Estação de irrigação com monitoramento solo/ar e gerenciamento via Aplicativo

Emanuel Fagan Bissacotti, Alessandro André Mainardi De Oliveira
Curso de Ciência da Computação
Universidade Franciscana - UFN
Santa Maria - RS
e.bissacotti@ufn.edu.br, alessandroandre@ufn.edu.br

***Resumo*—Este trabalho apresenta o desenvolvimento de uma estação de irrigação automatizada com monitoramento ambiental e gerenciamento remoto por meio de um aplicativo móvel, voltada para pequenas plantas em ambientes domésticos. A solução integra conceitos de Internet das Coisas (IoT) e automação, utilizando o microcontrolador ESP32 e os sensores de temperatura e umidade do ar (DHT11) e umidade do solo (FC-28). A comunicação entre hardware e aplicativo foi implementada por meio do protocolo de comunicação MQTT, enquanto o armazenamento e a autenticação dos dados foram realizados no Firebase. O desenvolvimento seguiu a metodologia ágil Kanban. Nos testes e resultados realizados, o sistema apresentou estabilidade na comunicação e correlação de Pearson superior a 0,9 entre sensores idênticos, validando a confiabilidade das medições. A função agendada do Firebase executou leituras e verificações para irrigação de forma automática a cada hora, com acerto total em cinco dias de testes com uma planta de manjericão, mantendo a umidade do solo dentro da faixa ideal (55–75%).**

***Palavras-chave*: IoT; Sensores; Automação; MQTT; Firebase**

I. Introdução

O aquecimento global elevou a temperatura média do planeta em cerca de 1,2°C desde a era pré-industrial (1850-1900), com projeções indicando um aumento contínuo caso as emissões de gases de efeito estufa não sejam reduzidas [1]. Esse aumento de temperatura intensifica a evapotranspiração, elevando a demanda hídrica, especialmente em mudas e pequenas plantas, que se tornam mais vulneráveis à escassez de água durante suas fases iniciais de desenvolvimento [2].

A implementação de sistemas baseados em Internet das Coisas (*Internet of Things - IoT)* e automação surge como uma solução promissora para enfrentar esses desafios. Tais sistemas permitem o monitoramento contínuo de variáveis ambientais, como temperatura, umidade do ar e umidade do solo, possibilitando ajustes precisos na irrigação e reduzindo o desperdício hídrico. Além disso, a automação de tarefas cotidianas, como o manejo de irrigação, não apenas melhora a eficiência no cultivo de plantas, mas também contribui para a qualidade de vida, ao liberar tempo e recursos para os usuários. Estudos recentes apontam que a integração de tecnologias IoT no setor agrícola pode aumentar a produtividade e a resiliência das plantas, ao mesmo tempo em que promove práticas mais sustentáveis [3].

A relevância deste trabalho está no desenvolvimento de um sistema completo que contribui para o manejo eficiente da irrigação e o monitoramento de variáveis ambientais em pequena escala. Diferentemente de soluções voltadas para grandes cultivos, este projeto foca em atender às necessidades de pequenos produtores e entusiastas da jardinagem, oferecendo uma ferramenta acessível e tecnológica para diminuir os impactos das mudanças climáticas no cultivo de plantas.

A. Objetivos

1) Objetivo geral: Desenvolver uma solução embarcada e um aplicativo móvel para automação da irrigação e monitoramento de temperatura, umidade do ar e umidade do solo.

2) Objetivos específicos:

- Desenvolver a automação no manejo de irrigação para pequenas plantas domésticas auxiliando no seu desenvolvimento;
- Utilizar o protocolo MQTT (*Message Queuing Telemetry Transport*) para comunicação aplicativo-hardware e hardware-aplicativo e utilizar o Firebase para persistência dos dados;
- Visualizar dados de temperatura, umidade do ar e umidade do solo via aplicativo;
- Visualizar os dados em tempo de execução ou gráficos de linhas com os dados históricos no aplicativo;
- Realizados testes em cenário real verificando a umidade do ar, do solo, temperatura ambiente e irrigando uma planta doméstica;

B. Estrutura do trabalho

Nas próximas seções deste trabalho, é apresentado o Referencial Teórico, apresentando conceitos fundamentais e as tecnologias empregadas no desenvolvimento do projeto. A seção de Trabalhos Correlatos analisa soluções existentes na literatura, destacando suas limitações e as contribuições do presente trabalho. Em seguida, a seção de Metodologia detalha o processo de desenvolvimento e em qual local as ferramentas adotadas são utilizadas no sistema. A seção de Resultados apresenta os principais avanços obtidos. Por fim, a Conclusão resume os pontos principais do trabalho.

II. Referencial Teórico

O manejo eficiente de recursos hídricos em pequena escala enfrenta desafios crescentes diante das mudanças climáticas, demandando soluções tecnológicas que combinem automação para otimizar processos como a irrigação de pequenas plantas. Este trabalho fundamenta-se em conceitos-chave de *IoT*, aplicados ao monitoramento contínuo de variáveis ambientais como temperatura, umidade do ar e umidade do solo e ao controle preciso do fluxo de água.

A. Internet das Coisas

A Internet das Coisas refere-se a uma rede de dispositivos físicos e objetos interconectados, equipados com sensores, atuadores e tecnologias de comunicação, que permitem a coleta, o processamento e a transmissão de dados por meio da internet sem intervenção humana direta [4]. Esses dispositivos, que abrangem desde eletrodomésticos até sistemas industriais, integram o mundo físico ao digital, possibilitando monitoramento remoto, automação e tomada de decisão em tempo de execução. Em aplicações como sistemas de irrigação inteligente, a IoT conecta sensores de umidade do solo e válvulas a plataformas de controle, otimizando o uso de recursos hídricos e promovendo eficiência energética [5]. Sistemas IoT representam uma evolução significativa na integração de tecnologias embarcadas e redes, com impacto crescente na automação em diversos setores.

B. Automação na Agricultura

De acordo com a Embrapa, a automação na agricultura pode ser implementada tanto nos processos operacionais quanto no monitoramento ambiental, sendo controlada por meio de máquinas, dispositivos eletrônicos ou sistemas computacionais [6]. Essa abordagem amplia significativamente a capacidade do trabalho humano, permitindo a execução de tarefas complexas com maior precisão e menor esforço. A automação promove a eficiência no processo de produção agrícola ao elevar o desempenho qualitativo e quantitativo, reduzindo perdas e otimizando recursos, como água e insumos, de forma expressiva [6].

Um destaque significativo na automação agrícola é a irrigação automatizada, que integra tecnologias como sensores de umidade do solo, microcontroladores e sistemas de comunicação remota [7]. Esses sistemas monitoram condições ambientais, ajustando a irrigação às necessidades das culturas, o que pode promover economia de água e aumentar a produtividade.

C. Sensores e Atuadores

Sensores são dispositivos que detectam ou medem mudanças no ambiente, convertendo essas variações físicas em sinais analógicos ou digitais. Funcionam como uma interface entre o mundo físico e o digital, permitindo que sistemas eletrônicos coletem dados para monitoramento ou controle [8].

Atuadores, por sua vez, são componentes que transformam sinais elétricos em ações físicas, como movimento, força ou alterações no ambiente. Eles complementam os sensores ao executar comandos gerados por sistemas de controle, possibilitando a interação ativa com o meio físico [8].

Assim, sensores e atuadores formam uma relação essencial para a automação e eficiência em sistemas onde os sensores captam informações do ambiente e os atuadores acionam dispositivos que podem controlar o ambiente.

1) Sensor DHT11: O sensor DHT11 é um sensor de temperatura e umidade do ar com um sinal digital em sua saída. Possui uma faixa de temperatura recomendada de -20 a 60 ºC com uma faixa de erro de ±2 °C e uma faixa de umidade recomendada de 5 a 95% com uma faixa de erro de 5%. [9]

2) Sensor de umidade do solo FC-28: O sensor FC-28 é um sensor de umidade do solo que utiliza o princípio da resistividade elétrica para medir o teor de água no solo. Ele é composto por duas sondas metálicas que detectam a condutividade elétrica entre elas a qual varia conforme o teor de água no solo: solos mais úmidos apresentam menor resistência elétrica, resultando em um sinal analógico de menor valor, enquanto solos secos exibem maior resistência, gerando um sinal maior. Uma limitação é a possível corrosão a longo prazo. [10]

3) Válvula solenoide: A válvula solenoide é um dispositivo eletromecânico utilizado para controlar o fluxo de fluidos, como líquidos ou gases, por meio de um atuador eletromagnético conhecido como solenoide. O solenoide consiste em uma bobina de fio condutor, enrolada em torno de um núcleo ferromagnético móvel (plunger ou pistão), que se desloca quando a bobina é energizada por uma corrente elétrica, criando um campo magnético. Esse movimento abre ou fecha um orifício na válvula, regulando o fluxo do fluido de maneira precisa e automática [11].

A válvula solenoide pode ser configurada como normalmente fechada (NF), mantendo o fluxo interrompido até que a bobina seja energizada para abri-lo, ou normalmente aberta (NA), permitindo o fluxo até que a energização o interrompa. Para este trabalho, optou-se por uma válvula solenoide normalmente fechada, de modo que o fluxo seja liberado apenas quando a bobina for ativada, minimizando o consumo energético, já que a válvula permanecerá fechada na ausência de corrente e evitando o fluxo indesejado de água quando o sistema está desligado [11].

4) Relé: É um dispositivo eletrônico utilizado como um interruptor controlado por uma corrente elétrica, permitindo que circuitos de baixa potência controlem circuitos de alta potência de forma segura e eficiente. O funcionamento básico de um relé envolve uma bobina que, ao receber uma tensão, gera um campo magnético que atrai um conjunto de contatos mecânicos, alterando seu estado entre normalmente aberto e normalmente fechado.[12]

D. Conversor 7805CT

O MC7805CT é um regulador de tensão linear positivo da série MC7800, projetado para fornecer uma saída fixa de 5 V com corrente de até 1 A. Ele opera com uma tensão de entrada tipicamente entre 7 V e 20 V, sendo necessário um mínimo de 2 V a mais do que a saída para garantir a regulação adequada.[13]

E. ESP32

O ESP32, desenvolvido pela Espressif Systems, é um microcontrolador de baixo custo e alta eficiência energética, amplamente utilizado em aplicações de Internet das Coisas (IoT). Destaca-se por seu baixo consumo de energia, graças ao coprocessador de ultrabaixo consumo (ULP). O ESP32 oferece 520 KB de RAM interna para dados e instruções, além de suporte a até 16 MB de memória Flash. Possui conectividade Wi-Fi e Bluetooth, 34 pinos GPIO (General-Purpose Input/Output em português Entrada/saída de uso geral) configuráveis, tornando-o uma solução versátil para sistemas embarcados [14].

F. C++

Microcontroladores como ESP32 utilizam em uma variante simplificada da linguagem C++, adaptada para sistemas embarcados. Essa versão preserva a sintaxe orientada a objetos do C++, permitindo o uso de classes e métodos, mas é limitada por restrições de hardware, como memória reduzida e ausência de sistema operacional completo [15].

G. Dart e Flutter

Dart é uma linguagem de programação de código aberto desenvolvida pela Google, caracterizada por sua tipagem forte opcional e suporte a compilações Just-In-Time (JIT) e Ahead-Of-Time (AOT), que favorecem o desenvolvimento rápido e desempenho otimizado [16]. Flutter, por sua vez, é um framework de desenvolvimento de interfaces baseado em Dart, também criado pela Google, que permite a construção de aplicações nativas para Android, iOS, web e desktop a partir de uma única base de código. Utiliza um motor de renderização próprio baseado em Skia[1], oferecendo alta performance e personalização visual por meio de widgets[2], o que o torna ideal para projetos que exigem interfaces responsivas e consistentes [17].

H. Firebase

O Firebase é uma plataforma de desenvolvimento de aplicações, atualmente mantida pelo Google, que oferece um conjunto integrado de ferramentas e serviços de backend. Disponibiliza funcionalidades como bancos de dados no SQL, autenticações para login, hospedagem de serviços, entre outros serviços de backend.[18]

[1]Skia é uma biblioteca de gráficos 2D de código aberto que permite criar gráficos de alta qualidade em aplicativos

[2]Um *widget* é um componente gráfico que facilita o acesso rápido a funcionalidades específicas.

1) Firebase Functions: O Firebase Functions é um framework serverless que permite o desenvolvimento de lógica de backend em linguagens como JavaScript, TypeScript ou Python. Por meio desse serviço, os desenvolvedores podem criar funções personalizadas que são implantadas diretamente na infraestrutura de nuvem do Google com o comando `Firebase deploy`. Após a implantação, o código torna-se acessível globalmente, eliminando a necessidade de gerenciar servidores próprios e oferecendo escalabilidade automática [19].

2) Firebase Authentication: O Firebase Authentication é um serviço de backend fornecido pelo Firebase que facilita a autenticação de usuários em aplicações, oferecendo suporte ao cadastro, login e gerenciamento de contas. Integra-se a provedores de terceiros, como Facebook, GitHub, Apple, Google e X, permitindo autenticação simplificada por meio de credenciais externas [20]. Além disso, disponibiliza métodos prontos para funcionalidades como alteração de senha, recuperação de acesso e desativação de contas, eliminando a necessidade de implementação manual dessas rotinas.

3) Firebase Firestore: O Firebase Firestore é um banco de dados flexível e escalonável para desenvolvimento focado em dispositivos móveis e Web. Ele mantém seus dados em sincronia em aplicativos cliente usando listeners em tempo de execução do sistema. Além disso, oferece suporte off-line para dispositivos móveis e Web para que o desenvolvedor possa criar aplicativos responsivos que funcionem independentemente da estabilidade da rede ou da conectividade com a Internet [21].

I. Javascript e Typescript

JavaScript é uma linguagem de programação interpretada, dinâmica e de alto nível, originalmente criada para adicionar interatividade a páginas web no lado do cliente, mas que evoluiu para uma solução versátil tanto no cliente quanto no servidor. Destaca-se por sua flexibilidade, suportando paradigmas funcional, imperativo e orientado a objetos, embora sua tipagem fraca possa levar a erros em projetos complexos. TypeScript, desenvolvido pela Microsoft, é um superconjunto de JavaScript que introduz tipagem estática opcional e recursos avançados, como interfaces e módulos, compilando para JavaScript puro para garantir compatibilidade universal [22].

J. Git e GitHub

Git é um sistema de controle de versão de código, projetado para rastrear alterações em arquivos e facilitar a colaboração em projetos de desenvolvimento de software. O Git permite que cada desenvolvedor mantenha uma cópia completa do repositório, incluindo todo o histórico de mudanças, o que agiliza operações como commits, branches e merges, além de suportar fluxos de trabalho não lineares. [23]

O GitHub, por sua vez, é uma plataforma baseada em nuvem que utiliza o Git como fundamento, oferecendo ferramentas adicionais para hospedagem de repositórios, colaboração em equipe e revisão de código por meio de pull requests [23]. Além disso, disponibiliza recursos como a vinculação de repositórios em projetos, centralizando requisitos e possibilitando a visualização de quadros Kanban para acompanhar tarefas concluídas, em desenvolvimento ou no backlog.

K. Protocolo MQTT

O MQTT (*Message Queuing Telemetry Transport*) é um protocolo de mensagens leve e baseado no modelo *publish/subscribe* ou pub/sub, projetado para comunicação eficiente entre dispositivos em redes com largura de banda limitada ou condições instáveis, sendo amplamente adotado em aplicações de IoT. Operando sobre TCP/IP, utiliza um software chamado *broker* central para gerenciar a troca de mensagens entre clientes que publicam e assinam tópicos, promovendo escalabilidade [24].

L. Kanban

De acordo com Pressman, o Kanban é uma metodologia que utiliza princípios visuais para gerenciar o progresso de tarefas, promovendo a transparência e a melhoria contínua em qualquer tipo de processo ou fluxo de trabalho [25]. O método se baseia em um quadro visual, geralmente dividido em no mínimo três colunas — "A Fazer", "Fazendo"e "Feito"—, que representam os estágios de desenvolvimento de cada elemento ou funcionalidade do software. À medida que o projeto avança, as tarefas são movidas de uma coluna para a próxima, permitindo que a equipe visualize o progresso, identifique gargalos e ajuste o fluxo de trabalho de forma iterativa [25]. Além disso, o Kanban enfatiza a limitação do trabalho em progresso (WIP, do inglês *Work in Progress*), garantindo que a equipe se concentre em concluir tarefas antes de iniciar novas, o que contribui para a entrega contínua e a redução de desperdícios no processo de desenvolvimento.

M. Estatísticas

1) Erro médio absoluto (Mean Absolute Error - MAE): Mede o erro médio entre os valores observados e comparados, considerando a diferença absoluta ponto a ponto. Valores próximos de 0 indicam maior proximidade entre as medições [26].

2) Raiz do erro quadrático médio (Root Mean Squared Error - RMSE): Também chamada de Erro Quadrático Médio, avalia a dispersão dos erros, atribuindo maior peso a desvios maiores. Valores próximos de 0 indicam baixa variabilidade e boa consistência entre os dados [27].

3) Correlação de Pearson: Mede o grau de relação linear entre duas variáveis, variando de –1 a 1. Valores próximos de 1 indicam forte correlação e comportamento semelhante entre as séries de dados comparadas [28].

III. Trabalhos Correlatos

Nesta seção são apresentados os trabalhos científicos correlacionados a este, de forma que auxiliaram no desenvolvimento da ideia do projeto, trazendo informações sobre sistemas IoT de irrigação.

A. Design and realization of low-cost solenoid valve remotely controlled, application in irrigation network

O trabalho de Benbatouche e Kadri [29] apresenta o desenvolvimento de um sistema de irrigação automatizado e de baixo custo, composto por uma válvula solenóide projetada pelos autores e um módulo de controle remoto baseado em IoT. A solução utiliza uma Raspberry Pi 3 B+ para gerenciar as válvulas, permitindo operação via interface web (PHP/MySQL) ou por SMS através de um módulo GSM, o que amplia a acessibilidade em regiões com conectividade limitada.

O sistema integra sensores DHT11 e EK1099 para monitorar temperatura, umidade do ar e umidade do solo em tempo de execução, além de uma câmera para supervisão visual. Os resultados mostram que o controle remoto funcionou de forma estável, garantindo monitoramento confiável e consumo energético reduzido, reforçando a viabilidade de soluções econômicas para irrigação automatizada.

B. An internet of things based smart irrigation using solenoid valve

O artigo de Kannan et al. [30] apresenta um sistema de irrigação inteligente baseado em IoT voltado para reduzir o consumo de água em regiões com escassez hídrica. O sistema utiliza um microcontrolador PIC 16F877A para processar dados de sensores de umidade do solo, temperatura e umidade do ar, controlando automaticamente uma válvula solenóide acoplada a uma bomba de água. A comunicação com o agricultor ocorre via módulo GSM, que envia atualizações em tempo de execução sobre as condições monitoradas.

Os resultados indicam que a solução diminui significativamente o uso de água em comparação com métodos tradicionais, além de oferecer monitoramento contínuo exibido em LCD e acionamento eficiente da válvula conforme os dados coletados. O sistema se destaca por ser simples, econômico e adequado para áreas agrícolas extensas ou regiões com limitações hídricas, proporcionando economia de tempo, mão de obra e maior sustentabilidade.

C. Web-based Internet of Things on environmental and lighting control and monitoring system using node-RED, MQTT and Modbus communications within embedded Linux platform

O artigo de Chen et al. [31] apresenta um sistema IoT para monitoramento e controle ambiental e de iluminação utilizando uma Raspberry Pi 4B, integrando Node-RED com os protocolos MQTT e Modbus TCP. A solução combina

sensores de temperatura, umidade, CO_2 e luminosidade com atuadores como LEDs, cortinas elétricas e ar-condicionado, operando por meio de scripts Python multithread que realizam comunicação bidirecional entre PLCs e o broker MQTT. A interface web, acessível em múltiplas plataformas, permite supervisão e controle em tempo real, oferecendo uma alternativa de baixo custo para modernizar sistemas baseados em PLC.

Os testes demonstraram funcionamento estável, permitindo ajuste de iluminação, controle de dispositivos via GPIO e visualização contínua dos dados ambientais. A abordagem mostrou-se robusta e escalável, destacando o potencial da integração MQTT–Modbus e do uso de Node-RED para aplicações de automação em ambientes laboratoriais, industriais e educacionais.

De modo geral, os trabalhos analisados abordam diferentes soluções de automação aplicadas a sistemas agrícolas ou de controle ambiental, utilizando tecnologias como sensores, válvulas solenóides, microcontroladores e comunicação remota. Embora todos tenham a proposta de automatizar aos processos de monitoramento e controle, apresentam algumas limitações em termos de interatividade e escalabilidade. O projeto de Benbatouche, por exemplo, utiliza comunicação via SMS e interfaces web básicas, enquanto o trabalho de Kannan et al. também se restringe a mensagens SMS, sem oferecer uma interface de controle. Por outro lado, o sistema desenvolvido por Chen et al. explora de maneira eficaz a comunicação MQTT e o controle web, mas foca em ambientes laboratoriais com automação de iluminação e climatização.

IV. Metodologia

Para o desenvolvimento do projeto, adotou-se a metodologia ágil Kanban, devido à sua simplicidade e à eficiência no acompanhamento das atividades. O Kanban foi escolhido por permitir uma organização visual clara das etapas de desenvolvimento, facilitando o controle das tarefas relacionadas ao hardware, ao software embarcado e ao aplicativo móvel. Essa abordagem possibilitou uma gestão contínua das prioridades, garantindo que cada funcionalidade fosse concluída antes do início de uma nova, mantendo o fluxo de trabalho enxuto e bem estruturado ao longo de todo o projeto.

A. Proposta

O sistema proposto é uma estação de irrigação automatizada para pequenas plantas, integrado a um aplicativo móvel, com o objetivo de otimizar o uso de água por meio do monitoramento de dados ambientais e da automação do processo de irrigação. A solução utiliza o microcontrolador ESP32 para coletar dados dos sensores DHT11 (temperatura e umidade do ar) e FC-28 (umidade do solo), controlando uma válvula solenóide para a irrigação automática com base nos valores de umidade do solo. A comunicação entre o ESP32 e o aplicativo móvel é realizada via protocolo MQTT, com um *Broker* MQTT intermediando a troca de dados, enquanto a persistência de dados é feita pelo *Firebase* com o *Firestore*, permitindo a visualização de dados históricos.

A Figura 1 ilustra a arquitetura do sistema e a comunicação entre os dispositivos. No lado da estação de irrigação, o ESP32 coleta dados dos sensores DHT11 e FC-28 e controla a válvula solenóide, enviando os dados coletados ao *Broker* MQTT via Wi-Fi. O *Broker* MQTT também permite que o ESP32 receba comandos do aplicativo móvel e do *Firebase* via *Firebase Functions* para ativar ou desativar a irrigação e coletar os dados dos sensores. No lado do sistema móvel, o aplicativo se conecta ao *Broker* MQTT para receber os dados dos sensores e enviar comandos para acionar a válvula solenóide, além de interagir com o *Firebase* para autenticação com o *Firebase Auth* e armazenamento de dados e recuperação dos mesmos com o *Firebase Firestore*.

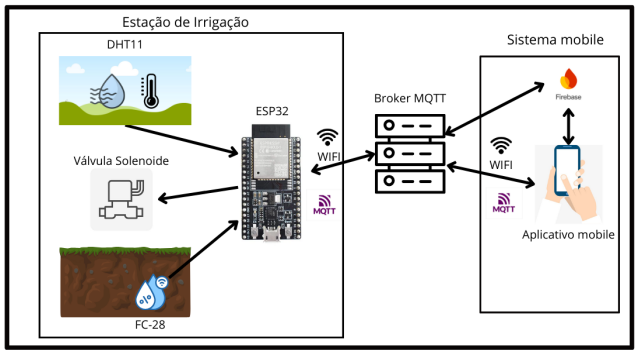

Figura 1. Comunicação entre as ferramentas e componentes

1) Projeto da Estação de irrigação: A Figura 2 ilustra o esquema do circuito eletrônico, destacando as conexões dos sensores e do relé ao ESP32, além dos componentes de suporte para o funcionamento do sistema.

O ESP32 é responsável por coordenar a leitura dos sensores e o controle da irrigação. No pino 12, está conectado o sensor DHT11 (Figura 2 letra A), que mede a temperatura e a umidade do ar. No pino 26, está conectado o sensor FC-28 (Figura 2 letra B), responsável por medir a umidade do solo. O FC-28 opera no modo analógico, então retorna valores dependendo da resistência captada por ele. Para converter esse valor em percentual de umidade do solo, aplica-se a seguinte equação:

$$\textit{porcentagem} = \frac{4095 - \texttt{valorSensor}}{4095 - \texttt{valorSoloSaturado}} \times 100$$

Nessa equação, `valorSensor` representa a leitura atual do sensor, `valorSoloSaturado` corresponde ao valor obtido durante a calibração quando o solo está completamente úmido (referência de 100%), e 4095 é o valor máximo possível do conversor analógico-digital do ESP32 (referência de solo seco). Dessa forma, o resultado expressa a umidade

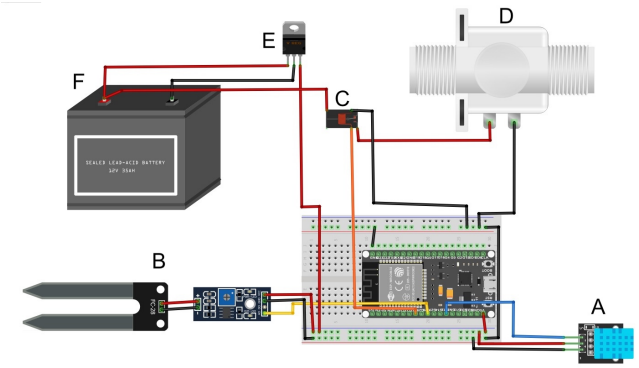

Figura 2. Circuito eletrônico estação de irrigação

do solo em porcentagem. Por fim, no pino 33, está conectado um módulo relé (Figura 2 letra C), que atua como uma chave eletrônica para controlar a válvula solenóide (Figura 2 letra D). Quando o ESP32 recebe do aplicativo móvel via MQTT que deve irrigar uma quantidade específica, ele ativa o relé mandando 3,3V, abrindo a válvula solenóide para iniciar a irrigação. O tempo necessário para irrigar é calculado por uma regra de três, conforme a equação abaixo:

$$segundosTotal = \frac{\texttt{irrigar}}{\texttt{vazaoSegundo}}$$

Aqui, `irrigar` é a quantidade de água desejada para irrigar (em mililitros), e `vazaoSegundo` é a vazão da válvula solenóide (em mililitro de água por segundo).

Para garantir o funcionamento estável dos componentes, o circuito inclui um regulador de tensão MC7805CT (Figura 2 letra E), que converte a tensão de entrada de uma fonte de energia 12v (Figura 2 letra F) para 5V, fornecendo energia ao ESP32 e o relé. O ESP32 utiliza sua conexão Wi-Fi para enviar os dados coletados ao *Broker MQTT*, que os disponibiliza ao aplicativo móvel, permitindo o monitoramento e controle dos dados.

Para representar o software embarcado que irá controlar os sensores, atuadores e o protocolo MQTT, foi desenvolvido um Diagrama de Atividade (Figura 3) para representar o funcionamento do mesmo. Ela apresenta que o processo inicia-se com a atividade de aguardar a conexão com o broker MQTT, fundamental para que o ESP32 possa receber comandos remotos do aplicativo móvel. Após a conexão, o sistema lê qual a função que ele precisa fazer, que pode ser "Irrigar"ou "Ler sensores".

Se o comando recebido for "Irrigar", o ESP32 realiza o cálculo do tempo de irrigação com base em parâmetros previamente definidos. Em seguida, aciona o relé, que por sua vez controla a válvula solenóide responsável por liberar a água. O sistema então aguarda o tempo calculado e, posteriormente, desativa o relé, encerrando o ciclo de irrigação.

Caso o comando recebido seja "Ler sensores", o ESP32 aciona a leitura dos sensores. O sensor DHT11 realiza a

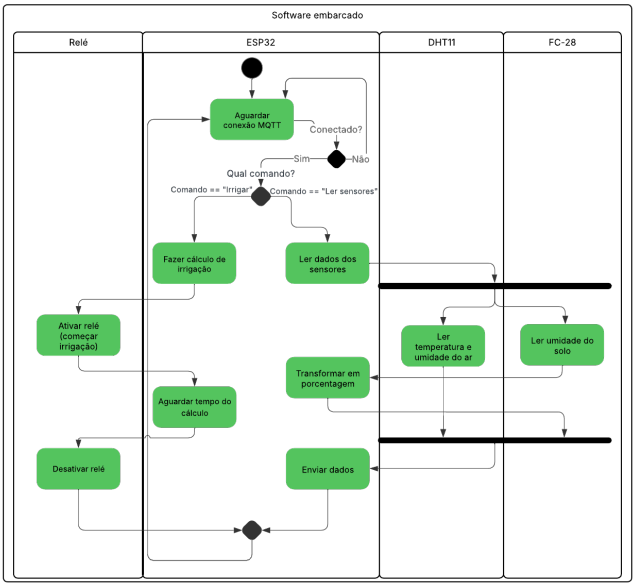

Figura 3. Diagrama de Atividade do Software Embarcado

leitura da temperatura e umidade do ar, enquanto o sensor FC-28 mede a umidade do solo. O dado de umidade do solo é então convertido em porcentagem, e posteriormente enviados de volta ao aplicativo via MQTT.

Optou-se por não realizar a leitura contínua dos sensores, a fim de preservar o consumo de energia, evitando leituras desnecessárias que drenariam energia. Assim, a leitura é executada apenas quando solicitada diretamente pelo usuário ou quando uma função do Firebase requisita os dados para armazenamento no Firestore de hora em hora.

Ao final de cada operação, o sistema retorna ao estado inicial de espera por novos comandos, permitindo o funcionamento contínuo e cíclico da aplicação embarcada.

2) Projeto do Aplicativo Móvel: Os protótipos de tela do aplicativo móvel (Apêndice A) foram desenvolvidos para posterior implementação com o *framework* Flutter, eles foram projetados para oferecer uma interface interativa ao usuário e buscando seguir as boas práticas de IHC (Interface humano-computador), adicionando ícones como metáforas e pouco conteúdo em cada tela. Foram desenvolvidos 3 protótipos de tela para o aplicativo, sendo elas: tela de login/cadastro, tela com os equipamentos (Estações de irrigação) cadastrados e outra tela com os dados atuais da Estação e um gráfico com os dados históricos, além disso o aplicativo contará com um *backend* desenvolvido com Firebase Functions, onde será criada uma função agendada que roda a cada uma hora pegando e salvando no Firestore os dados meteorológicos de todas estações de irrigação cadastradas no mesmo e já as verificando se precisa ou não irrigar olhando a umidade do solo.

A Figura 4 ilustra a estrutura do banco de dados *Firestore* como é um banco não relacional, ele é estruturado por

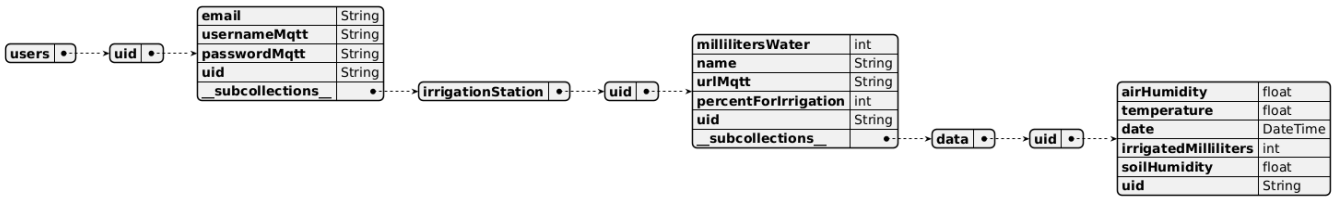

Figura 4. Diagrama do banco de dados

collections, que podem ser comparadas a "pastas". Dentro dessas *collections*, podem existir documentos (documents) ou subcoleções (subcollections).

B. Levantamento de requisitos

Para a construção do quadro Kanban, realizou-se o levantamento de requisitos de ambos os projetos: o aplicativo mobile e a estação de irrigação. Esse processo envolveu a identificação e a documentação das funcionalidades e necessidades específicas de cada componente do sistema, garantindo que todas as demandas fossem mapeadas de forma clara e organizada. Com base nesse levantamento, foram criados três quadros Kanban distintos para gerenciar o desenvolvimento. O primeiro quadro foi dedicado exclusivamente aos requisitos do aplicativo mobile, o segundo aos requisitos da estação de irrigação, e o terceiro quadro integrou os requisitos de ambos os projetos, proporcionando uma visão consolidada do andamento do projeto como um todo.

No Apêndice B, são apresentados os requisitos funcionais (RF) e não funcionais (RNF) referentes à estação de irrigação, abrangendo tanto o circuito eletrônico quanto o software embarcado no microcontrolador. O mesmo apêndice também reúne os requisitos do aplicativo móvel e de seu backend. Esses conjuntos de requisitos estabelecem a base necessária para o funcionamento integrado do sistema, assegurando que hardware e software atendam de forma coerente às necessidades do projeto.

V. Resultados

Nesta seção são apresentados os principais resultados do trabalho, abrangendo tanto a validação dos sensores utilizados quanto os testes práticos realizados com a estação de irrigação e o software de monitoramento.

A. Validação dos sensores

As métricas MAE, RMSE e correlação de Pearson foram utilizadas por permitirem quantificar o erro médio, a dispersão quadrática e a relação linear entre sensores, respectivamente. Para a validação, foi construída uma segunda estação apenas com os sensores (Figuras 5 e 6), colocando-os um ao lado do outro e registrando os dados no Firestore com a função agendada desenvolvida.

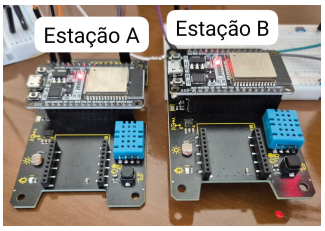

Figura 5. Posição DHT11

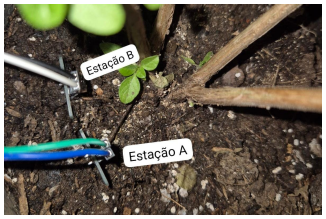

Figura 6. Posição FC-28

1) DHT11: As Figuras 7 e 8 mostram a dispersão dos dados de umidade do ar e temperatura, respectivamente. Foram feitas 104 medições ao todo para essa comparação.

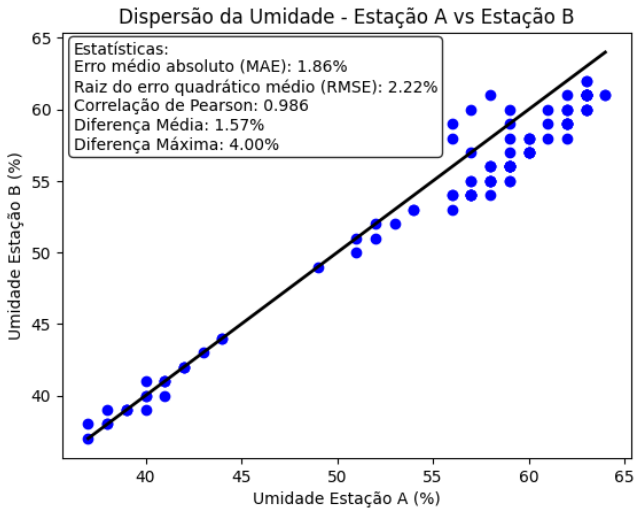

Figura 7. Dispersão Umidade do ar

Os resultados evidenciaram forte correlação entre os dois sensores DHT11, tanto para temperatura (correlação de Pearson = 0,941) quanto para umidade relativa do ar (correlação de Pearson = 0,986). A diferença média observada foi de 0,93 °C na temperatura e 1,57% na umidade do ar. Os erros médios foram MAE = 0,97 °C e RMSE = 1,03 °C para temperatura, e MAE = 1,86% e RMSE = 2,22% para umidade, com diferenças máximas de até 1,8 °C e 4,0%, respectivamente.

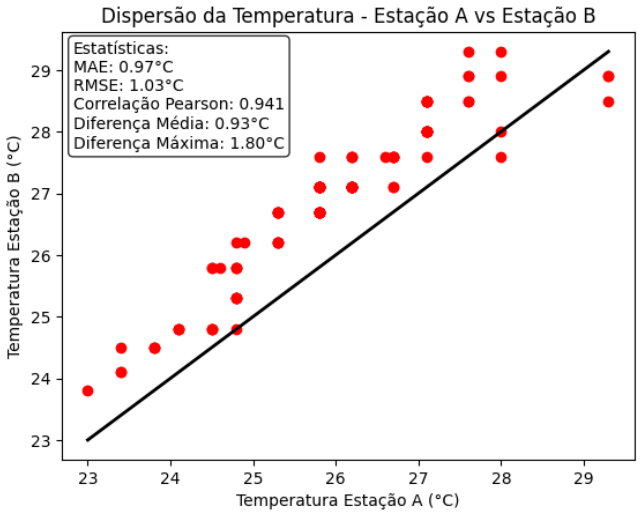

Figura 8. Dispersão Temperatura

A proximidade entre MAE e RMSE indica que as medições do DHT11 apresentaram baixa variação entre si, sem ocorrência de erros pontuais elevados. A estabilidade observada nas leituras, aliada aos altos coeficientes de correlação, mostra que ambos os sensores acompanharam de forma coerente as mudanças ambientais ao longo do tempo. Isso reforça que, apesar das limitações típicas desse sensor, seu desempenho é satisfatório para aplicações que dependem principalmente do acompanhamento de tendências e não exigem medições de alta precisão.

2) FC-28: A Figura 9 mostra a dispersão dos dados de umidade do solo. Para essa comparação, foram feitas 31 medições, apenas a fim de conseguir obter se ambos os sensores acompanham a perda de água do solo e qual a dispersão dos pontos.

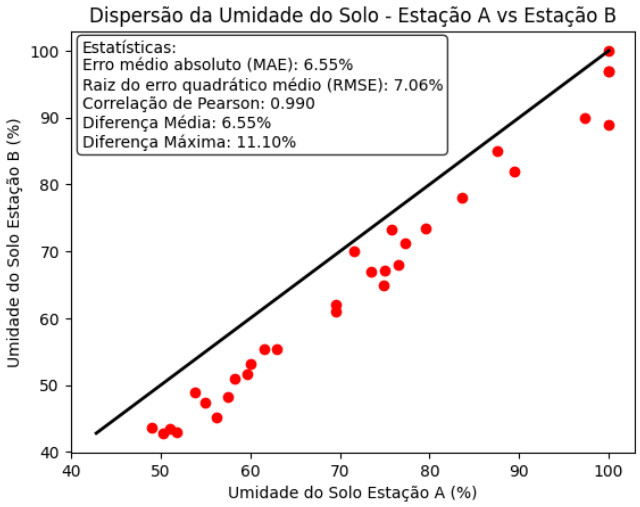

Figura 9. Dispersão Umidade do solo

Na comparação entre os dois sensores FC-28, calibrados de forma idêntica (com 100% definido no solo saturado), observou-se forte correlação entre as leituras (correlação de Pearson = 0,990), o que indica consistência na resposta à variação da umidade do solo.

Entretanto, verificou-se que o sensor da estação B apresentou valores menores que os da estação A, com diferença média de 6,55% e diferença máxima de 11,1% com erros MAE = 6,55% e RMSE = 7,06%, o que evidencia variabilidade distribuída ao longo das medições. Uma possível explicação é a influência de raízes muito próximas ao sensor, já que este começou a perder umidade mais rápido que o sensor da estação A.

B. Validação do sistema

O software desenvolvido apresentou estabilidade durante os testes. A comunicação entre o aplicativo e a estação (Figura 10) foi validada por meio de requisições automáticas realizadas a cada 30 segundos (Figura 11). Ainda na tela inicial (Figura 11), destacam-se, no canto superior direito, os botões para adicionar uma nova estação e inserir credenciais MQTT, e no canto superior esquerdo, o botão de *logout* do sistema.

Figura 10. Estação de irrigação em teste

A função agendada no servidor operou conforme esperado, executando a cada uma hora, solicitando os dados das estações e registrando-os no Firestore, conforme detalhado no Apêndice C. Esses dados podem ser visualizados no aplicativo em forma de gráficos (Figura 12), com possibilidade de navegação por dias ou seleção de datas em calendário.

Além disso, o aplicativo apresenta os dados atuais da estação e disponibiliza um recurso de irrigação manual (Figura 13), no qual o usuário pode definir quantos ml deseja irrigar.

Os testes práticos foram realizados com uma planta de manjericão. Segundo a literatura, essa espécie necessita de irrigação quando o solo apresenta entre 75% e 50% de umidade no solo [32]. Neste trabalho, definiu-se 55% para a irrigação automática, irrigando 150 ml de água. Durante

Figura 11. Tela Home

Figura 12. Tela de dados da estação de irrigação

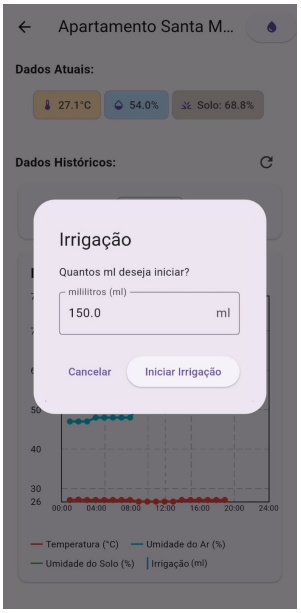

Figura 13. Diálogo irrigação manual

o período de 5 dias de testes, o sistema atendeu a todas as irrigações necessárias, operando de forma consistente com os parâmetros informados, sendo executadas duas irrigações de forma automática, irrigando 150 ml cada e quatro de forma manual, duas irrigando 50 ml e duas irrigando 100 ml. Foram feitas irrigações com um baixo volume de água a fim de não saturar o solo e assim conseguir fazer mais irrigações em um curto período de tempo.

VI. Conclusão

Este trabalho apresentou o desenvolvimento de um sistema de irrigação automático usando um microcontrolador ESP32, um sensor DHT11, um sensor FC-28 e uma válvula solenóide acionada a partir de um relé. Este sistema foi integrado utilizando o protocolo MQTT em um aplicativo que conseguia capturar os dados e fazer irrigações tanto via função agendada, salvando-os no banco de dados, quanto via aplicativo, não salvando os dados, mas podendo irrigar de forma manual.

A utilização da metodologia Kanban facilitou o desenvolvimento do trabalho, pois organizou as etapas do projeto de forma ágil e interativa assegurando o avanço constante e a priorização de entregas funcionais.

Os resultados obtidos demonstraram que a estação de irrigação apresentou desempenho satisfatório, com acionamento e fechamento corretos da válvula solenóide. Os sensores utilizados apresentaram boa consistência nas medições, de acordo com as análises estatísticas realizadas. Além disso, a comunicação entre o aplicativo móvel, a função agendada e a estação ocorreu de forma estável e eficiente, garantindo que todas as irrigações necessárias fossem feitas e mantendo o nível de umidade do solo acima do definido para irrigação automática.

Apesar dos bons resultados, observou-se uma limitação quanto à precisão do sensor FC-28, que se mostrou sensível às variações do solo, mesmo quando posicionado lado a lado com outro sensor idêntico. Ainda assim, o desempenho geral foi considerado adequado para aplicações de pequeno porte, como o cultivo doméstico.

Como proposta para trabalhos futuros, recomenda-se a integração de um hidrômetro à estação de irrigação, assim possibilitando a medição precisa do volume de água utilizado. Essa melhoria eliminaria a necessidade de estimativas baseadas em cálculos de tempo de abertura da válvula, tornando o sistema ainda mais preciso.

Referências

[1] H. Ritchie. *Not the End of the World: How We Can Be the First Generation to Build a Sustainable Planet*. New York: Little, Brown Spark, 2024.

[2] W. Lewandrowski et al. “Global change impacts on arid zone ecosystems: Seedling establishment processes are threatened by temperature and water stress”. Em: *Ecology and Evolution* 11.12 (2021), pp. 8071–8084.

[3] L. García et al. “IoT-Based Smart Irrigation Systems: An Overview on the Recent Trends on Sensors and IoT Systems for Irrigation in Precision Agriculture”. Em: (2020). URL: https://doi.org/10.3390/s20041042.

[4] L. Atzori, A. Iera e G. Morabito. “The Internet of Things: A Survey”. Em: *Computer Networks* 54.15 (2010), pp. 2787–2805.

[5] K. Rose, S. Eldridge e L. Chapin. *The Internet of Things: An Overview – Understanding the Issues and Challenges of a More Connected World*. Acessado em 07 de abril de 2025. 2015. URL: https://www.internetsociety.org/wp-content/uploads/2017/08/ISOC-IoT-Overview-20151221-en.pdf.

[6] Embrapa. Acessado em 25 de junho de 2025. URL: https://www.embrapa.br/tema-automacao-e-agricultura-de-precisao/sobre-o-tema#:~:text=A%20automa%C3%A7%C3%A3o%20agropecu%C3%A1ria%20pode%20ser,a%20capacidade%20de%20trabalho%20humano.

[7] Matthew Okner e David Veksler. "Automated Water Irrigation System". Em: *arXiv preprint arXiv:2501.10610* (2025).

[8] J. Fraden. *Handbook of Modern Sensors: Physics, Designs, and Applications*. 5ª ed. Cham: Springer, 2016.

[9] Mouser Electronics. *DHT11 Humidity Temperature Sensor*. Acessado em 08 de abril de 2025. URL: https://www.mouser.com/datasheet/2/758/DHT11-Technical-Data-Sheet-Translated-Version-1143054.pdf?srsltid=AfmBOorLky2dPG3M1PPQK_Q68OnTjNihmvD3O1mWgvL-VRhzc3mlbZf-%5C%3E.

[10] J. Fuller. *FC-28 Soil Moisture Sensor Module*. Acessado em 08 de abril de 2025. URL: https://www.datasheethub.com/fc-28-soil-moisture-sensor-module/.

[11] Tameson. *Solenoid Valve Types - A Comprehensive Guide*. Acessado em 08 de abril de 2025. 2024. URL: https://tameson.com/pages/solenoid-valve-types.

[12] MakerHero. *Módulo Relé 3.3V 8 Canais*. Acessado em 11 de maio de 2025. URL: https://www.makerhero.com/produto/modulo-rele-3-3v-8-canais/.

[13] ON Semiconductor. *MC7800, MC7800A, MC7800AE, NCV7800: Voltage Regulators Positive 1.0 A*. Publication Order Number: MC7800/D, Rev. 29. 2021. URL: https://www.onsemi.com/pdf/datasheet/mc7800-d.pdf.

[14] Espressif Systems. *ESP32 Datasheet*. Acessado em 05 de abril de 2025. 2018. URL: https://d229kd5ey79jzj.cloudfront.net/1013/esp32_datasheet_en.pdf.

[15] M. Banzi e M. Shiloh. *Getting Started with Arduino*. 3ª ed. Maker Media, 2014.

[16] E. Windmill. *Flutter in Action*. Manning Publications, 2021.

[17] M. Miletto e E. Napoli. *Beginning Flutter: A Hands-On Guide to App Development*. Wiley, 2020.

[18] Google. *Firebase: Produtos para Construção*. Acessado em 05 de abril de 2025. 2023. URL: https://firebase.google.com/products-build?hl=pt-br.

[19] Google. *Cloud Functions para Firebase*. Acessado em 05 de abril de 2025. 2025. URL: https://firebase.google.com/docs/functions?hl=pt-br.

[20] Google. *Firebase Authentication*. Acessado em 05 de abril de 2025. 2025. URL: https://firebase.google.com/docs/auth?hl=pt-br.

[21] Google. *Cloud Firestore*. Acessado em 06 de abril de 2025. 2025. URL: https://firebase.google.com/docs/firestore?hl=pt-br.

[22] Microsoft. *TypeScript for the New Programmer*. Acessado em 06 de abril de 2025. URL: https://www.typescriptlang.org/docs/handbook/typescript-from-scratch.html.

[23] GitHub. *Sobre o Git*. Acessado em 06 de abril de 2025. URL: https://docs.github.com/pt/get-started/using-git/about-git#about-version-control-and-git.

[24] HiveMQ. *MQTT Essentials: All Core Concepts Explained*. Acessado em 07 de abril de 2025. 2025. URL: https://www.hivemq.com/mqtt/.

[25] R. Pressman e B. Maxim. *Engenharia de Software: Uma Abordagem Profissional*. 9ª ed. São Paulo: McGraw-Hill Brasil, 2021.

[26] Ananta N. et al. "Development of data-driven models for wind speed forecasting in Australia". Em: *Predictive modelling for energy management and power systems engineering*. Elsevier, 2021, pp. 143–190.

[27] J. Frost. *Erro Quadrático Médio (RMSE)*. Acessado em 08 de outubro de 2025. URL: https://statisticsbyjim.com/regression/root-mean-square-error-rmse/.

[28] F. Britto e A. S. Júnior. "Desvendando os Mistérios do Coeficiente de Correlação de Pearson (r)". Em: *Revista política hoje* 18.1 (2009), pp. 115–146.

[29] A. Benbatouche e B. Kadri. "Design and realization of low-cost solenoid valve remotely controlled, application in irrigation network". Em: *Bulletin of Electrical Engineering and Informatics* 11.3 (2022), pp. 1779–1788. ISSN: 2302-9285. DOI: 10.11591/eei.v11i3.4123. URL: https://journal.beei.org/index.php/EEI/article/view/4123.

[30] K. Kannan et al. "An internet of things based smart irrigation using solenoid valve". Em: *International Journal of Recent Technology and Engineering (IJRTE)* 9.1 (2020), pp. 2018–2023.

[31] C. Chen et al. "Web-based Internet of Things on environmental and lighting control and monitoring system using node-RED, MQTT and Modbus communications within embedded Linux platform". Em: *Internet of Things* 27 (2024), p. 101305.

[32] S. Nam, S. Kang e J. Kim. "Maintaining a constant soil moisture level can enhance the growth and phenolic content of sweet basil better than fluctuating irrigation". Em: *Agricultural Water Management* 238 (2020), p. 106203.

APÊNDICE

APÊNDICE A: PROTÓTIPOS DE TELA

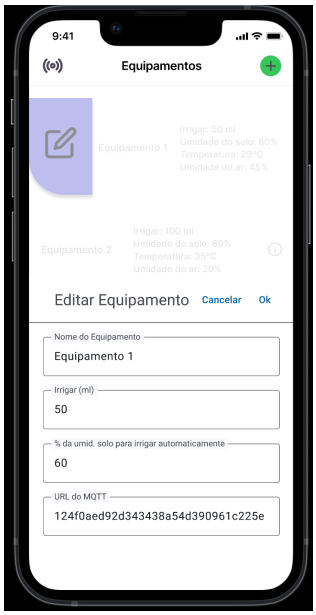

Figura 14. Tela Home do Aplicativo

Figura 15. Tela de Equipamento

APÊNDICE B: REQUISITOS FUNCIONAIS E NÃO FUNCIONAIS DO PROJETO

Tabela I
REQUISITOS ESTAÇÃO DE IRRIGAÇÃO

RNF01	Enviar dados somente após receber uma solicitação de conexão via MQTT.
RF01	Utilizar o protocolo MQTT para comunicação bidirecional entre o microcontrolador e o aplicativo mobile.
RF02	Irrigar a quantidade de Mililitro (ml) de água especificada, recebida como parâmetro via MQTT do aplicativo mobile.
RF03	Operar com uma tensão de no mínimo 12v, regulada para 5 V pelo MC7805CT, para alimentar o microcontrolador ESP32.
RF04	Utilizar sensor DHT11 e FC-28 para verificar dados ambientais.
RF05	Utilizar uma válvula solenoide com um relé para acionamento da irrigação.

Tabela II
REQUISITOS DO APLICATIVO MÓVEL

RNF01	Garantir que o aplicativo seja compatível com dispositivos Android e iOS.
RF01	Utilizar o *Firebase Auth* para o cadastro e login de usuários.
RF02	Utilizar o banco de dados *Firebase Firestore* para a persistência dos dados.
RF03	Permitir o cadastro de módulos de irrigação no aplicativo.
RF04	Permitir que o usuário defina uma porcentagem de umidade do solo para ativação da irrigação automática.
RF05	Utilizar o protocolo MQTT para comunicação com o ESP32 enviando como parâmetro a quantidade de mililitro de água como parâmetro para o Módulo de Irrigação.
RF06	Utilizar o *Firebase Functions* com uma função *scheduler* para, a cada hora, solicitar os dados de todos os Módulos de Irrigação cadastrados no banco e verificar se a umidade do solo é inferior à definida pelo usuário, acionando a irrigação automática, se necessário.
RF07	Visualizar com gráficos de linha os dados históricos salvos no banco *Firestore*.
RF08	Possibilitar a visualização dos dados em tempo de execução na tela do aplicativo movel.

APÊNDICE C: LOG DA FUNÇÃO AGENDADA CAPTURANDO OS DADOS AMBIENTAIS E VERIFICANDO SE PRECISA IRRIGAR

2025-10-01 11:55:09.093	scheduleDataGet	2g8zm3jxuub1		Tentando conectar com usuário: tcc-teste
2025-10-01 11:55:09.637	scheduleDataGet	2g8zm3jxuub1		Conectado à estação BmueRwZ7W7ePostTyrPz via MQTT
2025-10-01 11:55:09.638	scheduleDataGet	2g8zm3jxuub1		Comando 'Ler sensor' enviado para estação BmueRwZ7W7ePostTyrPz
2025-10-01 11:55:16.582	scheduleDataGet	2g8zm3jxuub1		Estação BmueRwZ7W7ePostTyrPz - Recebido esp32/umidade: 57
2025-10-01 11:55:16.587	scheduleDataGet	2g8zm3jxuub1		Estação BmueRwZ7W7ePostTyrPz - Recebido esp32/umidade-solo: 63.4
2025-10-01 11:55:16.587	scheduleDataGet	2g8zm3jxuub1		Estação BmueRwZ7W7ePostTyrPz - Recebido esp32/temperatura: 26.2
2025-10-01 11:55:16.588	scheduleDataGet	2g8zm3jxuub1		Todos os dados recebidos para estação BmueRwZ7W7ePostTyrPz:
2025-10-01 11:55:16.589	scheduleDataGet	2g8zm3jxuub1		Dados do sensor recebidos:
2025-10-01 11:55:16.589	scheduleDataGet	2g8zm3jxuub1		Iniciando irrigação para a estação BmueRwZ7W7ePostTyrPz
2025-10-01 11:55:16.591	scheduleDataGet	2g8zm3jxuub1		Salvando dados no caminho: users/Lq9WmXcvnuUu5cadk8qXffuv2o92/irrigation_stations/BmueRwZ7W7ePostTyrPz/data
2025-10-01 11:55:16.696	scheduleDataGet	2g8zm3jxuub1		Conexão MQTT fechada para estação BmueRwZ7W7ePostTyrPz

Figura 16. Log da função agendada