

**CENTRO PAULA SOUZA**  
**ETEC PROFESSORA ILZA NASCIMENTO PINTUS**  
**Técnico em Desenvolvimento de Sistemas**

**Bruno dos Santos Oliveira**

**Gustavo da Costa Gomes**

**Igor Corrêa de Souza**

**Thaís Pires dos Santos**

**RAINTRACK: SISTEMA INTELIGENTE DE MONITORAMENTO**  
**METEOROLÓGICO**

**São José dos Campos**

**2025**

**Bruno dos Santos Oliveira**

**Gustavo da Costa Gomes**

**Igor Corrêa de Souza**

**Thaís Pires dos Santos**

## **RAINTRACK: SISTEMA INTELIGENTE DE MONITORAMENTO METEOROLÓGICO**

Trabalho de Conclusão de Curso apresentado  
como exigência parcial para obtenção do título  
de Técnico em Desenvolvimento de Sistemas  
pelo Centro Paula Souza – Escola Técnica  
Estadual Professora Ilza Nascimento Pintus -  
São José dos Campos.

São José dos Campos, 02 de dezembro de 2025.

**Banca Examinadora:**

---

Jean Carlos Lourenço Costa

---

Neymar Siqueira Dellareti

## RESUMO

O crescimento da frequência e intensidade de desastres naturais, em especial aqueles relacionados a chuvas intensas, evidencia a necessidade de soluções tecnológicas que auxiliem no monitoramento meteorológico e na prevenção de danos sociais, ambientais e econômicos. Nesse contexto, este Trabalho de Conclusão de Curso apresenta o desenvolvimento do *RainTrack*, um sistema inteligente de monitoramento meteorológico em tempo real. O projeto tem como objetivo coletar, armazenar e disponibilizar informações ambientais de forma acessível e confiável, contribuindo para a tomada de decisões em segurança pública e planejamento urbano. A metodologia envolve a utilização da placa microcontroladora ESP32 integrada a sensores meteorológicos, com transmissão de dados via protocolo MQTT e gerenciamento em servidor Flask conectado a um banco de dados MySQL. Para a interpretação dos dados, foi desenvolvido um dashboard interativo em ambiente web, utilizando tecnologias como HTML, CSS, JavaScript e a biblioteca Highcharts. Os resultados esperados incluem maior precisão na coleta de dados meteorológicos, transmissão estável das informações e visualização clara em tempo real, favorecendo ações preventivas e mitigando os impactos de fenômenos climáticos extremos. Dessa forma, o *RainTrack* se apresenta como uma solução prática, acessível e escalável para o monitoramento meteorológico, com potencial de aplicação em comunidades urbanas e rurais.

**Palavras-chave:** Meteorologia; Internet das Coisas; ESP32; MQTT; Monitoramento Meteorológico.

## ABSTRACT

The increasing frequency and intensity of natural disasters, especially those caused by heavy rainfall, highlight the need for technological solutions that support meteorological monitoring and disaster prevention. In this context, this Final Paper presents the development of *RainTrack*, an intelligent real-time weather monitoring system. The project aims to collect, store, and provide environmental information in a reliable and accessible way, supporting decision-making in public safety and urban planning. The methodology involves the use of the ESP32 microcontroller board integrated with meteorological sensors, data transmission through the MQTT protocol, and management via a Flask server connected to a MySQL database. For data interpretation, an interactive web-based dashboard was developed using technologies such as HTML, CSS, JavaScript, and the Highcharts library. The expected results include higher accuracy in meteorological data collection, stable data transmission, and clear real-time visualization, enabling preventive actions and reducing the impacts of extreme weather events. Thus, *RainTrack* stands out as a practical, accessible, and scalable solution for meteorological monitoring, with potential applications in both urban and rural communities.

**Keywords:** Meteorology; Internet of Things; ESP32; MQTT; Meteorological Monitoring.

## LISTA DE ABREVIATURAS E SIGLAS

IDE *Integrated Development Environment*

MQTT *Message Queuing Telemetry Transport*

VS *Visual Studio*

SQL *Structured Query Language*

HTML *HyperText Markup Language*

CSS *Cascading Style Sheets*

JS *JavaScript*

JSON *JavaScript Object Notation*

CPF *Cadastro de Pessoa Física*

## ÍNDICE DE ILUSTRAÇÕES

Figura 1 - Trello do Projeto.....	27
Figura 2 - Diagrama de Caso de Uso do Sistema.....	28
Figura 3 - Diagrama do Caso de Uso do Administrador.....	28
Figura 4 - Diagrama do Banco de Dados. ....	29
Figura 5 – Tela de Login. ....	30
Figura 6 – Tela Sobre Nós A e B.....	31
Figura 7 – Tela de Boas-Vindas A e B .....	32
Figura 8 - Tela de Gráficos (sem gráfico). ....	32
Figura 9 – Tela de Gráficos (com gráfico). ....	33
Figura 10 – Tela de Parâmetros. ....	33
Figura 11 – Tela de Cadastro de Parâmetros. ....	34
Figura 12 – Tela de Postos. ....	35
Figura 13 – Tela de Cadastro de Postos. ....	35
Figura 14 – Tela de Cadastro de Usuários.....	36
Figura 15 – Tela de Usuários. ....	37
Figura 16 – Tela de Edição de Usuários. ....	37
Figura 17 – Tela de Edição de Parâmetros.....	38
Figura 18 – Tela de Edição de Postos. ....	39
Figura 19 – Tela Meu Perfil. ....	39
Figura 20 – Protótipo do Hardware. ....	40

# SUMÁRIO

<b>1 INTRODUÇÃO</b>	<b>10</b>
1.2 Justificativa	11
1.3 Hipóteses	11
1.4 Objetivo geral	12
1.5 Objetivos específicos	12
1.5.1 Metodologia	12
1.6 Referencial teórico	13
<b>2 CONTEXTUALIZAÇÃO TECNOLÓGICA</b>	<b>14</b>
2.1 Tecnologias Utilizadas	14
2.1.1 Git e GitHub	14
2.1.2 Visual Studio Code	14
2.1.3 Trello	15
2.1.4 MySQL	15
2.1.5 XAMPP	16
2.1.6 Python	16
2.1.7 Flask	17
2.1.8 HTML	17
2.1.9 CSS	17
2.1.10 JavaScript	18
2.1.11 Highcharts	18
2.1.12 ESP32	19
2.1.13 MQTT	19
2.1.14 Eclipse Mosquitto	19
2.2 Soluções Existentes	20
2.2.1 Estações Meteorológicas Comercias	21
2.2.1 Plataformas de Monitoramento IoT	21
2.3 Levantamento de Requisitos	21
2.3.1 Definição de <i>Stakeholders</i>	22
2.3.2 Metodologia Utilizada	22

2.3.3 Requisitos Funcionais .....	22
2.3.3.1 Coleta de Dados Meteorológicos .....	23
2.3.3.2 Criação de Gráficos.....	23
2.3.4.1 Desempenho .....	23
2.3.4.2 Confiabilidade.....	23
2.3.4.3 Escalabilidade .....	23
2.3.4.4 Segurança .....	24
2.3.4.5 Usabilidade .....	24
3 DESENVOLVIMENTO .....	25
3.1 Materiais e Métodos .....	25
3.1.1 Softwares .....	25
3.1.2 Linguagens de Programação .....	25
3.1.3 Hardware .....	26
3.1.4 Cronograma .....	26
3.2 Modelagem.....	27
3.2.1 Diagrama de Caso de Uso do Sistema .....	27
3.2.2 Diagrama de Caso de Uso do Administrador .....	28
3.2.3 Modelo Lógico .....	29
4 RESULTADOS E DISCUSSÃO .....	30
4.1 Protótipo do <i>Site</i> .....	30
4.1.1 Tela de <i>Login</i> .....	30
4.1.2 Tela Sobre Nós .....	31
4.1.3 Tela de Boas-Vindas .....	31
4.1.4 Tela de Gráficos .....	32
4.1.5 Tela de Parâmetros .....	33
4.1.6 Tela de Cadastro de Parâmetros.....	34
4.1.7 Tela de Postos .....	34
4.1.8 Tela de Cadastro de Postos .....	35
4.1.9 Tela de Cadastro de Usuários .....	36
4.1.10 Tela de Usuários.....	36
4.1.11 Tela de Edição de Usuários .....	37
4.1.12 Tela de Edição de Parâmetros .....	38
4.1.13 Tela de Edição de Postos .....	38



<b>4.1.14 Tela Meu Perfil .....</b>	<b>39</b>
<b>4.2 <i>Hardware</i> do Projeto .....</b>	<b>40</b>
<b>4.3 Análise dos Resultados .....</b>	<b>40</b>
<b>5 CONCLUSÃO .....</b>	<b>41</b>
<b>REFERÊNCIAS.....</b>	<b>42</b>

# 1 INTRODUÇÃO

O estudo da Meteorologia é estudar detalhadamente e a fundo as diferentes condições da atmosfera ao longo do tempo. Sendo muito importante para as atividades humanas, avanços na sociedade, e também essencial para a previsão de fenômenos climáticos extremos por vir, assim ajudando a criar medidas preventivas para a redução dos impactos negativos (MUNDO EDUCAÇÃO, 2025).

Porém, em 2023, o mundo sofreu um grande aumento no número de desastres naturais, registrando em torno de 240 catástrofes em diversas regiões. Como consequência, quase 74 mil pessoas perderam a vida. Entre todos os eventos impactantes, os desastres provocados por chuvas intensas se destacam (EOS DATA ANALYTICS, 2024; FOLHA DE S. PAULO, 2024).

Os desastres citados acabam dificultando muitas áreas do local afetado, como a destruição de ecossistemas, a perda de biodiversidade e o desequilíbrio ambiental, prejudicando imensamente os ciclos biogeoquímicos. Além disso, a poluição do solo, da água e do ar intensificam o surgimento de doenças, enquanto os danos à estrutura causam perdas materiais e o deslocamento de pessoas. Como resultado, há prejuízos econômicos para o Estado e para a população (BRASIL ESCOLA, 2025).

A crescente intensidade dos desastres naturais reforça a importância de investimentos em meteorologia para prever e minimizar seus impactos. Aprimorar os sistemas de monitoramento climático e alerta adiantado, empregando tecnologias como sensores e radares, permite atitudes preventivas, minimizando danos materiais e protegendo vidas.

Além disso, a parceria entre governos, cientistas, e a população, é fundamental para reforçar a resiliência das comunidades vulneráveis, dado que, a difusão de informações meteorológicas confiáveis, possibilita ações precoces que diminuem perigos e perdas, decorrentes de fenômenos climáticos severos. Dessa forma, a ciência meteorológica se torna um pilar fundamental para a mitigação dos impactos ambientais e sociais dos desastres naturais (PLUGFIELD, 2025).

Diante do crescente impacto de desastres naturais provocados por chuvas intensas, este Trabalho de Conclusão de Curso (TCC) propõe o desenvolvimento do sistema 'RainTrack', focado no monitoramento meteorológico em tempo real. O sistema visa coletar e processar dados ambientais, facilitando a análise das condições climáticas e auxiliando na segurança pública e no planejamento urbano.

Usando tecnologia tipo ESP32 e MQTT, o sistema vai assegurar uma transmissão de dados boa e confiável. Além disso, está previsto um painel interativo que vai facilitar a interpretação e visualização das informações coletadas. Espera-se, assim, entregar uma solução prática e eficaz para melhorar a gestão das condições climáticas, ajudando a reduzir os efeitos de desastres naturais.

## **1.2 Justificativa**

A crescente frequência e intensidade de desastres relacionados a chuvas intensas reforçam a importância de um monitoramento meteorológico eficiente. O desenvolvimento do 'RainTrack' visa não apenas coletar dados em tempo real, mas também proporcionar uma análise visual acessível, contribuindo para a segurança pública e o planejamento urbano. A implementação de tecnologias como o ESP32 e o protocolo MQTT permitirá uma transmissão de dados confiável e em tempo real, essencial para a tomada de decisões informadas. Dessa forma, busca-se solucionar a problemática identificada, oferecendo um sistema capaz de otimizar a coleta e organização das informações meteorológicas, facilitando sua análise e aplicação.

## **1.3 Hipóteses**

A hipótese central deste projeto é que a implementação do sistema de monitoramento meteorológico 'RainTrack' permitirá uma coleta de dados mais precisa e em tempo real, resultando em uma melhor análise das condições climáticas e contribuindo para a redução dos impactos de desastres naturais. Além disso, espera-se que a visualização dos dados por meio de um dashboard interativo facilite a interpretação das informações e a tomada de decisões pelos usuários.

## 1.4 Objetivo geral

O objetivo geral desse trabalho é desenvolver um sistema de monitoramento meteorológico automatizado.

## 1.5 Objetivos específicos

Para alcançar o objetivo geral serão necessários desenvolver os seguintes objetivos específicos:

- Implementar um servidor *Flask* conectado a um banco de dados MySQL;
- Criar uma comunicação entre o posto meteorológico (ESP32) e o servidor flask, utilizando o protocolo MQTT;
- Desenvolver um dashboard interativo que apresente as informações meteorológicas;
- Garantir a integridade e confiabilidade dos dados coletados, minimizando perdas;
- Realizar o *deploy* da aplicação web na plataforma PythonAnywhere, assegurando sua disponibilidade online, acesso remoto e facilidade de manutenção.

### 1.5.1 Metodologia

A metodologia adotada para o desenvolvimento do "RainTrack" inclui as seguintes etapas:

Levantamento de requisitos para definição dos dados a serem coletados e escolha das tecnologias adequadas a serem utilizadas. Realizar a programação da placa microcontrolador ESP32 para leitura dos sensores meteorológicos e envio dos

dados via protocolo MQTT. Implementar um servidor *Flask* para recepção, processamento e armazenamento dos dados em um banco de dados MySQL. Desenvolver um dashboard web interativo para visualização gráfica dos dados em tempo real. Verificarmos a precisão dos dados coletados, estabilidade da comunicação entre os dispositivos e desempenho do sistema.

## **1.6 Referencial teórico**

O referencial teórico deste estudo abordará conceitos fundamentais sobre monitoramento meteorológico, análise de dados climáticos e desenvolvimento de softwares especializados para o setor. Serão exploradas pesquisas acadêmicas e publicações que tratam da importância do monitoramento climático para a sociedade, destacando sua relevância para áreas como segurança pública, agricultura e planejamento urbano.

Além disso, será discutido o uso de tecnologias de transmissão de dados, como o protocolo MQTT, que possibilita a comunicação eficiente entre dispositivos IoT, e o ESP32, um microcontrolador amplamente utilizado em projetos de monitoramento ambiental. A importância da visualização de dados em tempo real também será abordada, considerando sua influência na interpretação e tomada de decisões.

A literatura revisada servirá como base para a fundamentação teórica do projeto e para a análise dos resultados obtidos, garantindo que a abordagem adotada esteja alinhada com estudos e soluções existentes na área.

## 2 CONTEXTUALIZAÇÃO TECNOLÓGICA

Neste capítulo, detalharemos as tecnologias utilizadas para a solução dos problemas, uma pesquisa das soluções existentes e um levantamento de requisitos.

### 2.1 Tecnologias Utilizadas

Esta seção do relatório Técnico abordará as principais tecnologias utilizadas para solução do nosso problema.

#### 2.1.1 Git e GitHub

O Git é um sistema de controle de versões amplamente utilizado no desenvolvimento de software. Ele permite o acompanhamento detalhado das alterações no código e facilita a colaboração entre desenvolvedores, mantendo o histórico completo de modificações. Segundo a documentação oficial, “o Git é um sistema de controle de versão distribuído, rápido, escalável e com um conjunto de comandos incomumente rico que oferece operações de alto nível e acesso completo aos seus recursos” (GIT-SCM, 2025). A cada nova atualização no código, o Git cria um "registro" do que foi alterado, o que facilita a integração de modificações feitas por diferentes pessoas sem conflitos.

A principal razão para utilizar o Git é sua capacidade de oferecer uma solução segura e eficiente para o controle de versões do código, especialmente em projetos com equipes grandes ou distribuídas. Além disso, ele oferece uma maneira robusta de gerenciar conflitos de código e facilita o trabalho com ramificações (*branches*), permitindo que os desenvolvedores testem alterações sem afetar a versão principal do projeto.

#### 2.1.2 Visual Studio Code

O *Visual Studio Code* é um editor de código desenvolvido pela Microsoft, é amplamente utilizado na programação devido sua facilidade, robustez e agilidade. É

gratuita e com código aberto, sendo compatível com vários sistemas operacionais, como Linux, macOS e Windows.

Contém uma interface intuitiva e com ampla gama de personalização, permitindo o usuário a melhorar seu fluxo de trabalho. Ele oferece suporte nativo para diversas linguagens de programação, através das extensões disponíveis, pode adquirir diversas funcionalidades como formatação automática, integração com sistemas de controle de versão e integração a plataformas de hospedagem de código (VISUAL STUDIO CODE, 2025).

Sua ampla comunidade garante atualizações frequentes e uma enorme variedade de plugins, permitindo a adaptabilidade a qualquer projeto. O uso do VS Code contribui com aumento na produtividade, organização e eficiência na escrita do código, contendo no mesmo ambiente a maior parte das ferramentas necessárias para o desenvolvimento e testes de um projeto.

### **2.1.3 Trello**

Baseada no método Kanban, é usada para o gerenciamento de tarefas e projetos, permitindo melhor organização no fluxo de trabalho de maneira visual e colaborativa, conforme a Atlassian, “o Trello é uma ferramenta de colaboração que organiza seus projetos em quadros, permitindo acompanhar tarefas e fluxos de trabalho de forma simples e visual” (TRELLO, 2025). Sendo utilizada por equipes no planejamento, desenvolvimento e conclusão de atividades.

Tem como vantagem sua facilidade, a possibilidade de colaboração em tempo real de todos os membros da equipe, e a integração com diversas ferramentas como Microsoft Teams e GitHub. A partir dessas integrações ele otimiza a comunicação e realização do progresso do trabalho.

### **2.1.4 MySQL**

O gerenciador de banco de dados relacional, é muito utilizado em sistemas e projetos *web* que necessitam de armazenamento e manipulação de dados, de acordo com a Oracle, “o MySQL é o sistema de gerenciamento de banco de dados de código aberto mais popular do mundo” (MYSQL, 2025). Tem código aberto e utiliza a

linguagem de programação SQL, o que permite a criação, consulta, edição e exclusão de dados. Possuindo uma ampla comunidade e extensa documentação, se tornando muito fácil resolver problemas.

Entre suas características se destaca sua alta performance, portabilidade, confiabilidade, e a compatibilidade com inúmeras linguagens de programação, como Python, Java e C#. Também possui suporte para transações, integridade referencial, índices e mecanismos de armazenamento sofisticados, como o InnoDB, permitindo a adaptação do banco para a necessidade do projeto.

### **2.1.5 XAMPP**

É um pacote de *softwares* livre e multiplataforma, fornecendo um ambiente integrado para o desenvolvimento e teste dos projetos web de forma local, segundo a Apache Friends, “o XAMPP é uma distribuição Apache fácil de instalar que contém MariaDB, PHP e Perl” (APACHE FRIENDS, 2025). Seu nome vem do acrônimo X de multiplataforma, A de Apache, M de MariaDB, P de PHP e P de Perl, assim juntando todos os recursos necessários para simular um servidor.

Pela sua facilidade de configuração e instalação, permite aos usuários executarem seus projetos sem a necessidade de configurar um servidor remoto. Ele é usado para o desenvolvimento, depuração e validação de sistemas antes de serem publicados, assim agilizando o processo de implementação e reduzindo os custos do projeto.

### **2.1.6 Python**

É uma linguagem de programação, de alto nível e interpretada, reconhecida pela sua simplicidade, versatilidade e legibilidade. Foi criada por Guido van Rossum em 1991, adotando uma sintaxe clara, facilitando a aprendizagem e acelerando o desenvolvimento dos projetos (PYTHON SOFTWARE FOUNDATION, 2025).

Possui uma extensa biblioteca de pacotes e *frameworks*, permitindo a aplicação em diversas áreas, indo desde o desenvolvimento *web* e análise de dados até inteligência artificial e computação científica. Podendo ser executada em diversos sistemas operacionais por ser multiplataformas. Tem uma comunidade muito grande e ativa, contribuindo com novas ferramentas e suporte a problemas.



### 2.1.7 Flask

De acordo com o Pallets Project, “Flask é um *microframework web* escrito em Python, leve, flexível e fácil de usar” (PALLETS PROJECT, 2025), muito utilizado no desenvolvimento de aplicações e serviços *web*. É classificado como um *microframework*, pois fornece apenas o essencial para o desenvolvimento do projeto, permitindo a integração a extensões necessárias.

É leve e com curva de aprendizado acessível, facilitando a integração com bibliotecas e banco de dados. Oferece um suporte e roteamento das URLs, *templates* dinâmicos e mecanismos de autenticação, contando com ampla comunidade e documentação robusta. Não impõe ao projeto uma estrutura rígida, permitindo a liberdade ao desenvolver para organizar o código. O tornando ideal tanto para projetos pequenos quanto aos de grande escala.

### 2.1.8 HTML

O *HyperText Markup Language* é usado para a estruturação e de aplicações *web*. Ele é uma linguagem de marcação que define o que é mostrado nos navegadores, permitindo a criação de diversos tipos de elementos, como: títulos, *links*, tabelas, imagens, formulários, entre outros que compõe a interface do sistema (WHATWG, 2025).

Sua principal função é estruturar o conteúdo de forma hierárquica, sendo a base do desenvolvimento *web*. Pode ser combinado com outras tecnologias, como o CSS para a estilização, o JavaScript responsável pela interação, formando o conjunto essencial para a criação de interfaces modernas.

### 2.1.9 CSS

O *Cascading Style Sheets* é responsável por definir a aparência e o estilo das páginas de uma aplicação *web*, de acordo com o W3C, “CSS é o mecanismo usado para adicionar estilo — como fontes, cores e espaçamento — a documentos *web*” (W3C, 2025). Enquanto o HTML é responsável pela estrutura o CSS atua na camada de apresentação, controlando os aspectos visuais, como cores, tamanhos, fontes, espaçamentos, animações, bordas, entre outros.

Sua principal função é a separação entre conteúdo e *design*, facilitando a manutenção do código, além de proporcionar maior consistência nas diversas páginas de um projeto. Ele possibilita também a criação de interfaces responsivas, que se

adaptam a diferentes tamanhos de tela, garantindo uma melhor experiência para o usuário. Ele pode ser combinado com diversas bibliotecas e frameworks que agilizam o processo de desenvolvimento.

### 2.1.10 JavaScript

É uma linguagem de programação interpretada, amplamente usada no desenvolvimento web para adicionar interatividade e dinamismo as páginas do projeto. Sendo responsável pela lógica e comportamento dos elementos da interface. Conforme a MDN, “JavaScript é a linguagem de programação que permite a implementação de funcionalidades complexas em páginas da *web*” (MOZILLA DEVELOPER NETWORK, 2025).

Entre suas principais funcionalidades estão: a validação de formulários, o tratamento de eventos, a criação de animações e a comunicação assíncrona com os servidores. Isso permite que as aplicações sejam mais interativas e responsivas, melhorando a experiência dos usuários. Uma das suas grandes vantagens é a versatilidade, já que roda diretamente nos navegadores. Possui uma grande variedade de bibliotecas e *frameworks*, como *React*, *Angular*, e *Vue.js*, usadas para otimizar o desenvolvimento de aplicações modernas.

### 2.1.11 Highcharts

Segundo a documentação oficial, “o Highcharts é uma biblioteca JavaScript para criação de gráficos interativos para sites e aplicações *web*” (HIGHCHARTS, 2025). Sendo muito utilizada em *dashboards* e sistemas de visualização de dados, oferece suporte a vários tipos de gráficos, como linhas, colunas, pizza, áreas e até gráficos em tempo real, possibilitando a apresentação de grandes volumes de dados da forma mais clara possível.

Sua principal vantagem é a fácil integração com outras tecnologias, podendo ser utilizado diretamente com HTML e CSS e *frameworks* de *backend* como Flask e Django. Além disso, é altamente personalizável, permitindo o ajuste de cores, legendas, animações e eventos de interação, garantindo maior flexibilidade na apresentação dos dados. Foi projetado para ser responsivo se adaptando automaticamente os gráficos a diferentes dispositivos e tamanhos de tela.

### 2.1.12 ESP32

De acordo com a Espressif, “o ESP32 é uma série de microcontroladores de baixo custo e baixo consumo de energia com *Wi-Fi* e *Bluetooth* integrados” (ESPRESSIF, 2025). Utiliza um processador *dual-core* de 32 *bits* com conectividade integrada a *Wi-fi* e *Bluetooth*, o que o torna uma solução completa para aplicações que necessitam uma comunicação sem fio.

Entre suas principais características está o baixo consumo de energia, diversas interfaces de entrada e saída, suporte a diferentes protocolos de comunicação e possibilidade de integração com diversos sensores. Assim o ESP32 pode ser utilizado em diversos tipos de aplicações, desde a automação residencial e monitoramento ambiental até sistemas embarcados mais complexos.

### 2.1.13 MQTT

O *Message Queuing Telemetry Transport* (MQTT) é um protocolo de comunicação leve, muito utilizado em projetos de Internet das Coisas, devido ao seu baixo consumo de banda e energia (OASIS, 2025). Se baseia no modelo publicador/assinante, permitindo a troca de mensagens entre dispositivos através de um *broker*, que atua como intermediário entre os emissores e receptores.

Sua principal vantagem é a simplicidade e conseguir operar com recursos ilimitados, assim pode garantir a confiabilidade da entrega das mensagens por meio de diferentes níveis de qualidade de serviço. Ele independe de plataforma, podendo ser implementado diferentes dispositivos, linguagens e aplicações. Sendo indicado para aplicações que exigem comunicação em tempo real e monitoramento contínuo.

### 2.1.14 Eclipse Mosquitto

É um *broker* de código aberto, que é utilizado para implementar o protocolo MQTT, sendo muito encontrado em aplicações de Internet da Coisas e em sistemas de monitoramento (ECLIPSE FOUNDATION, 2025). Têm como principal função administrar a comunicação entre assinantes e publicadores, garantindo a eficiência, confiabilidade e baixo consumo de recursos na transmissão da mensagem.

Possui entre suas características a escalabilidade, leveza e compatibilidade com os diferentes níveis de qualidade de serviço que o MQTT define, assim permitindo o projeto se adaptar a entrega de mensagens que necessita. Ele pode ser executado em Windows, MacOS e Linux além de poder estar presente em sistemas embarcados

pelo baixo consumo de dados. Sua fácil configuração e a possibilidade de integrá-lo a diversas linguagens de programação o torna adequado para projetos pequenos e até de larga escala.

## 2.2 Soluções Existentes

Nesta seção apresentaremos uma tabela para a comparação com outras soluções, para assim mostrar as vantagens que possuímos

Tabela 1: Comparativo com soluções existentes

<b>Características</b>	<b>Davis Instruments Vantage Pro2</b>	<b>Weather Underground</b>	<b>Plugfield</b>	<b>RainTrack</b>
Coleta de dados em tempo real	X	X	X	X
Baixo custo de implementação		X		X
Acesso via plataforma web		X	X	X
Dashboard interativo com gráficos			X	X
Uso de tecnologias IoT (ESP32, MQTT)		X	X	X
Customização e escalabilidade				X
Integração com banco de dados próprio				X
Foco em comunidades locais			X	X
Custo de manutenção baixo				X
Alta precisão dos sensores	X		X	X
Acesso gratuito		X		X
Código aberto				X
Visualização em dispositivos móveis	X	X	X	X

Objetivo principal	Monitoramento profissional de alta precisão	Compartilhamento colaborativo de dados meteorológicos	Sustentabilidade e prevenção de desastres	Monitoramento meteorológico local em tempo real
--------------------	---	---	---	---

### 2.2.1 Estações Meteorológicas Comerciais

As estações meteorológicas comerciais, como as produzidas pela *Davis Instruments* ou *Oregon Scientific*, oferecem sensores para temperatura, umidade, pressão atmosférica, direção e velocidade do vento, além de pluviômetros. Esses sistemas possuem boa confiabilidade, mas apresentam custo elevado e pouca flexibilidade de integração com aplicações personalizadas.

### 2.2.1 Plataformas de Monitoramento IoT

Com o avanço da Internet das Coisas, surgiram plataformas como o *Weather Underground* e soluções baseadas em NodeMCU ou ESP32, que permitem usuários compartilharem dados meteorológicos coletados em tempo real. Essas soluções apresentam baixo custo e grande comunidade de suporte, porém, em muitos casos, não oferecem interfaces robustas de visualização ou garantias de confiabilidade dos dados.

## 2.3 Levantamento de Requisitos

O levantamento de requisitos é uma etapa essencial para definir as funcionalidades, restrições e objetivos do projeto. Nesta fase, são identificados os elementos que devem compor o sistema, bem como os atores envolvidos e as metodologias de desenvolvimento a serem aplicadas.

### **2.3.1 Definição de *Stakeholders***

Este termo se refere a qualquer pessoa, grupo ou organização, que impactem ou influenciem, direta ou indiretamente em um projeto. Podendo desempenhar papéis passivos, onde apenas são afetados pelas consequências das ações do projeto, ou então ativos, contribuindo no desenvolvimento e tomada de decisões.

Podem obter duas classificações, internos, sendo membros da equipe, gerentes ou acionistas, e externos, como clientes, fornecedores e a comunidade. gestão destes é fundamental para o bom andamento e conclusão de um projeto, pois através deles os riscos diminuem, o alinhamento de expectativas é mais fácil e garantem que as necessidades e preocupações de todos envolvidos sejam consideradas.

Sendo o *stakeholder* do nosso projeto o Diogo Branquinho professor na FATEC e dono da empresa TECSUS, que nos requisitou esse sistema que é voltado as ações de sua empresa, que atua na área da sustentabilidade, monitoramento da água, gestão de energia elétrica, controle do gás e etc.

### **2.3.2 Metodologia Utilizada**

A metodologia aplicada no desenvolvimento do *RainTrack* foi baseada em práticas ágeis, utilizando elementos do Scrum e ferramentas de apoio como o Trello para gerenciamento de tarefas. O desenvolvimento foi dividido em sprints, cada uma focada em um conjunto de funcionalidades, como programação da ESP32, implementação do servidor Flask, integração com o banco de dados e criação do dashboard.

Essa abordagem permitiu maior controle sobre o progresso do trabalho, flexibilidade para ajustes de requisitos ao longo do projeto e colaboração contínua entre os membros da equipe.

### **2.3.3 Requisitos Funcionais**

Os requisitos funcionais do projeto envolvem uma série de funcionalidades para garantir resultados satisfatórios no funcionamento do projeto, esta seção abordará os principais requisitos funcionais.

#### **2.3.3.1 Coleta de Dados Meteorológicos**

Serão coletados dados de temperatura, umidade e precipitação via sensores Esp32

#### **2.3.3.2 Criação de Gráficos**

A partir dos dados coletados pelos sensores, serão criados gráficos responsivos e interativos, no qual permitirão que o utilizador tenha facilidade para interpretação de dados e detalhes.

#### **2.3.4 Requisitos Não Funcionais**

Os requisitos não funcionais definem características de qualidade que o sistema deve atender, assegurando não apenas seu funcionamento, mas também seu desempenho, segurança e usabilidade.

##### **2.3.4.1 Desempenho**

O sistema deverá garantir tempo de resposta inferior a 3 segundos para atualização dos dados no dashboard, a fim de manter a visualização em tempo real.

##### **2.3.4.2 Confiabilidade**

O sistema deverá assegurar taxa mínima de entrega de 95% das mensagens transmitidas via protocolo MQTT, reduzindo perdas de dados.

##### **2.3.4.3 Escalabilidade**

O sistema deverá permitir a integração futura de novos sensores e postos meteorológicos, sem a necessidade de grandes modificações na arquitetura já existente.

#### **2.3.4.4 Segurança**

A comunicação entre a posto meteorológico e o servidor deverá utilizar autenticação no broker MQTT, além de gerenciamento de permissões no banco de dados para evitar acessos não autorizados.

#### **2.3.4.5 Usabilidade**

O dashboard deverá apresentar interface intuitiva, responsiva e compatível com dispositivos móveis, permitindo que usuários com pouco conhecimento técnico consigam interpretar os dados apresentados.



## 3 DESENVOLVIMENTO

Neste capítulo, apresentaremos o processo de desenvolvimento do projeto *RainTrack*. Divididas em: Materiais e Métodos, modelagem, contendo os Diagramas de Caso de Uso e Modelo Lógico, pôr fim a Programação do Sistema. O desenvolvimento do projeto foi gerenciado por meio do controle de versão Git e do repositório público no GitHub, disponível em: <https://github.com/RainTrackTCC/RainTrack>

### 3.1 Materiais e Métodos

Nos tópicos a seguir serão detalhadas as configurações do ambiente, contendo os softwares e linguagens de programação utilizados no desenvolvimento do projeto. Também serão abordados os resultados da pesquisa de campo e o cronograma de tarefas.

#### 3.1.1 Softwares

Para o desenvolvimento do projeto, foram utilizados os seguintes softwares:

- *Visual Studio Code* versão 1.104 como editor de código;
- O *framework* e as *bibliotecas* da linguagem Python Flask versão 3.1.0, Blinker versão 1.9.0, Click versão 8.1.8, Colorama versão 0.4.6, Itsdangerous versão 2.2.0, Jinja2 versão 3.1.6, MarkupSafe versão 3.0.2, Mysql-connector versão 2.2.9, Paho-mqtt versão 2.1.0, PyMySQL versão 1.1.1, Python-dotenv versão 1.1.0 e Werkzeug 3.1.3;
- O banco de dados utilizado foi o MySQL versão 8.4;
- Para o versionamento dos códigos foi usado o Git versão 2.51.0 e a hospedagem da produção do trabalho foi utilizado o GitHub;

#### 3.1.2 Linguagens de Programação

A linguagem de programação utilizada foi o Python em conjunto com seu *framework* Flask. Também foi usada a linguagem de hipertexto HTML, o CSS para a estilização das telas e o JS para a adição de funcionalidades e criação dos gráficos interativos.

### 3.1.3 Hardware

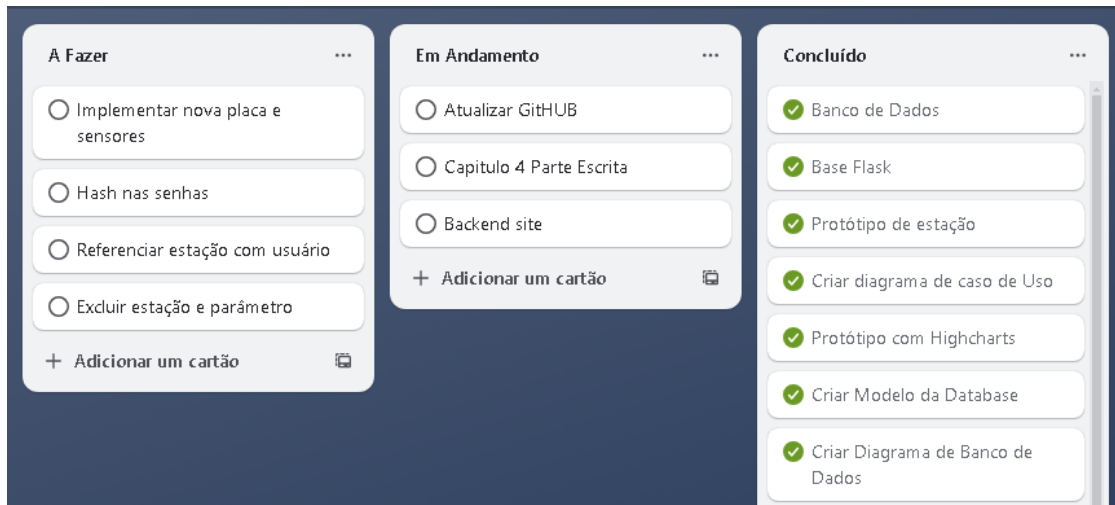
Neste tópico serão descritas as configurações dos computadores utilizados na produção do desenvolvimento do RainTrack:

- *Notebook 1*  
Processador: Intel(R) Core (TM) i3-10110U CPU @ 2.10GHz (2.59 GHz)  
Memória Instalada (RAM): 4,00 GB  
SSD: 480 GB
- *Notebook 2*  
Processador: Intel(R) Core (TM) i7-12650 HX CPU @ 3.30GHz (4.70GHz)  
Memória Instalada (RAM): 16,00GB  
SSD: 128GB  
Placa de Vídeo: Nvidia RTX 2060
- *Desktop*  
Processador: AMD Ryzen 5 5500 @ 3.6Ghz  
Memória Instalada (RAM): 16,00GB  
SSD: 540GB  
Placa de Vídeo: Ryzen rx6600

### 3.1.4 Cronograma

Para o gerenciamento do cronograma foi utilizado a plataforma Trello, onde eram colocadas as tarefas a serem realizadas e conforme ela fosse sendo executada o status dela era atualizada. Através dessa ferramenta o cronograma do desenvolvimento do projeto se tornou muito mais organizado e visível, facilitando o controle e a visualização do desenvolvimento projeto. A Figura 1, ilustra o desenvolvimento do projeto no Trello.

Figura 1 - Trello do Projeto



Fonte: Autores, (2025)

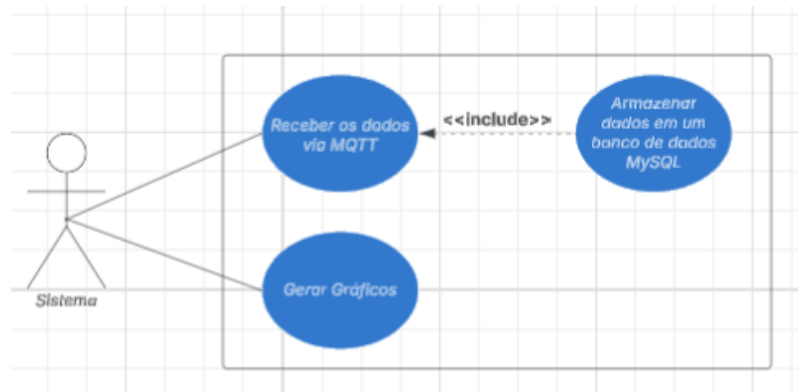
## 3.2 Modelagem

A modelagem dos dados é uma etapa essencial para o projeto, já que ela vai definir as características de funcionamento do sistema, estabelecendo como esses dados serão organizados e quais são os relacionamentos presentes entre eles. Os principais objetivos dessa etapa, são: representar os dados de forma estruturada, realizar a documentação, fornece mecanismos de validação e identificar os processos de relacionamento.

### 3.2.1 Diagrama de Caso de Uso do Sistema

O Diagrama de Caso de Uso serve para descrever as diversas funcionalidades de um *software*, o comportamento que cada ator assume e como eles se relacionam entre si. Na Figura 2, o Sistema atua como principal do sistema onde ele recebe os dados via MQTT, os armazena em um banco de dados MySQL e por fim gera os gráficos através dos dados armazenados.

Figura 2 - Diagrama de Caso de Uso do Sistema

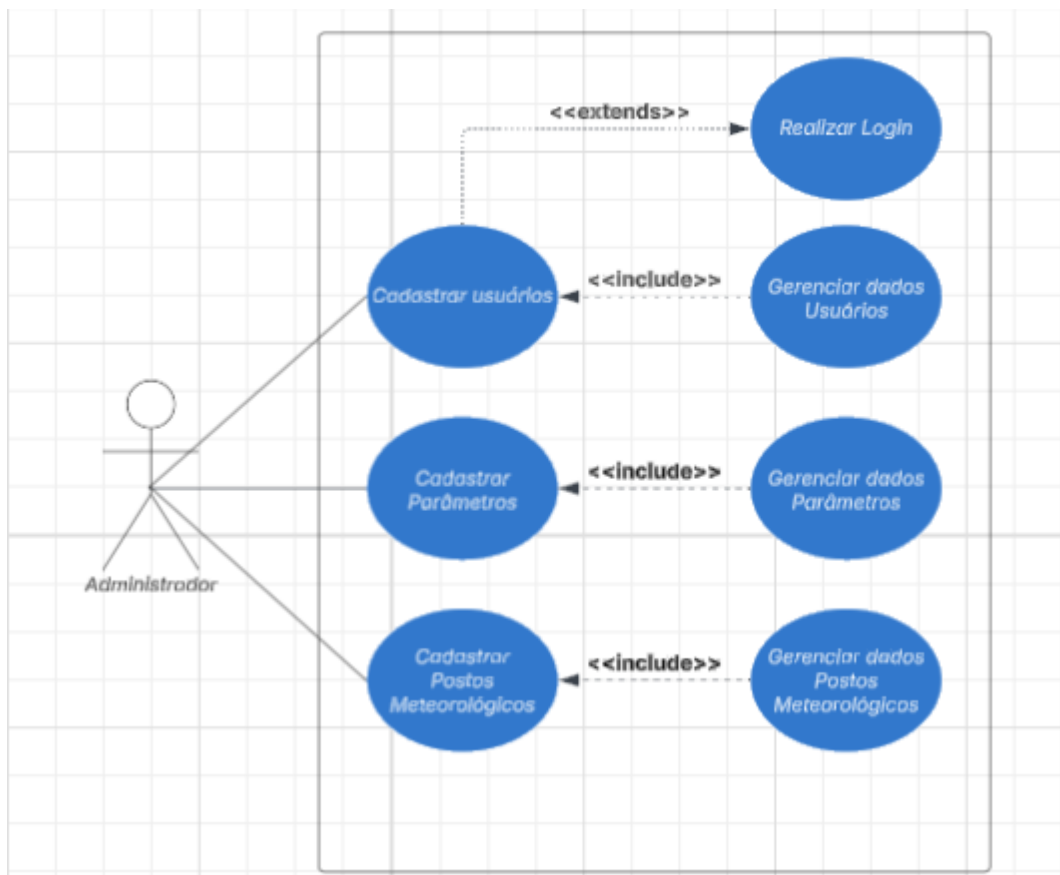


Fonte: Autores, (2025)

### 3.2.2 Diagrama de Caso de Uso do Administrador

O diagrama com o Administrador como ator, mostrado na Figura 3 abaixo, o Administrador pode realizar o *login* e gerenciar os usuários, parâmetros e postos.

Figura 3 - Diagrama do Caso de Uso do Administrador.



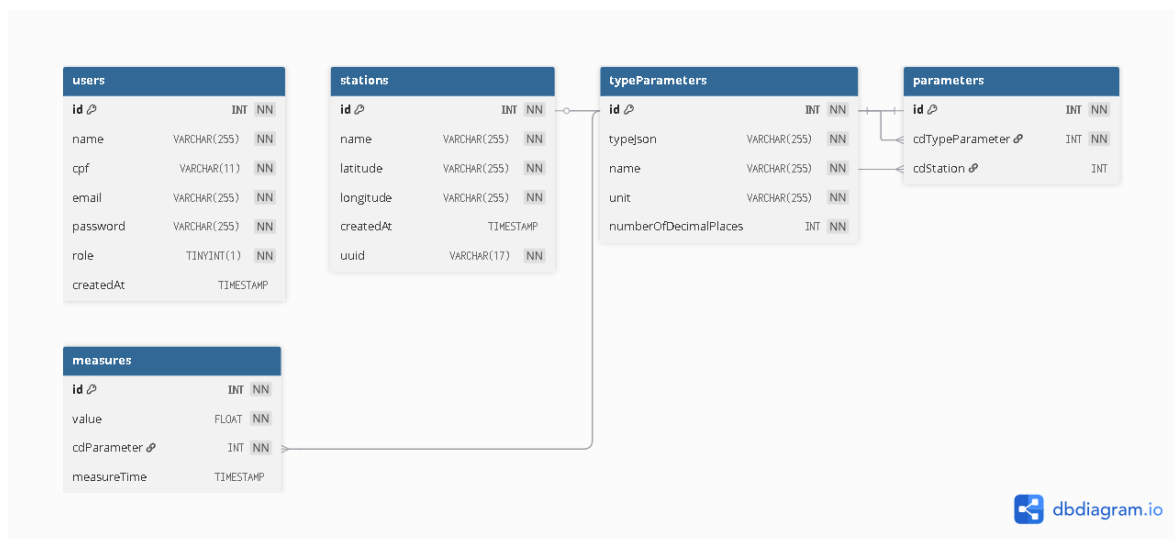
Fonte: Autores, (2025)

### 3.2.3 Modelo Lógico

Para mapear e organizar os dados, foi utilizado o modelo lógico, no qual são verificadas as funcionalidades contidas no projeto, para assim poder avaliar a sua viabilidade. Essa etapa é de suma importância, pois a partir pode-se definir as entidades, atributos e relacionamento, através dela também é definido as chaves primárias e estrangeiras, além do tratamento adequado das ligações entre entidades.

No Modelo Lógico, apresentado na figura 4, o Banco de Dados foi representado pelo Diagrama Entidade-Relacionamento, especificando como as entidades se relacionam e auxiliando na construção do modelo físico do Banco de Dados. Na figura abaixo, o Diagrama ilustra o relacionamento entre o Sistema e o Administrador e as principais funcionalidades do sistema.

Figura 4 - Diagrama do Banco de Dados.



Fonte: Autores, (2025)

## 4 RESULTADOS E DISCUSSÃO

Neste capítulo serão apresentados os resultados obtidos e a discussão em nosso projeto RainTrack.

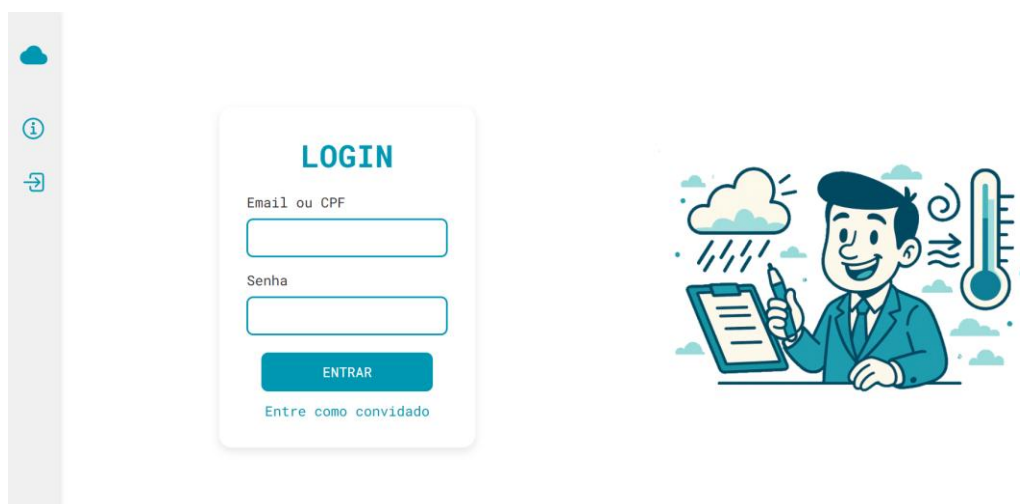
### 4.1 Protótipo do *Site*

Neste tópico iremos apresentar o nosso site e o *hardware* do projeto, mostrando tela por tela e explicando suas funcionalidades. Dos tópicos 4.1.1 até o 4.1.4 e o 4.1.14 as telas apresentadas podem ser acessadas por todos os usuários comuns e do 4.1.5 até o 4.1.13 apenas os administradores podem visualizar.

#### 4.1.1 Tela de *Login*

A tela de *login*, apresentada na figura 5, é também a tela *index* do nosso site, sendo a primeira a aparecer quando entramos nele. A partir dela podemos realizar a entrada como usuário comum ou administrador do sistema, também podemos entrar como convidados, mas isso delimita as ações que se pode fazer.

Figura 5 – Tela de Login.



Fonte: Autores, (2025)

### 4.1.2 Tela Sobre Nós

Na tela sobre nós, apresentada na figura 6 A e B, é encontrado uma breve descrição sobre nosso projeto e nossos ideais.

Figura 6 – Tela Sobre Nós A e B

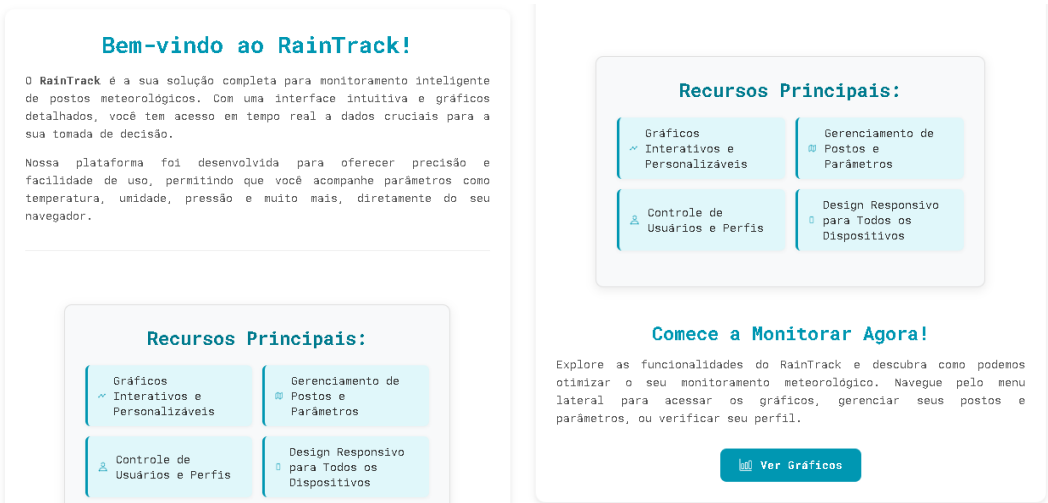


Fonte: Autores, (2025)

### 4.1.3 Tela de Boas-Vindas

A tela de boas-vindas, apresentada na figura 7 A e B, é a primeira que aparece após realizar o *login* ou entrar como convidado, nela é encontrada informações sobre o site, nossos principais recursos e por fim uma área sobre o monitoramento.

Figura 7 – Tela de Boas-Vindas A e B



Fonte: Autores, (2025)

#### 4.1.4 Tela de Gráficos

A tela de gráficos, apresentada nas figuras 8 (sem gráfico) e 9 (com gráfico), é onde é apresentado o gráfico dos dados meteorológicos de cada posto, sendo possível também fazer um filtro para um período específico.

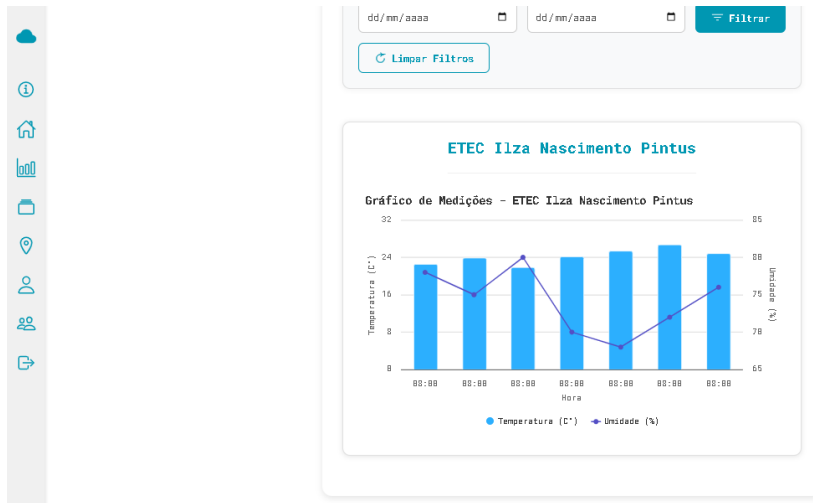
Figura 8 - Tela de Gráficos (sem gráfico).



Fonte: Autores, (2025)



Figura 9 – Tela de Gráficos (com gráfico).



Fonte: Autores, (2025)

#### 4.1.5 Tela de Parâmetros

Na tela de parâmetros, apresentada na figura 10, é encontrado os parâmetros já cadastrados e também o *link* para a página de cadastro de novos parâmetros.

Figura 10 – Tela de Parâmetros.

**Parâmetros Cadastrados**

**Temperatura**

Unidade: C°

Tipo: number

Casas Decimais: 2

**Umidade**

Unidade: %

Tipo: number

Casas Decimais: 2

[+ ADICIONAR NOVO PARÂMETRO](#)

[Voltar para o Início](#)

Fonte: Autores, (2025)

#### 4.1.6 Tela de Cadastro de Parâmetros

Na tela de cadastro de parâmetros, apresentada na figura 11, podemos cadastrar os parâmetros que os postos podem receber, para o cadastro é necessário: seu nome, a sua unidade de medida, o tipo JSON e a quantidade de casas decimais.

Figura 11 – Tela de Cadastro de Parâmetros.



A interface de usuário para o cadastro de novos parâmetros. À esquerda, uma barra lateral vertical contém ícones para: nuvem, informações, casa, gráfico de barras, calendário, localização, perfil de usuário, grupo de usuários e uma seta para cima. O formulário principal, intitulado "Cadastrar Novo Parâmetro", possui os seguintes campos de entrada: "Nome do Parâmetro", "Unidade", "Tipo JSON" e "Casas decimais". Abaixo dos campos, há um botão azul com o texto "CADASTRAR PARÂMETRO" e um link azul com o texto "← Voltar para Parâmetros".

Fonte: Autores, (2025)

#### 4.1.7 Tela de Postos

Na tela de postos, apresentada na figura 12, é mostrado os postos meteorológicos já cadastradas e também o *link* para a tela de cadastro de novos postos.

Figura 12 – Tela de Postos.

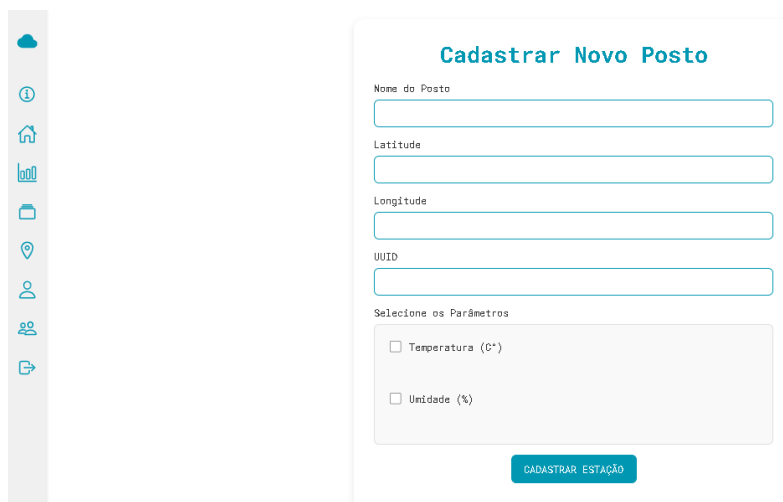


Fonte: Autores, (2025)

#### 4.1.8 Tela de Cadastro de Postos

Na tela de cadastro de postos, apresentada na figura 13, podemos cadastrar os postos meteorológicos, para o cadastro é necessário: seu nome, a latitude, a longitude, seu UUID e os parâmetros que ela vai medir.

Figura 13 – Tela de Cadastro de Postos.



Fonte: Autores, (2025)

#### 4.1.9 Tela de Cadastro de Usuários

Na tela de cadastro de usuários, apresentada na figura 14, é realizado o cadastro de novos usuários, para isso é necessário: seu nome, email, CPF, a sua senha e o tipo de usuário (comum ou administrador).

Figura 14 – Tela de Cadastro de Usuários.

Bem-vindo, Admin (Admin)

Cadastrar novo usuário

Nome

Email

CPF

Senha

Tipo de usuário

Usuário comum

CADASTRAR

[Ir para Início](#)

Fonte: Autores, (2025)

#### 4.1.10 Tela de Usuários

Na tela de usuários, apresentada na figura 15, encontramos os usuários já cadastrados e suas informações, o link para a página de edição e para página de cadastro desses usuários e podemos realizar a exclusão do usuário.

Figura 15 – Tela de Usuários.



Fonte: Autores, (2025)

#### 4.1.11 Tela de Edição de Usuários

Na tela de edição de usuários, apresentada na figura 16, é possível alterar as informações cadastradas de um usuário.

Figura 16 – Tela de Edição de Usuários.

A interface 'Editar Usuário' contém os seguintes campos de entrada:

- Nome:
- Email:
- CPF:
- Nova Senha (deixe em branco para não alterar):
- Nível de Acesso:

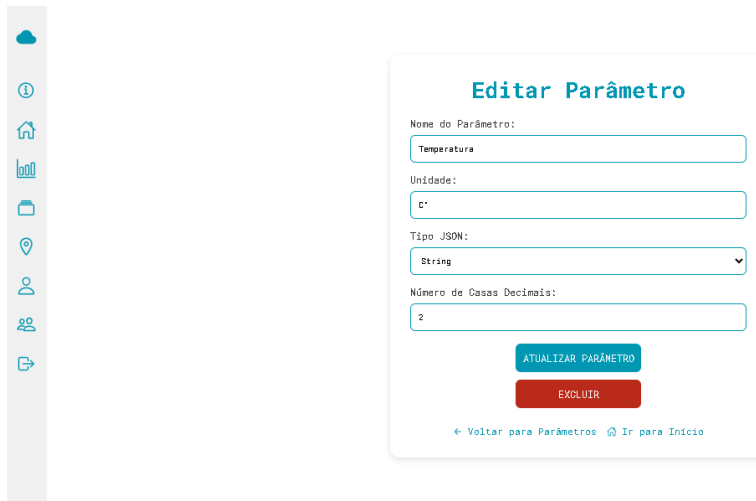
Abaixo dos campos, há um botão 'ATUALIZAR USUÁRIO' e dois links de navegação: 'Voltar para Usuários' e 'Ir para Início'.

Fonte: Autores, (2025)

#### 4.1.12 Tela de Edição de Parâmetros

Na tela de edição de parâmetros, apresentada na figura 17, é possível alterar as informações cadastradas de um parâmetro.

Figura 17 – Tela de Edição de Parâmetros.



A imagem mostra a interface de usuário para a edição de parâmetros. À esquerda, há uma barra lateral vertical com ícones para navegação: uma nuvem, um relógio, uma casa, um gráfico de barras, um documento, um ponto de localização, um usuário, um grupo de pessoas e uma seta para fora. O formulário principal, intitulado "Editar Parâmetro", contém os seguintes campos: "Nome do Parâmetro:" com o valor "Temperatura"; "Unidade:" com o valor "C°"; "Tipo JSON:" com um menu suspenso selecionando "String"; e "Número de Casas Decimais:" com o valor "2". Abaixo dos campos, há dois botões: "ATUALIZAR PARÂMETRO" em azul e "EXCLUIR" em vermelho. Na base do formulário, há links de navegação: "← Voltar para Parâmetros" e "🏠 Ir para Início".

Fonte: Autores, (2025)

#### 4.1.13 Tela de Edição de Postos

Na tela de edição de postos, apresentada na figura 18, é possível alterar as informações cadastradas de um posto.

Figura 18 – Tela de Edição de Postos.

**Editar Posto**

Nome do Posto:  
ETEC Ilza Nascimento Pintos

Latitude:  
111111111111111111

Longitude:  
222222222222222222

UUID:  
1

Parâmetros Associados:

- ☒ Temperatura (C°)
- ☒ Umidade (%)

ATUALIZAR ESTAÇÃO

EXCLUIR

[Voltar para Postos](#) [Ir para Início](#)

Fonte: Autores (2025)

#### 4.1.14 Tela Meu Perfil

Na tela meu perfil, apresentada na figura 19, o usuário comum pode verificar suas informações.

Figura 19 – Tela Meu Perfil.

**Meu Perfil**

**Gustavo da Costa Gomes**  
Usuário

Email: gustavo@gmail.com

CPF: 00000000002

Membro desde: 19/09/2025

ID do Usuário: #3

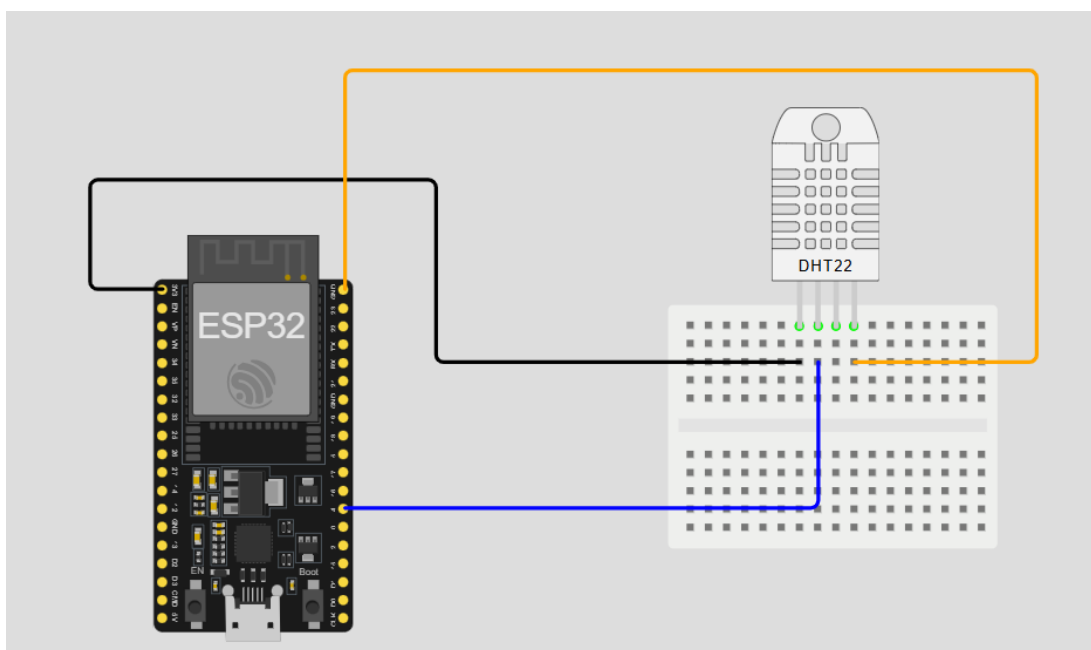
[Voltar ao Início](#) [Sair da Conta](#)

Fonte: Autores, (2025)

## 4.2 Hardware do Projeto

Este é o hardware do nosso projeto, apresentado na figura 20, utilizamos o site Wokwi para sua prototipação. Nele contém um microcontrolador ESP32, o sensor DHT22 que é semelhante ao que foi utilizado no projeto e uma *protoboard*.

Figura 20 – Protótipo do Hardware.



Fonte: Autores, (2025)

## 4.3 Análise dos Resultados

O projeto *RainTrack* foi concluído com êxito, alcançando seus objetivos, sendo eles: fazer um posto meteorológico utilizando um microcontrolador ESP32 junto de alguns sensores que faça a coleta dos dados meteorológicos, a passagem desses dados coletados via Protocolo MQTT, o armazenamento desses dados em um banco de dados MySQL, uma *dashboard* utilizando o Python Flask para o *backend* e o HTML, CSS e JS para o *frontend* onde são disponibilizados os gráficos interativos feitos através do HighCharts que contém os dados armazenados no banco, dentro dessa dashboard os administradores do sistema conseguem fazer o cadastro de novos usuários, parâmetros e postos meteorológicos.

O principal problema encontrado durante o desenvolvimento foi a dificuldade de comunicação com nosso *stakeholder*, com grande demora para responder, assim gerando um atraso para a conclusão dos processos pela falta de *feedback*.



## 5 CONCLUSÃO

O desenvolvimento do sistema *RainTrack* teve como propósito criar uma solução inteligente e acessível para o monitoramento meteorológico em tempo real, contribuindo para a prevenção de desastres naturais e para o planejamento urbano. Durante o processo, todos os objetivos específicos estabelecidos foram alcançados de forma satisfatória.

Inicialmente, foi implementado um servidor Flask integrado a um banco de dados MySQL, responsável pelo armazenamento e gerenciamento seguro dos dados meteorológicos. Em seguida, foi desenvolvida a comunicação entre o ESP32 e o sistema, utilizando o protocolo MQTT, o que garantiu a transmissão estável e contínua das informações coletadas pelos sensores.

O projeto também contemplou a criação de um dashboard interativo, desenvolvido com HTML, CSS, JS e Highcharts, permitindo a visualização clara e dinâmica dos dados em tempo real. Além disso, foram aplicadas boas práticas para assegurar a confiabilidade e integridade das informações, reduzindo perdas e aumentando a precisão das medições.

Outro marco importante foi a realização do *deploy* da aplicação na plataforma PythonAnywhere, o que possibilitou a disponibilização online do sistema, ampliando seu alcance e facilitando o acesso remoto tanto para usuários comuns quanto para administradores.

Portanto, o *RainTrack* atingiu plenamente seus objetivos, demonstrando viabilidade técnica, eficiência na coleta e visualização de dados e potencial para expansão futura. O projeto se apresenta como uma ferramenta útil e escalável, com grande aplicabilidade em contextos educacionais, ambientais e de segurança pública.

## REFERÊNCIAS

APACHE FRIENDS. XAMPP documentation. Disponível em: <https://www.apachefriends.org/docs>. Acesso em: 23 set. 2025.

BRASIL ESCOLA. Desastres ambientais. Disponível em: <https://brasilecola.uol.com.br/geografia/desastres-ambientais.htm>. Acesso em: 29 mar. 2025.

ECLIPSE FOUNDATION. Eclipse Mosquitto – An open source MQTT broker. Disponível em: <https://mosquitto.org>. Acesso em: 23 set. 2025.

EOS DATA ANALYTICS. Desastres naturais 2023. Disponível em: <https://eos.com/pt/blog/desastres-naturais-2023>. Acesso em: 29 mar. 2025.

ESPRESSIF. ESP32 Series Datasheet. Disponível em: <https://www.espressif.com/en/products/socs/esp32>. Acesso em: 23 set. 2025.

FOLHA DE S. PAULO. Desastres naturais ao redor do mundo causaram mais de US\$ 250 bilhões em prejuízos em 2023. Disponível em: <https://www1.folha.uol.com.br/ambiente/2024/01/desastres-naturais-ao-redor-do-mundo-causaram-mais-de-us-250-bilhoes-em-prejuizos-em-2023.shtml>. Acesso em: 29 mar. 2025.

GIT-SCM. Documentação oficial do Git. Disponível em: <https://git-scm.com/doc>. Acesso em: 23 set. 2025.

HIGHCHARTS. Official documentation. Disponível em: <https://www.highcharts.com/docs>. Acesso em: 23 set. 2025.

MOZILLA DEVELOPER NETWORK (MDN). JavaScript documentation. Disponível em: <https://developer.mozilla.org/docs/Web/JavaScript>. Acesso em: 23 set. 2025.

MYSQL. Documentação oficial do MySQL. Disponível em: <https://dev.mysql.com/doc>. Acesso em: 23 set. 2025.

MUNDO EDUCAÇÃO. O que é meteorologia? Disponível em: <https://mundoeducacao.uol.com.br/geografia/o-que-e-meteorologia.htm>. Acesso em: 29 mar. 2025.

OASIS. MQTT Version 3.1.1 Plus Errata 01. Disponível em: <https://docs.oasis-open.org/mqtt/mqtt/v3.1.1>. Acesso em: 23 set. 2025.

PALLETS PROJECT. Flask documentation. Disponível em: <https://flask.palletsprojects.com>. Acesso em: 23 set. 2025.

PLUGFIELD. Saiba como o monitoramento climático atua na sustentabilidade e prevenção de desastres naturais. Disponível em: <https://portal.plugfield.com.br/saiba-como-o-monitoramento-climatico-atua-na-sustentabilidade-e-prevencao-de-desastres-naturais>. Acesso em: 29 mar. 2025.

PYTHON SOFTWARE FOUNDATION. Python documentation. Disponível em: <https://docs.python.org/3/>. Acesso em: 23 set. 2025.

TRELLO. O que é o Trello? Disponível em: <https://trello.com/pt-BR/guide>. Acesso em: 23 set. 2025.

VISUAL STUDIO CODE. Documentation. Disponível em: <https://code.visualstudio.com/docs>. Acesso em: 23 set. 2025.

W3C. Cascading Style Sheets (CSS). Disponível em: <https://www.w3.org/Style/CSS>. Acesso em: 23 set. 2025.

WHATWG. HTML Living Standard. Disponível em: <https://html.spec.whatwg.org>. Acesso em: 23 set. 2025.