntu 22.04 LTS e os testes se seguiram usando o navegador Chrome (Versão 121.0.6167.139, 64-bit), e o protótipo físico apresentado na seção anterior (Figuras 5, 6). Os casos de teste foram pensados para agrupar requisitos funcionais que possuem a mesma lógica no fluxo de execução.

- **[CT01] Configuração de Wi-fi: Descrição:** Testar se a configuração inicial do Wi-fi é funcional (RF01) e se os dados de pré cadastro são salvos no banco (RF02).

  Passos: Usuário deve acessar a tela de configuração e inserir os dados do Wi-fi.

  Resultado esperado: Deve ser salvo no banco uma pré configuração do dispositivo. O usuário deve ser redirecionado para tela com código do dispositivo.

  Resultado do teste: Sucesso.

- **[CT02] Leitura e salvamento de dados: Descrição:** Testar se o microcontrolador está fazendo leituras no tempo definido (RF11), e se todos os sensores estão salvando os dados de leitura (RF04, RF05, RF06, RF07, RF15). **Passos:** Dado o tempo pré estabelecido o microcontrolador deve acessar a leitura dos sensores e fazer uma requisição para a API back-end para salvamento desses dados.

  Resultado esperado: Deve ser salvo no banco os dados das leituras dos sensores.

  Resultado do teste: Sucesso.

- **[CT03] Execução de atuação manual: Descrição:** Testar se o microcontrolador está realizando atuações quando requisitado pelo usuário (RF08, RF09, RF10). E se após a execução as informações de execução são salvas pela API back-end. **Passos:** Dado um comando de ação dado pelo usuário utilizando a plataforma as ações do microcontrolador (adubar, iluminar ou regar) deve ser executada.

  Resultado esperado: Deve ser salvo no banco os dados das atuações. Deve ser executada a ação correspondente a requisitada pelo usuário.

  Resultado do teste: Sucesso.

- **[CT04] Execução de atuação automática: Descrição:** Testar se o microcontrolador está realizando atuações no intervalo de tempo correto quando os dados

do ambiente lido pelo sensor estão abaixo do nível pré estabelecido. **Passos:** De acordo com os dados lidos nos sensores o microcontrolador deve decidir por executar ou não alguma atuação.

Resultado esperado: Deve ser salvo no banco os dados das atuações caso elas tenham sido executadas. Caso não esteja no intervalo de leitura estabelecido não deve ser executada nenhuma ação.

Resultado do teste: Sucesso.

- **[CT05] Cadastro de usuário: Descrição:** Testar se um novo usuário consegue realizar a função de cadastro (RF16). **Passos:** Acessar a página inicial da aplicação e clicar em "Criar nova conta".

  Resultado esperado: Deve ser salvo no banco os dados do novo usuário.

  Resultado do teste: Sucesso.

- **[CT06] Login de usuário: Descrição:** Testar se um usuário consegue acessar a plataforma dado seu nome de usuário e senha (RF18). **Passos:** Acessar a página inicial da aplicação adicionar seus dados de login e clicar em "Acessar".

  Resultado esperado: O usuário já cadastrado deve ter acesso a página de dispositivos e o usuário não cadastrado deve permanecer na página de Login.

  Resultado do teste: Sucesso

- **[CT07] Cadastro de dispositivo: Descrição:** Testar se um usuário consegue cadastrar um novo dispositivo usando o código que foi fornecido ao adicionar o Wi-fi no microcontrolador (RF19). **Passos:** Após login na plataforma acessar a opção "Novo Dispositivo", adicionar todas as informações e clicar em "Salvar". Após isso o usuário deve ser redirecionado para a página de dispositivos onde deve ser possível ver o dispositivo cadastrado. Não deve permitir que usuários que não fizeram login realizem o cadastro. Não deve permitir códigos de dispositivos que não existem no banco de dados.

  Resultado esperado: O usuário já cadastrado deve ter acesso a página de cadastrar dispositivo e deve ser capaz de realizar esse cadastro.

  Resultado do teste: Sucesso

- **[CT08] Visualização de última leitura do dispositivo: Descrição:** Testar se um usuário consegue visualizar os últimos dados dos sensores lidos pelo microcontrolador (RF21). **Passos:** Após login na plataforma acessar a opção "Ver"em algum dos dispositivos.

  Resultado esperado: O usuário já cadastrado deve ter acesso a página de visualizar dispositivo e deve ser capaz de ver a ultima leitura salva no banco de dados.

Resultado do teste: Sucesso

- **[CT09] Visualização de histórico de leituras do dispositivo: Descrição:** Testar se um usuário consegue visualizar os gráficos de acompanhamento das leituras feitas pelo dispositivo ao longo do tempo (RF22). **Passos:** Após login na plataforma acessar a opção "Ver"em algum dos dispositivos.

  Resultado esperado: O usuário já cadastrado deve ter acesso a página de visualizar dispositivo e deve ser capaz de ver os gráficos de acompanhamento das ultimas leituras dos sensores ao longo do tempo.

  Resultado do teste: Sucesso

- **[CT10] Visualização de histórico de atuação do dispositivo: Descrição:** Testar se um usuário consegue visualizar a tabela de acompanhamento das atuações feitas pelo dispositivo ao longo do tempo (RF23). **Passos:** Após login na plataforma acessar a opção "Ver"em algum dos dispositivos.

  Resultado esperado: O usuário já cadastrado deve ter acesso a página de visualizar dispositivo e deve ser capaz de ver a tabela de acompanhamento das ultimas atuações dos sensores ao longo do tempo.

  Resultado do teste: Sucesso

- **[CT11] Realizar atuações: Descrição:** Testar se um usuário consegue visualizar os botões de atuação direta ao dispositivo (RF24). **Passos:** Após login na plataforma acessar a opção "Ver"em algum dos dispositivos.

  Resultado esperado: O usuário já cadastrado deve ter acesso a página de visualizar dispositivo e deve ser capaz de ver clicar nos botões de atuação e realizar uma ação direta na planta.

  Resultado do teste: Sucesso

4.2 Análise dos resultados

Dado o mapeamento de casos de teste descritos é possível deduzir que as principais funcionalidades que a aplicação deve cobrir estão funcionais. No "Anexo B - Matriz de Rastreabilidade Casos de Teste e Requisitos Funcionais"é possível acompanhar a relação entre os requisitos funcionais levantados e cobertura dos casos de teste, que validaram 87,5% desses requisitos. Apesar disso, todos os testes foram feitos apenas pelo desenvolvedor desse sistema e seria necessário incluir novos usuários para validar o produto de forma abrangente, como também incluir diferentes tipos de plantas para essa validação. Além disso, é preciso ressaltar que os testes podem não ter sido suficientes para cobrir toda a gama de possibilidades de interações do usuário com a plataforma. Mesmo assim, todo

o desenvolvimento e testes realizados trazem bons resultados para auxiliar os processos de cuidados de plantas e maior facilidade para acompanhar as métricas do ambiente.

5 CONCLUSÕES E TRABALHOS FUTUROS

O sistema desenvolvido se mostrou uma ferramenta capaz de auxiliar o cuidado de plantas em ambiente doméstico. A plataforma permite que o usuário tenha conhecimento do estado em que a planta se encontra além de um histórico detalhado de todas as ações que foram executadas nela. Isso permite que quem a use tenha maior autonomia de controlar remotamente as necessidades da planta como também ter mais tempo para dedicar a outras atividades.

Entretanto, é preciso ressaltar que a plataforma ainda necessita de algumas melhorias como adicionar um sistema de mensageria para validação do cadastro do usuário como também recuperação da conta em caso de perca de alguns dados. Além disso, seria necessário incluir requisitos de acessibilidade como também melhorias na responsividade para dispositivos mobile, embora a maioria das telas já incorporem essa funcionalidade.

Como melhorias simples de funcionalidade a serem implementadas está a paginação dos gráficos de leituras dos sensores e das atuações para que seja possível acompanhar um período de tempo maior da planta como também um estudo de qual seria o tempo de permanência recomendado para armazenamento desses dados ao longo do tempo.

Em adição, como projetos futuros seria interessante incluir técnicas de inteligência artificial para identificar padrões de cuidado das plantas e fornecer essas informações como valores pré definidos para usuários ainda menos experientes no cuidado de plantas. Levando em consideração que a plataforma foi pensada para ser usada no controle de vários dispositivos existe também a possibilidade de testar essa aplicação em ambientes maiores do que vasos domésticos como por exemplo estufas.

Outra grande contribuição como seguimento dessa pesquisa seria analisar o que foi desenvolvido, tentando adaptar para a construção de um framework ou biblioteca que seja capaz de incorporar novos sensores e atuadores de forma flexível.

REFERÊNCIAS

[1] THOMPSON, Richard. Gardening for health: a regular dose of gardening. **Clinical medicine**, v. 18, n. 3, p. 201-205, 2018.

[2] BOTELHO, Juliana de Mello; LAMANO-FERREIRA, Ana Paula do Nascimento; FERREIRA, Mauricio Lamano. Prática de cultivo e uso de plantas domésticas em diferentes cidades brasileiras. Ciência Rural, v. 44, n. 10, p. 1810-1815, 2014.

[3] MACHADO, Altair Toledo; MACHADO, CT de T. Agricultura urbana. 2002.

[4] ROESE, Alexandre Dinnys. Agricultura urbana. 2003.

[5] ASHTON, Kevin et al. That 'internet of things' thing. **RFID journal**, v. 22, n. 7, p. 97-114, 2009.

[6] MOUHA, Radouan Ait Radouan Ait et al. Internet of things (IoT). **Journal of Data Analysis and Information Processing** , v. 9, n. 02, p. 77, 2021.

[7] FETTE, Ian; MELNIKOV, Alexey. **The websocket protocol**. 2011.

[8] DATE, Christopher J. **Introdução a sistemas de bancos de dados**. Elsevier Brasil, 2004.

[9] JATANA, Nishtha et al. A survey and comparison of relational and non-relational database. **International Journal of Engineering Research & Technology**, v. 1, n. 6, p. 1-5, 2012.

[10] DUTONDE, Pratiksha D. et al. Website Developmemt Technologies: A Review. **International Journal for Research in Applied Science and Engineering Technology**, v. 10, n. 1, p. 359-366, 2022.

[11] ZOLKIFLI, Nazatul Nurlisa; NGAH, Amir; DERAMAN, Aziz. Version control system: A review. **Procedia Computer Science**, v. 135, p. 408-415, 2018.

[12] TUMPA, Samira Akter et al. Iot and artificial intelligence based smart gardening and irrigation system. **International Research Journal of Modernization in Engineering Technology and Science**, v. 5, p. 8997-9005, 2023.

[13] PEREIRA, G. P.; CHAARI, M. Z.; DAROGE, F. IoT-Enabled Smart Drip Irrigation System Using ESP32. IoT, 4 (3), 221–243. 2023.

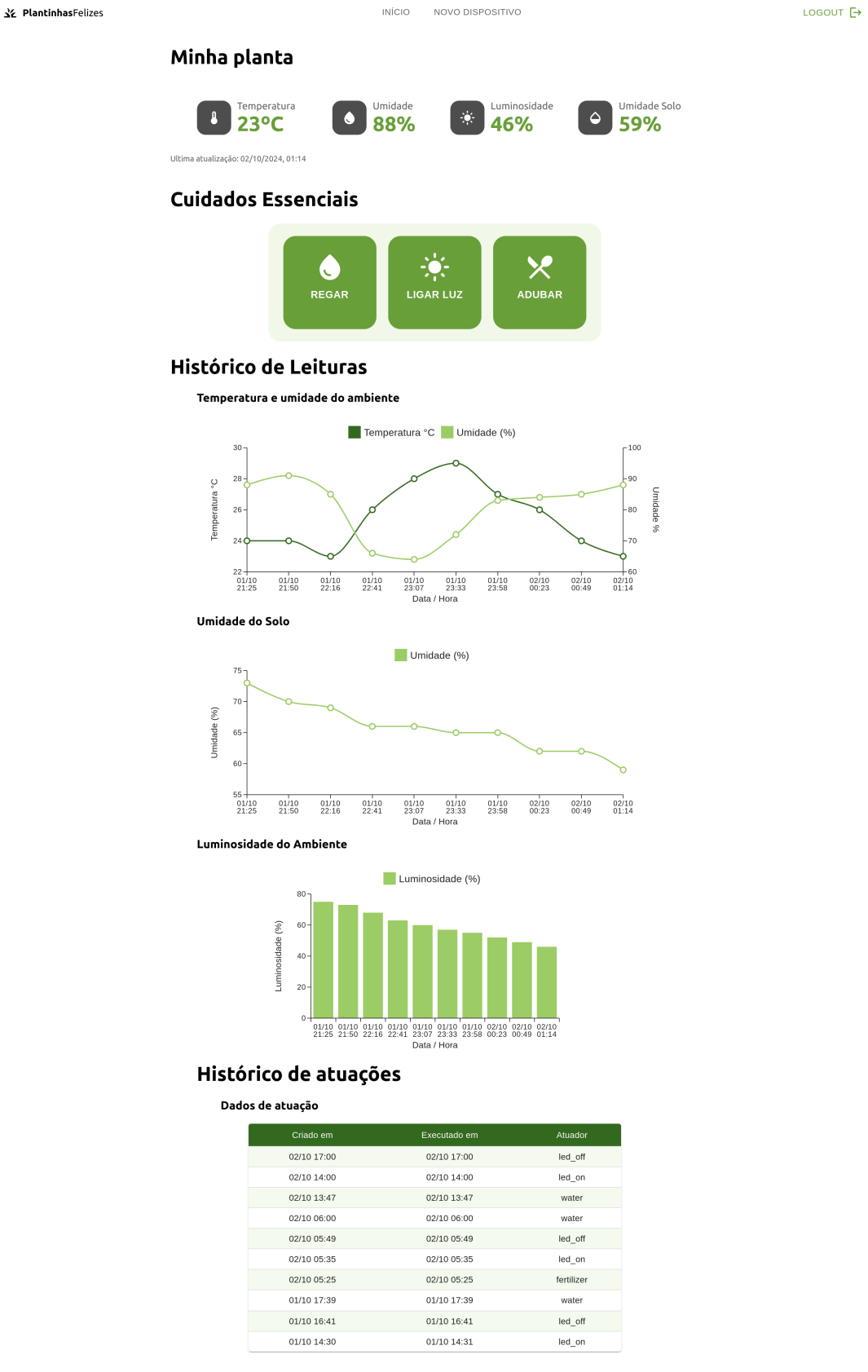
[14] SONG, Mi-Hwa. A study on IoT based real-time plants growth monitoring for smart garden. **International Journal of Internet, Broadcasting and Communication**, v. 12, n. 1, p. 130-136, 2020.

[15] CHOUDHARI, Gauri R. et al. IoT-based Smart Gardening System. In: **Journal of Physics: Conference Series**. IOP Publishing, 2023. p. 012006.

[16] ANÁSTACIO, Lucas; SILVA, Rômulo Augusto Salles da; SAMPAIO, Vitor de Oliveira. Sistema de cultivo inteligente. 2023.

[17] SHARMA, Sachin et al. Smart home gardening management system: A cloud-based internet-of-things (iot) application in vanet. In: **2020 11th International Conference on Computing, Communication and Networking Technologies (ICCCNT)**. IEEE, 2020. p. 1-5.

ANEXO A – Tela de histórico do dispositivo



Fonte: Própria do autor.

ANEXO B – Matriz de Rastreabilidade Casos de Teste e Requisitos Funcionais

Matriz de Rastreabilidade Casos de teste por Requisitos Funcionais																							
	Requisitos funcionais																						
Casos de teste	RF01	RF02	RF03	RF04	RF05	RF06	RF07	RF08	RF09	RF10	RF11	RF12	RF13	RF14	RF15	RF16	RF17	RF18	RF19	RF20	RF21	RF22	RF23	RF24
CT01	X	X																						
CT02				X	X	X	X				X				X									
CT03								X	X	X														
CT04											X	X	X	X										
CT05																X								
CT06																		X						
CT07																			X					
CT08															X						X			
CT09															X							X		
CT10																							X	
CT11																								X

Fonte: Própria do autor.