



**UNIVERSIDADE PRESBITERIANA MACKENZIE**  
**Faculdade de Computação e Informática**

**Irrigação Automática: Sistema de Irrigação Automática para Plantações em Pequena Escala.**

**Laryssa Fernandes Mariotto <sup>1</sup>, Nicolas Soares Santos <sup>2</sup>, Leandro Carlos Fernandes <sup>3</sup>**

Universidade Presbiteriana Mackenzie (UPM)

Rua da Consolação, 930 Consolação, São Paulo - SP, 01302-907 – Brazil

nicolassoares.santos@mackenzista.com.br, laryssa.mariotto@mackenzista.com.br

leandro.fernandes@mackenzie.br

**Abstract.** *This project proposes the utilization of NodeMCU (ESP32) as a low-cost alternative to automate irrigation processes in small-scale crops. The automated irrigation system aims to simplify daily life and promote plant health, thus contributing to water usage optimization. The combination of Soil Moisture sensors and a mini water pump plays a crucial role in efficient irrigation management. Furthermore, the paper provides an overview of the methods used in system development, offering a comprehensive understanding of the techniques implemented to achieve the proposed objectives.*

**Resumo.** *Este projeto propõe a utilização da NodeMCU(ESP32) como uma alternativa de baixo custo para automatizar o processo de irrigação em cultivos de pequena escala. O sistema de irrigação automatizada visa facilitar a vida cotidiana e promover a saúde das plantas, contribuindo assim para a otimização do uso da água. A combinação de sensores de Umidade do Solo e minibomba de água desempenha um papel fundamental na gestão eficiente da irrigação. Além disso, o artigo apresenta uma visão geral dos métodos utilizados no desenvolvimento do sistema, proporcionando uma compreensão completa das técnicas implementadas para alcançar os objetivos propostos.*

## 1. Introdução

**A água: um recurso vital sob ameaça.** Segundo o Relatório Mundial das Nações Unidas sobre o Desenvolvimento dos Recursos Hídricos 4, a agricultura consome até 70% da água doce global, chegando a 90% em economias em rápido crescimento. Essa demanda crescente coloca em risco a segurança hídrica e a sustentabilidade ambiental, especialmente para pequenos agricultores que muitas vezes não possuem acesso a tecnologias eficientes de irrigação.

### 1.1. Irrigação Automática: Uma Solução Inovadora para a Agricultura de Pequena Escala.

É nesse contexto que surge o projeto "Irrigação Automática: Sistema de Irrigação Automática para Plantações em Pequena Escala". Através da implementação de um sistema inovador baseado no NodeMCU ESP32, buscamos otimizar o uso da água na agricultura de pequena escala, promovendo a sustentabilidade ambiental, a segurança alimentar e a inclusão social.

### 1.2. O Sistema em Ação:

**Monitoramento preciso da umidade do solo:** O sistema utiliza sensores para monitorar a umidade do solo em tempo real, garantindo que a irrigação seja realizada apenas quando necessário.

**Irrigação automatizada e eficiente:** Com base nos dados coletados, o sistema ajusta automaticamente a irrigação, evitando o desperdício de água e garantindo que as plantas recebam a quantidade ideal de água para seu desenvolvimento.

### 1.3. Contribuindo para os Objetivos de Desenvolvimento Sustentável (ODS):

O projeto está alinhado aos ODS 2 (Fome Zero e Agricultura Sustentável) e 3 (Saúde e Bem-Estar) da Agenda 2030 da ONU. Através da otimização do uso da água, o projeto contribui para:

- **Aumento da produtividade das plantações e da segurança alimentar:** Com o uso eficiente da água, as plantações podem se desenvolver com mais saúde e vigor, resultando em maior produtividade e segurança alimentar para as comunidades locais.

- **Promoção da agricultura sustentável:** A redução do desperdício de água e a otimização da irrigação contribuem para a preservação dos recursos hídricos e a promoção de práticas agrícolas mais sustentáveis.
- **Melhoria da saúde e do bem-estar:** O acesso à água potável e à segurança alimentar é fundamental para a saúde e o bem-estar das comunidades, especialmente em áreas rurais.

Acessibilidade e Inclusão Social: O NodeMCU ESP32, plataforma de desenvolvimento utilizada no projeto, é acessível e fácil de usar, tornando o sistema viável para agricultores com diferentes níveis de escolaridade. Essa característica promove a inclusão social e a democratização do acesso à tecnologia no campo.

#### **1.4. Evolução Histórica da Irrigação**

A irrigação é uma prática antiga que tem desempenhado um papel crucial na agricultura e no desenvolvimento das civilizações ao longo da história. Desde os primeiros sistemas rudimentares até as tecnologias modernas, a evolução da irrigação reflete a necessidade humana de controlar o uso da água para fins agrícolas.

##### **1.4.1. Primórdios da Irrigação (6000 a.C. - 1000 d.C.)**

Egito (3.100 a.C.): Durante a primeira dinastia do Egito, o faraó Menes liderou a construção de represas e canais, desviando as águas do Nilo para o lago "Moeris", possibilitando a agricultura em áreas anteriormente inabitáveis. (ENCYCLOPAEDIA BRITANNICA, 2024)

##### **1.4.2. Idade Média e Renascimento (1000 d.C. - 1800 d.C.)**

Expansão para o Mediterrâneo e Europa: A irrigação se espalhou pelo Império Romano, com a construção de aquedutos e sistemas de canais, e posteriormente, pelos árabes na Península Ibérica, introduzindo novas técnicas como as norias.

Inovações tecnológicas: A invenção da bomba d'água no século XVI e a canalização de rios no século XVII impulsionaram a irrigação em novas áreas e aumentaram a eficiência do processo.

##### **1.4.3. Revolução Industrial e Era Moderna (1800 d.C. - Presente)**

Modernização da irrigação: No século XIX, surgem os primeiros sistemas de irrigação por aspersão e gotejamento, aumentando a eficiência e a economia de água.

Desenvolvimento de tecnologias: O século XX foi marcado pelo desenvolvimento de materiais como o PVC e Alumínio, tubos e bombas mais eficientes, e a automação dos sistemas de irrigação.

Desafios e perspectivas: A escassez de água e a necessidade de aumentar a produção agrícola exigem o desenvolvimento de métodos de irrigação mais eficientes e sustentáveis, como a irrigação por microaspersão, reuso de água e agricultura de precisão.

### **1.5. Revisão de Trabalhos Correlatos**

Diversos projetos já exploraram o uso de microcontroladores, como o Arduino, para automatizar a irrigação. Um exemplo notável é o trabalho realizado por Pedro Henrique Silva Medeiros na Universidade Federal de Ouro Preto, intitulado "SISTEMA DE IRRIGAÇÃO AUTOMATIZADO PARA PLANTAS CASEIRAS." Este projeto apresenta uma abordagem abrangente, integrando sistemas de controle, hidráulico e uma interface web amigável para otimizar a irrigação de plantas domésticas (MEDEIROS, 2022).

Outro trabalho relevante no campo é o desenvolvido por Renato Tavares, disponível no repositório GitHub. Este projeto utiliza as plataformas NodeMCU, ESP8266, IoT e integração com a assistente virtual Alexa para aprimorar o controle e a interatividade no processo de irrigação (TAVARES, 2022). Essa iniciativa destaca-se pela incorporação de tecnologias modernas e pela integração com assistentes virtuais, proporcionando uma experiência avançada de automação na irrigação de plantas

### **1.6. O Projeto Irrigação Automática**

Este artigo científico tem como objetivo central apresentar e analisar o Projeto de Irrigação Automática, focando na capacidade da tecnologia em conferir autonomia ao cuidado das plantas em ambientes específicos, como a agricultura familiar e pequenas plantações independentes, como plantações caseiras. A proposta visa proporcionar uma visão aprofundada sobre o projeto, destacando não apenas sua funcionalidade técnica, mas também sua aplicabilidade prática em contextos irrigação de menor escala.

## **1.7. Estrutura do Artigo**

### 1. Introdução

1.1. Irrigação Automática: Uma Solução Inovadora para a Agricultura de Pequena Escala

### 1.2. O Sistema em Ação

### 1.3. Contribuindo para os Objetivos de Desenvolvimento Sustentável (ODS)

### 1.4. Evolução Histórica da Irrigação

### 1.5. Revisão de Trabalhos Correlatos

### 1.6. O Projeto Irrigação Automática

### 1.7. Estrutura do Artigo

## 2. Materiais e Métodos

### 2.1. Descrição dos Módulos/Hardware Empregados

#### 2.1.1. NodeMCU + ESP32-DevKitC V4

##### 2.1.1.1. ESP32: O Microcontrolador Multifacetado

##### 2.1.1.2. NodeMCU: A Plataforma Amigável para Desenvolvedores

#### 2.1.2. Especificações ESP32

#### 2.1.3. Pinagem NodeMCU(ESP32)

#### 2.1.4. Sensor de Umidade de Solo HL-69

#### 2.1.5. Minibomba de Água (D'água) para Arduino Rs-385

#### 2.1.6. Módulo Relé 5v 10a 1 Canal Com Optoacoplador

#### 2.1.7. Alimentação de Energia

#### 2.1.8. Conexões

##### 2.1.8.1. ½ Metro De Fio Paralelo 0,5mm

##### 2.1.8.2. 1,5 Metros De Mangueira Para Aquário

##### 2.1.8.3. Jumpers

#### 2.1.8.4. Adaptador de Placa ESP32 ou Placa de Ensaio

### 2.2. Protocolos de Comunicação e Serviços Utilizados

#### 2.2.1. Protocolo MQTT

#### 2.2.2. Broker MQTT

#### 2.2.3. Configuração do MQTT

### 2.3. Descrição do Software Desenvolvido

#### 2.3.1. Desenvolvimento do Circuito

#### 2.3.2. Programação do NodeMCU(ESP32)

#### 2.3.3. Configuração do MQTT e Conexão Wi-Fi

#### 2.3.4. Implementação e Integração

### 2.4. Montagem do Protótipo

### 2.5. Funcionamento do Projeto

#### 2.5.1. Configuração do Ambiente de Desenvolvimento

#### 2.5.2. Conexão WiFi e MQTT

#### 2.5.3. Leitura da Umidade do Solo

#### 2.5.4. Controle da Bomba de Água

##### 2.5.4.1. Lógica de Bloqueio de Relé/Bomba

##### 2.5.4.2. Bloqueio do Sistema de Irrigação em Caso de Estagnação

#### 2.5.5. Integração com o Broker MQTT

#### 2.5.6. Monitoramento e Controle Remoto

## 3. Resultados

### 3.1. Descrição do Projeto

#### 3.1.1. Imagem 1: Protótipo Completo

#### 3.1.2. Imagem 2: Sensor de Umidade

#### 3.1.3. Imagem 3: Relé e Atuador

#### 3.1.4. Imagem 4: Interface do Aplicativo Móvel

- 3.2. Vídeo-Demonstração
- 3.3. Medições de Tempo de Resposta
- 3.4. Gráficos e Capturas de Tela
- 3.5. Link para o Repositório GitHub
- 4. Conclusões
  - 4.1. Alcanço dos Objetivos
  - 4.2. Principais Problemas Enfrentados e Soluções
  - 4.3. Vantagens e Desvantagens do Projeto
    - 4.3.1. Vantagens
    - 4.3.2 Desvantagens
  - 4.4 Melhorias Futuras

## **2. Materiais e Métodos**

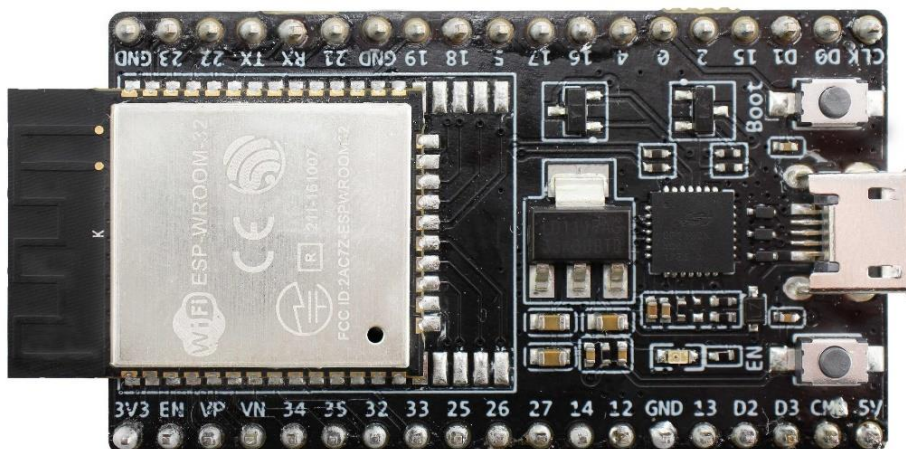
Nesta seção, serão detalhados os materiais e métodos utilizados para a construção do projeto. A escolha dos componentes e as especificações técnicas são fundamentais para garantir o funcionamento correto e eficiente do projeto. A seguir, são apresentados os módulos e hardware empregados, juntamente com suas descrições detalhadas, os métodos utilizados para desenvolver o projeto.

### **2.1. Descrição dos Módulos/Hardware Empregados**

Este projeto de irrigação automatizada utiliza uma combinação de hardware e software para monitorar e controlar a umidade do solo e fornecer água para as plantas conforme necessário, seguir os hardwares utilizados:

### 2.1.1. NodeMCU + ESP32-DevKitC V4:

Figura 1. NodeMCU + ESP32. Fonte: [Espressif](#)



#### 2.1.1.1. ESP32: O Microcontrolador Multifacetado

O ESP32 é um avançado microcontrolador desenvolvido pela Espressif que integra um processador dual-core, conectividade Wi-Fi e Bluetooth de última geração, e possui baixo consumo de energia. Este microcontrolador oferece desempenho superior em comparação com seu predecessor, o ESP8266, sendo ideal para uma ampla gama de projetos eletrônicos, desde simples sensores até sistemas complexos de automação.

#### 2.1.1.2. NodeMCU: A Plataforma Amigável para Desenvolvedores

NodeMCU é uma plataforma open-source que inclui um firmware pré-instalado no ESP32, facilitando a programação e prototipagem de projetos. Suportando linguagens como Lua e C/C++, a NodeMCU simplifica o desenvolvimento, atraindo tanto iniciantes quanto desenvolvedores experientes. A plataforma é sustentada por uma comunidade vibrante, proporcionando suporte e colaboração contínuos.

A integração do ESP32 com a plataforma NodeMCU resulta em uma poderosa ferramenta para o desenvolvimento de projetos de eletrônica.

### 2.1.2. Especificações ESP32

Se você quiser entrar em detalhes técnicos e específicos, pode dar uma olhada nas seguintes especificações detalhadas do ESP32 (KUONGSHUN ELECTRONIC SHOP) — para mais detalhes, consulte a folha de dados, resumidamente:

- Conectividade sem fio:

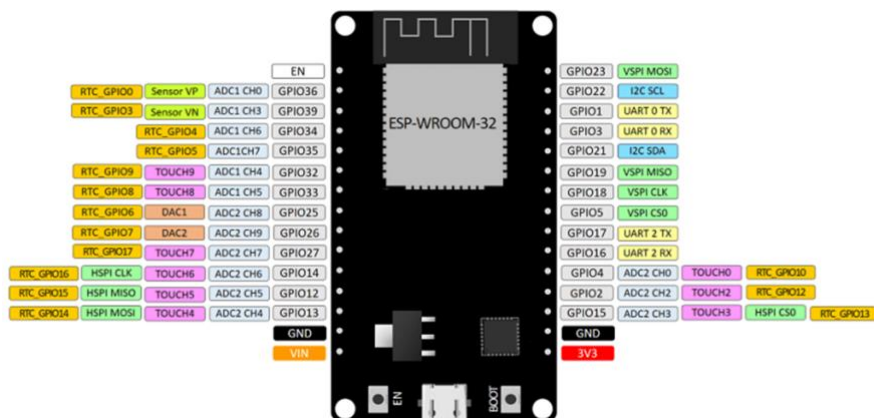


- WiFi: taxa de dados de 150,0 Mbps com HT40
- Bluetooth: BLE (Bluetooth de Baixa Energia) e Bluetooth Clássico
- Processador:
  - Processador microcontrolado de 32 bits LX6 Dual-Core Tensilica Xtensa, rodando a 160 ou 240 MHz
- Memória:
  - ROM: 448 KB (para inicialização e funções principais)
  - SRAM: 520 KB (para dados e instruções)
  - SRAM rápida do RTC: 8 KB (para armazenamento de dados e CPU principal durante a inicialização do RTC a partir do modo de sono profundo)
  - SRAM lenta do RTC: 8 KB (para acesso ao co-processador durante o modo de sono profundo)
  - eFuse: 1 Kbit (dos quais 256 bits são usados para o sistema (endereço MAC e configuração do chip) e os 768 bits restantes são reservados para aplicativos do cliente, incluindo criptografia de Flash e ID do Chip)
- Flash embutido: flash conectado internamente via IO16, IO17, SD\_CMD, SD\_CLK, SD\_DATA\_0 e SD\_DATA\_1 no ESP32-D2WD e ESP32-PICO-D4.
  - 0 MiB (chips ESP32-D0WDQ6, ESP32-D0WD e ESP32-S0WD)
  - 2 MiB (chip ESP32-D2WD)
  - 4 MiB (módulo SiP ESP32-PICO-D4)
- Baixo consumo de energia: garante que você ainda possa usar conversões ADC, por exemplo, durante o sono profundo.
- Entrada/Saída Periférica:
  - Interface periférica com DMA que inclui toque capacitivo
  - ADCs (Conversor Analógico-Digital)
  - DACs (Conversor Digital-Analógico)
  - I<sup>2</sup>C (Inter-Integrated Circuit)

- UART (Receptor/Transmissor Assíncrono Universal)
- SPI (Interface Periférica Serial)
- I<sup>2</sup>S (Som Interchip Integrado)
- RMI (Interface Independente de Mídia Reduzida)
- PWM (Modulação por Largura de Pulso)
- Segurança: aceleradores de hardware para AES e SSL/TLS.

### 2.1.3. Pinagem NodeMCU(ESP32)

Figura 2 - Pinagem NodeMCU(ESP32). Fonte: [UsinaInfo](#)



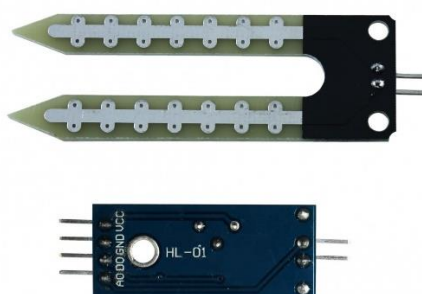
A entradas do ESP32 incluem:

- 18 canais de Conversor Analógico-Digital (ADC)
- 3 interfaces SPI
- 3 interfaces UART
- 2 interfaces I2C
- 16 canais de saída PWM
- 2 Conversores Digital-Analógico (DAC)
- 2 interfaces I2S
- 10 GPIOs de detecção capacitiva

Os recursos do conversor analógico-digital (ADC) e do conversor digital-analógico (DAC) estão atribuídos a pinos estáticos específicos. No entanto, você pode decidir quais pinos serão UART, I2C, SPI, PWM, etc. - basta atribuí-los no código. Isso é possível devido à característica de multiplexação do chip ESP32. Embora seja possível definir as propriedades dos pinos no software, há pinos atribuídos por padrão, conforme mostrado na figura abaixo. (KUONGSHUN ELECTRONIC SHOP)

#### 2.1.4. Sensor de Umidade de Solo HL-69

Figura 3 - Sensor de Umidade de Solo HL-69. Fonte: [UsinalInfo](#)



Este sensor avalia a umidade do solo através da resistência elétrica, fornecendo dados cruciais para o controle automático da irrigação. Seu design inclui uma sonda para inserção no solo e uma saída analógica que varia com o nível de umidade.

#### 2.1.5. Minibomba de Água (D'água) para Arduino Rs-385

Figura 4 - Minibomba de Água para Arduino 12V RS385. Fonte: [DataSheet4u](#)



**Funcionalidade:**

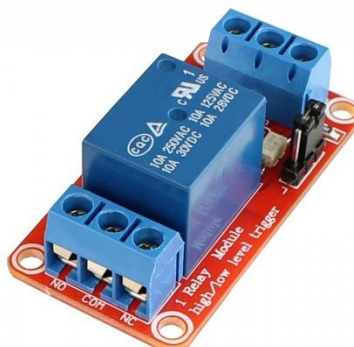
- **Irrigação Automatizada:** A bomba ideal para sistemas de irrigação inteligentes, controlada pelo NodeMCU (ESP32) para irrigar com precisão e eficiência.
- **Transporte de Líquidos:** Adequada para diversos projetos que exigem movimentação de líquidos, desde experimentos científicos até sistemas robóticos.

#### **Características:**

- **Motor RS-385:** Garante operação suave e confiável, ideal para integração com microcontroladores.
- **Fluxo Ajustável:** Fornece entre 1500ml e 2000ml por minuto, atendendo a necessidades variadas.
- **Eficiência e Precisão:** Combinação perfeita para projetos que exigem controle preciso do fluxo de água.
- **Compacta e Leve:** Facilita a integração em diversos tipos de projetos, sem comprometer a portabilidade.
- **Versatilidade:** Aplicações em carrinhos robóticos, sistemas hidráulicos, irrigação automatizada e muito mais.
- **Tensão de Operação:** 9V a 15V, compatível com diversas fontes de alimentação.
- **Elevador de Água:** Altura máxima de elevação de até 3 metros.
- **Aspiração Eficaz:** Aspira água em profundidades de até 2 metros.

### 2.1.6. Módulo Relé 5v 10a 1 Canal Com Optoacoplador

Figura 5 - Relé 5v 10a 1 Canal Com Optoacoplador. Fonte: [UsinaInfo](#)



Este módulo é empregado para controlar a minibomba de água, isolando eletricamente o circuito de controle do circuito de carga, o que permite a operação de dispositivos de alta potência de maneira segura.

### 2.1.7. Alimentação de Energia

Figura 6 - Fonte De Alimentação Chaveada 12vdc 1ª. Fonte: [UsinaInfo](#)



A fonte de alimentação chaveada 12V 1A, conforme descrito, é uma fonte de alimentação amplamente utilizada em projetos de automação residencial e eletrônica de baixa potência. (USINAINFO)

## 2.1.8. Conexões

### 2.1.8.1. ½ Metro De Fio Paralelo 0,5mm

Figura 7 - Fio Paralelo. Fonte: [Santil](#)



#### DETALHES:

- Este fio paralelo de 0,5mm é utilizado para fazer conexão com o Relé, Fonte de 12V e bomba de água.
- É flexível e fácil de manusear.

### 2.1.8.2. 1,5 Metros De Mangueira Para Aquário

Figura 8 - Mangueira Para Aquário. Fonte: [Atrantidaaquiros](#)



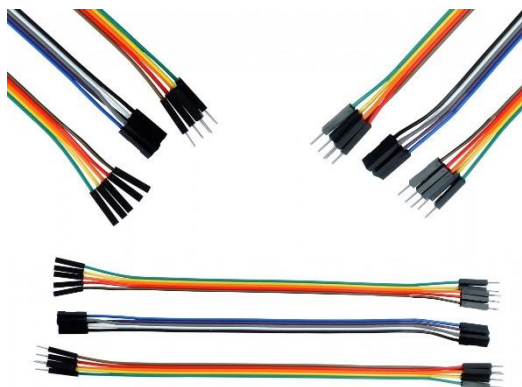
#### DETALHES:

- Esta mangueira é utilizada para transportar a água da minibomba para o local desejado no projeto.

- É resistente à água e flexível o suficiente para ser dobrada conforme necessário.

### 2.1.8.3. Jumpers

Figura 9 – Jumpers. Fonte: [UsinalInfo](#)



#### DETALHES:

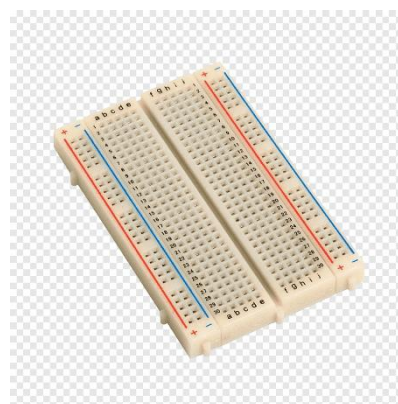
- Os jumpers são pequenos cabos com pinos machos em ambas as extremidades, utilizados para fazer conexões temporárias entre os pinos de componentes eletrônicos.
- Eles são utilizados para facilitar a prototipagem e testes de circuitos.

### 2.1.8.4. Adaptador de Placa ESP32 ou Placa de Ensaio

Figura 10 - Adaptador De Terminal De Parafuso Da Placa De Expansão. Fonte: [Probots](#)



Figura 11 - Placa de ensaio. Fonte: [Pngegg](#)



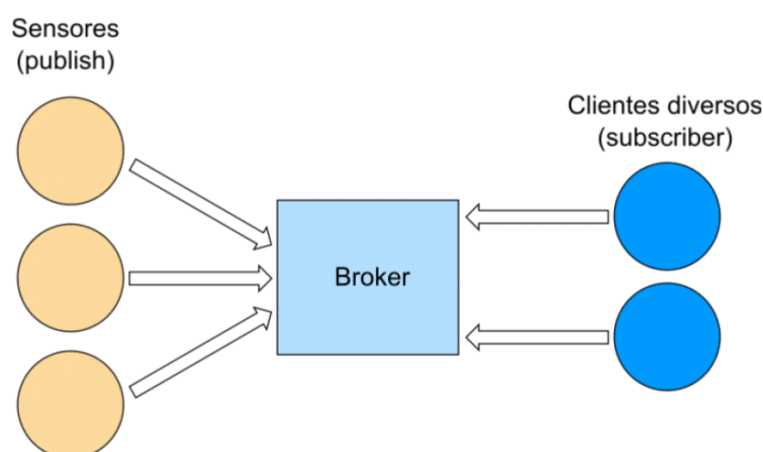
Utilizados para a prototipagem e organização do projeto, estes adaptadores permitem a fácil conexão do NodeMCU (ESP32) com outros componentes eletrônicos, assegurando uma montagem eficiente e flexível.

## 2.2. Protocolos de Comunicação e Serviços Utilizados

Para garantir uma comunicação eficiente entre os componentes do sistema, especialmente para o monitoramento remoto e o controle da irrigação, o protocolo MQTT foi escolhido. O MQTT (Message Queuing Telemetry Transport) é um protocolo de comunicação leve, ideal para dispositivos IoT (Internet das Coisas), devido ao seu baixo consumo de banda e confiabilidade.

### 2.2.1. Protocolo MQTT

Figura 12 - Funcionamento do MQTT (Message Queue Telemetry Transport). Fonte: [Embarcados](#)



O padrão de troca de mensagens no MQTT segue o modelo publish/subscribe (publicador/subscritor), conforme descrito por **Barros (2015)**. Neste modelo, os elementos da rede que desejam receber informações subscrevem-se a elas, solicitando a um broker, o intermediário no processo de comunicação, que gerencie as publicações e subscrições. Da mesma forma, os elementos que desejam publicar informações o fazem através do broker, enviando-lhe as informações que possuem. Este paradigma não é novo e é observado em outros protocolos, como a troca de informação de controle (links) em redes Foundation Fieldbus.

### 2.2.2. Broker MQTT

Foi utilizado um broker MQTT público para este projeto, facilitando a comunicação entre os dispositivos. Optou-se pelo uso do broker público `broker.emqx.io`, devido à



sua disponibilidade e facilidade de configuração. Embora uma opção local pudesse ter sido considerada para maior controle sobre a infraestrutura, a escolha por um broker público foi feita para simplificar o processo de implementação e testes.

### **2.2.3. Configuração do MQTT**

A conexão com o broker MQTT foi estabelecida utilizando a porta TCP padrão 1883. Além disso, o projeto também suporta conexões WebSocket e SSL/TLS para garantir a segurança e flexibilidade da comunicação.

## **2.3. Descrição do Software Desenvolvido**

A implementação do sistema de irrigação automatizada foi dividida em várias etapas, conforme descrito a seguir:

### **2.3.1. Desenvolvimento do Circuito**

- **Montagem do Circuito:** Inclui a conexão do NodeMCU(ESP32), sensor de umidade, módulo relé e minibomba de água. Utilizando a placa de ensaio e jumpers para facilitar as conexões.
- **Teste do Circuito:** Verificação do funcionamento de cada componente individualmente e do sistema integrado.

### **2.3.2. Programação do NodeMCU(ESP32)**

- **Desenvolvimento do Código:** Utilizando o Visual Studio Code com a extensão PlatformIO, foram criados scripts para leitura do sensor de umidade, controle da bomba de água e comunicação via MQTT.
- **Teste do Código:** Realização de testes unitários para assegurar o funcionamento correto do software.

### **2.3.3. Configuração do MQTT e Conexão Wi-Fi**

- **Configuração do MQTT:** O servidor MQTT foi configurado para garantir a comunicação eficiente entre os dispositivos, permitindo a publicação de dados pelo sensor de umidade e o recebimento de comandos pelo NodeMCU(ESP32).

- **Configuração da Rede Wi-Fi:** O NodeMCU(ESP32) foi configurado com as credenciais da rede Wi-Fi utilizando scripts no Visual Studio Code com a extensão PlatformIO.

#### 2.3.4. Implementação e Integração

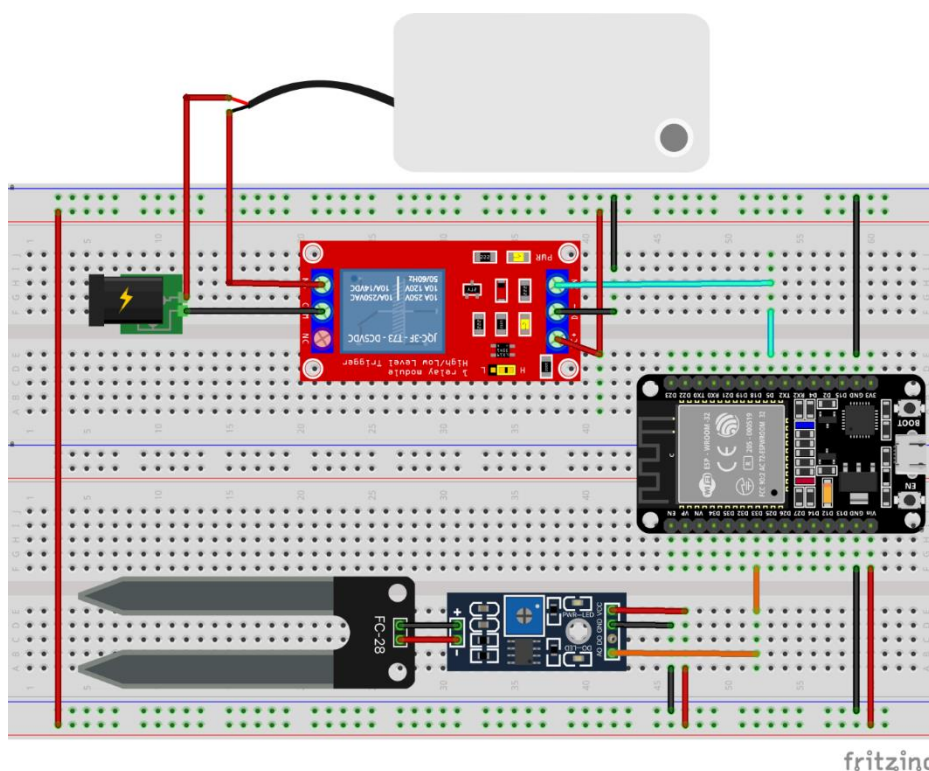
A última etapa do desenvolvimento inclui a integração de todos os componentes e sistemas, seguido por testes de campo para assegurar o funcionamento do sistema em um ambiente real. Os dados coletados são utilizados para ajustar os parâmetros de irrigação, garantindo eficiência e economia de água. Durante esta etapa, os seguintes processos foram realizados:

- **Integração de Hardware e Software:** Os componentes de hardware, como o NodeMCU(ESP32), sensor de umidade, módulo relé e minibomba de água, foram integrados com o software desenvolvido para garantir o funcionamento correto do sistema de irrigação automatizada.
- **Testes de Campo:** O sistema foi testado em campo para verificar sua operacionalidade em um ambiente real, garantindo que a irrigação fosse realizada conforme necessário com base nos dados de umidade do solo.
- **Ajustes e Otimizações:** Com base nos dados coletados durante os testes de campo, foram feitos ajustes e otimizações no sistema para melhorar sua eficiência e economia de água. Por exemplo a criação da lógica de bloqueio de bomba.

Com esses métodos de desenvolvimento e implementação detalhados, o projeto de irrigação automatizada está pronto para ser desenvolvido e implementado, proporcionando um sistema eficiente e confiável para o monitoramento e controle da umidade do solo.

## 2.4. Montagem do Protótipo

Figura 13 - Esquemático do Projeto. Fonte: Autores



O protótipo, desenvolvido no software Fritzing, ilustra de maneira detalhada a disposição e a interconexão dos componentes essenciais do sistema de irrigação automática. No centro da protoboard está o microcontrolador ESP32, escolhido como o cérebro do sistema devido às suas notáveis capacidades de processamento e conectividade Wi-Fi e Bluetooth. Este microcontrolador é crucial não apenas para controlar os atuadores, mas também para gerenciar a comunicação com o broker MQTT, que é fundamental para o monitoramento remoto e controle do sistema.

À esquerda do ESP32, encontra-se o módulo relé 5V. Este componente funciona como um interruptor eletrônico que, ao ser acionado pelo ESP32, ativa a minibomba de água. A minibomba é responsável por transportar água do reservatório para o sistema de irrigação, e sua operação é rigorosamente controlada para garantir eficiência e evitar desperdícios.

No canto superior direito do protótipo, o sensor de umidade do solo é conectado diretamente ao ESP32. Este sensor é fundamental para monitorar a umidade do solo, fornecendo dados precisos que são enviados ao microcontrolador. Com base nestas

informações, o ESP32 pode tomar decisões inteligentes sobre a necessidade de irrigação, otimizando o uso da água e preservando a saúde das plantas.

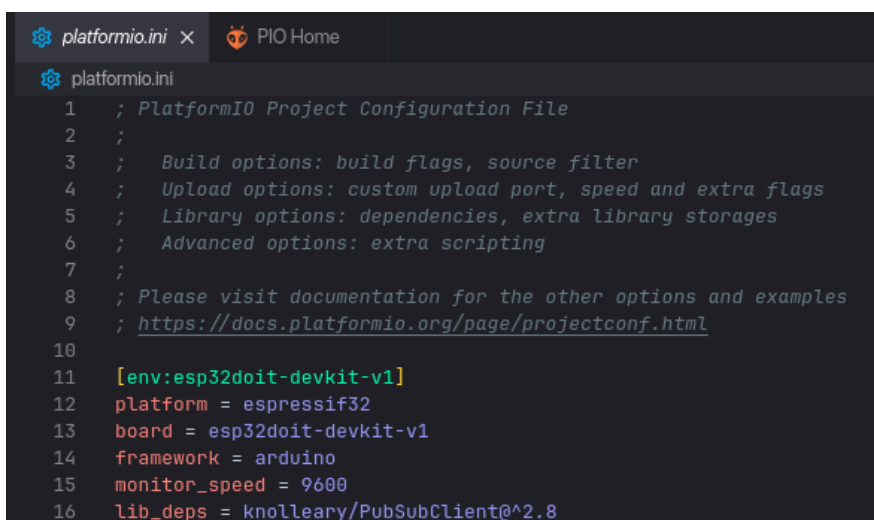
## 2.5. Funcionamento do Projeto

O cérebro do projeto é o NodeMCU ESP32, seu primeiro processo é a conexão com o WiFi, e posteriormente a Conexão como Broker para monitorar a umidade do solo e controlar a irrigação automática por meio de uma minibomba de água. A integração com um broker MQTT permite a comunicação e controle remoto, facilitando a interação com dispositivos móveis para monitoramento e comandos em tempo real. A seguir, detalha-se o funcionamento dos sensores e atuadores, bem como a integração com o broker MQTT.

### 2.5.1. Configuração do Ambiente de Desenvolvimento

O projeto utiliza o PlatformIO como ambiente de desenvolvimento integrado (IDE), o que facilita a gestão de bibliotecas e a compilação do código para o ESP32. O arquivo `platformio.ini` configura o ambiente de desenvolvimento da seguinte forma:

Figura 14 - Código do arquivo `platformio.ini`. Fonte: Autoria Própria

A screenshot of the PlatformIO IDE interface. The top bar shows 'platformio.ini' and 'PIO Home'. The main editor area displays the contents of the 'platformio.ini' file. The code is as follows:

```
1 ; PlatformIO Project Configuration File
2 ;
3 ; Build options: build flags, source filter
4 ; Upload options: custom upload port, speed and extra flags
5 ; Library options: dependencies, extra library storages
6 ; Advanced options: extra scripting
7 ;
8 ; Please visit documentation for the other options and examples
9 ; https://docs.platformio.org/page/projectconf.html
10
11 [env:esp32doit-devkit-v1]
12 platform = espressif32
13 board = esp32doit-devkit-v1
14 framework = arduino
15 monitor_speed = 9600
16 lib_deps = knolleary/PubSubClient@^2.8
```

- **platform:** Define a plataforma utilizada, neste caso, `espressif32`, que é específica para dispositivos ESP32.
- **board:** Especifica a placa utilizada, `esp32doit-devkit-v1`, uma das variantes do ESP32.
- **framework:** Indica o framework de desenvolvimento, `arduino`, facilitando o uso de bibliotecas e funções do Arduino.

- **monitor\_speed:** Define a velocidade do monitor serial em 9600 bps, que é a velocidade de comunicação entre o ESP32 e o monitor serial.
- **lib\_deps:** Lista as dependências de bibliotecas utilizadas no projeto. Aqui, a biblioteca PubSubClient é utilizada para comunicação MQTT.

### 2.5.2. Conexão WiFi e MQTT

No início do ciclo de funcionamento do projeto, a placa NodeMCU ESP32 se conecta à rede Wi-Fi configurada e ao broker MQTT especificado. O código de inicialização está localizado no arquivo WiFiHelper.cpp, onde a rede Wi-Fi é definida, e em MQTTHelper.cpp, onde a conexão com o broker MQTT é estabelecida. No setup() da aplicação principal (main.cpp), essas conexões são iniciadas:

### 2.5.3. Leitura da Umidade do Solo

O sensor de umidade de solo HL-69 está conectado ao ESP32, que lê os dados analógicos do sensor e os converte em umidade percentual. A função lerUmidade(), localizada em UmidadeHelper.cpp, realiza a leitura do sensor, faz uma média de múltiplas leituras para maior precisão e mapeia o valor para uma escala percentual:

Figura 15. Função de Ler Umidade. Fonte: Autoria Própria

```
int lerUmidade() {  
    int somaLeituras = 0;  
    for (int i = 0; i < NUM_LEITURAS; i++) {  
        somaLeituras += analogRead(PINO_UMIDADE);  
        delay(10);  
    }  
  
    int umidadePercentual = map(somaLeituras / NUM_LEITURAS, 0,  
                                4095, 100, 0);  
    return umidadePercentual;  
}
```

- somaLeituras / NUM\_LEITURAS: Calcula a média das leituras analógicas da umidade.
- map(valor, de\_min, de\_max, para\_min, para\_max): Mapeia o valor da média das leituras para uma nova faixa de valores.
  - valor: O valor a ser mapeado (média das leituras).
  - de\_min: O valor mínimo original (0, valor mínimo da leitura analógica).

- de\_max: O valor máximo original (4095, valor máximo da leitura analógica).
- para\_min: O valor mínimo da nova faixa (0, correspondente a 0% de umidade).
- para\_max: O valor máximo da nova faixa (100, correspondente a 100% de umidade).
- umidadePercentual: O valor mapeado para a escala percentual de umidade.

#### 2.5.4. Controle da Bomba de Água

A bomba de água é controlada por um relé conectado ao ESP32. A lógica de acionamento do relé está implementada em RelayHelper.cpp, que define as funções para ligar e desligar o relé:

**Figura 16. Funções do Relé. Fonte: Autoria Própria**

```
void inicializarRelay(PubSubClient& mqttClient) {
    pinMode(PINO_RELE, OUTPUT);
    desligarRelay(mqttClient);
}

void publicarMensagemRelayAcionado(PubSubClient& mqttClient) {
    mqttClient.publish("status/relay", "Acionado");
}

void publicarMensagemRelayDesligado(PubSubClient& mqttClient) {
    mqttClient.publish("status/relay", "Desligado");
}

void acionarRelay(PubSubClient& mqttClient) {
    digitalWrite(PINO_RELE, HIGH);
    publicarMensagemRelayAcionado(mqttClient);
}

void desligarRelay(PubSubClient& mqttClient) {
    digitalWrite(PINO_RELE, LOW);
    publicarMensagemRelayDesligado(mqttClient);
}
```

Quando a umidade do solo cai abaixo de um valor mínimo predefinido (UMIDADE\_MINIMA), o relé é acionado para ligar a bomba de água, como mostrado no loop principal do main.cpp:

Figura 17. Lógica Principal do Arquivo Main.cpp.Fonte: Autoria Própria

```
void loop() {
    // ... código de reconexão ao Wi-Fi e MQTT omitido para brevidade ...

    int umidadePercentual = lerUmidade();

    if (umidadePercentual >= 0 && umidadePercentual <= 100) {
        Serial.print("Umidade: ");
        Serial.print(umidadePercentual);
        Serial.println("%");

        enviarUmidadeMQTT(umidadePercentual, mqttClient);

        if (umidadePercentual <= UMIDADE_MINIMA && !bloqueador) {
            acionarRelay(); // Aciona o relé
            Serial.println("Relé acionado");
            listaUmidades.adicionarUmidade(umidadePercentual, bloqueador, mqttClient);
            listaUmidades.imprimirUmidadeArmazenada();

            if (!bloqueador && !limita_mensagem) {
                mqttClient.publish("status/reservatorio", "Normal - Bomba Desbloqueada");
                limita_mensagem = true;
            }

            if (bloqueador) {
                mqttClient.publish("status/reservatorio", "Vazio - Bomba Bloqueada");
                limita_mensagem = false;
            }
        } else {
            desligarRelay(); // Desliga o relé
            Serial.println("Relé desligado");
        }
    } else {
        Serial.println("Erro ao ler umidade. Tentando novamente...");
    }

    delay(1000);
}
```

#### 2.5.4.1. Lógica de Bloqueio de Relé/Bomba

Além das funcionalidades descritas anteriormente, o protótipo implementa uma lista encadeada para armazenar as últimas cinco leituras de umidade do solo. Esta lista é essencial para garantir que o sistema de irrigação funcione de forma eficiente e

segura, mesmo em condições onde o reservatório de água acabe ou o abastecimento seja interrompido.

A função `adicionarUmidade()` da classe `LinkedList` é responsável por adicionar uma nova leitura de umidade à lista. No entanto, ela também é projetada para limitar a lista a cinco ciclos de umidade. Isso é alcançado por meio da seguinte lógica:

1. Sempre que uma nova leitura de umidade é recebida, um novo nó é criado e adicionado à lista.
2. Se a lista já tiver cinco nós (ou seja, cinco leituras de umidade), o sistema verifica se as últimas cinco leituras foram iguais. Se sim, isso indica uma estagnação na umidade do solo.
3. Se a umidade do solo estiver estagnada por cinco ciclos consecutivos, o sistema bloqueia o funcionamento da bomba de água. Isso é feito definindo o bloqueador como verdadeiro, o que impede que a bomba de água seja acionada, mesmo que a umidade do solo caia abaixo do limite mínimo.

#### **2.5.4.2. Bloqueio do Sistema de Irrigação em Caso de Estagnação**

O bloqueio do sistema de irrigação é uma medida de segurança para evitar o desperdício de água e proteger as plantas contra danos causados por excesso de irrigação. Se a umidade do solo permanecer inalterada por cinco ciclos consecutivos de acionamento da bomba de água, o sistema de irrigação é bloqueado automaticamente.

Este recurso garante uma irrigação eficiente e econômica, protegendo as plantas contra danos e otimizando o uso de recursos hídricos. É uma demonstração do cuidado e da inteligência incorporados ao design do protótipo, visando a sustentabilidade e a eficiência no cultivo de plantas.

#### **2.5.5. Integração com o Broker MQTT**

O sistema utiliza o protocolo MQTT para publicar dados de umidade e receber comandos para desbloquear a bomba de água. A biblioteca `PubSubClient` é utilizada para gerenciar a comunicação MQTT. Para este projeto utilizados um Broker publico, e a função estruturada para realizar essa conexão com o Broker “`broker.emwx.io`” com a porta TCP 1883 foi a ‘`connectMQTT`’.



Figura 18. Função de Conexão como Broker. Fonte: Autoria Própria

```
const char* MQTT_SERVER = "broker.emqx.io";
const int MQTT_PORT = 1883;

WiFiClient espClient;
PubSubClient mqttClient(espClient);

void connectMQTT() {
  Serial.println("Conectando ao servidor MQTT...");
  mqttClient.setServer(MQTT_SERVER, MQTT_PORT);
  while (!mqttClient.connected()) {
    if (mqttClient.connect("ESP32Cliente")) {
      Serial.println("Conectado ao broker MQTT");
    } else {
      Serial.print("Falha na conexão, código de retorno=");
      Serial.print(mqttClient.state());
      Serial.println(", tentando novamente em 5 segundos");
      delay(5000);
    }
  }
}
```

O ESP32 publica a umidade lida no tópico "umidade" e recebe comandos no tópico "botao\_pressao". A função de callback, definida em MQTTHelper.cpp, é acionada quando uma mensagem MQTT é recebida:

Figura 19. Função callback, principal função MQTT com Client. Fonte: Autoria Própria

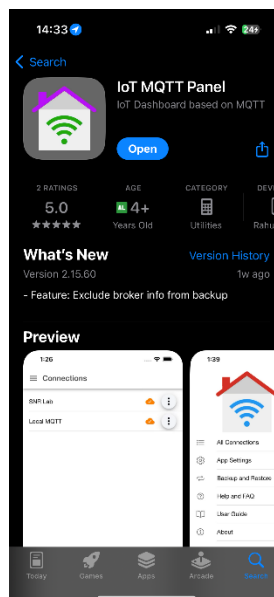
```
void callback(char* topic, byte* payload, unsigned int length) {
    Serial.println("Mensagem MQTT recebida:");
    Serial.print("Tópico: ");
    Serial.println(topic);
    Serial.print("Payload: ");
    Serial.write(payload, length);
    Serial.println();

    if (strcmp(topic, "botao_pressao") == 0) {
        if (payload[0] == '1') {
            bloqueador = false;
            Serial.println("Bloqueador desbloqueado");
            listaUmidades.limparLista(); // Limpa a lista de umidades
            Serial.println("Lista de umidades zerada após desbloqueio.");
        }
    }
}
```

### 2.5.6. Monitoramento e Controle Remoto

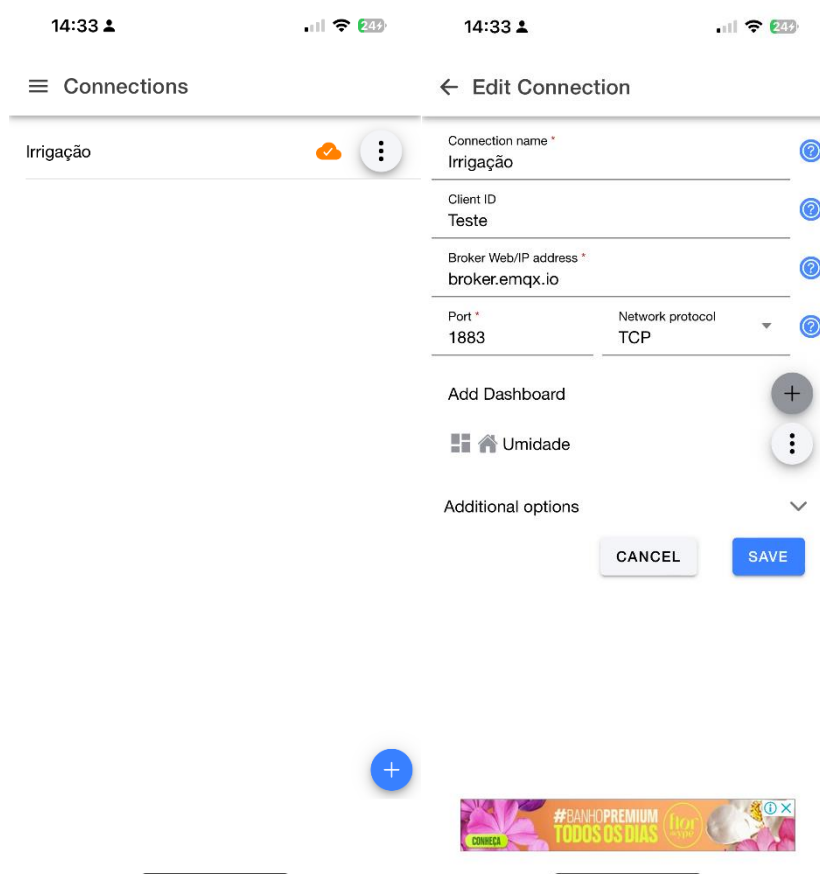
O protótipo é controlado remotamente através de um aplicativo móvel IoT MQTT Pnael, desenvolvido por IoT Dashboard based on MQTT

Figura 20. Imagem do Aplicativo na App Store. Fonte: Autoria Própria.



Esse aplicativo se conecta ao broker MQTT. Este aplicativo exibe dados em tempo real, incluindo um gráfico de gauge para a umidade do solo, um LED para monitorar o estado da bomba de água, um botão para desbloquear a bomba, e uma indicação do estado do reservatório (vazio ou normal). As mensagens publicadas pelo ESP32, como o estado do reservatório e a umidade do solo, são exibidas no aplicativo, permitindo um controle e monitoramento eficientes e remotos.

**Figura 21. Conexão MQTT com o Aplicativo. Fonte: Autoria Própria**



O layout do aplicativo móvel inclui:

- **Gráfico de Gauge de Umidade:** Visualiza a umidade do solo em tempo real.
  - Topic: umidade
  - Payload min: 0
  - Payload max: 100
- **LED Indicador:** Mostra o estado da bomba de água (acionada ou desligada).

- Status/relay
- Payload on: Acionado
- Payload off: Desligado
- **Botão de Desbloqueio:** Permite desbloquear a bomba de água manualmente.
  - **Topic:** botão\_pressao
  - **Payload:** 1
- **Indicador de Estado do Reservatório:** Exibe se o reservatório está "Vazio" ou "Normal".
  - **Topic:** status/reservatório
  - **Max visible line:** 2
  - **Max persistence:** 10

Figura 22. Dashboard via MQTT. Fonte: Autoria Própria



Imagens e gráficos do aplicativo móvel podem ser adicionados aqui para ilustrar a interface e o funcionamento remoto do sistema

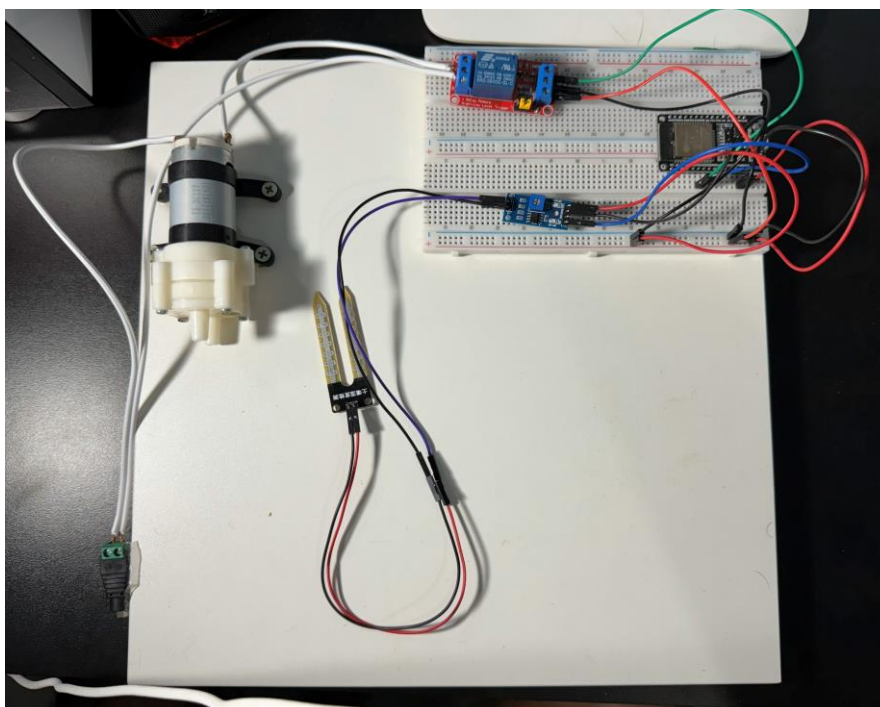
### 3. Resultados

#### 3.1. Descrição do Projeto

Abaixo estão as imagens do protótipo em funcionamento, juntamente com descrições detalhadas do que cada imagem representa:

##### 3.1.1. Imagem 1: Protótipo Completo

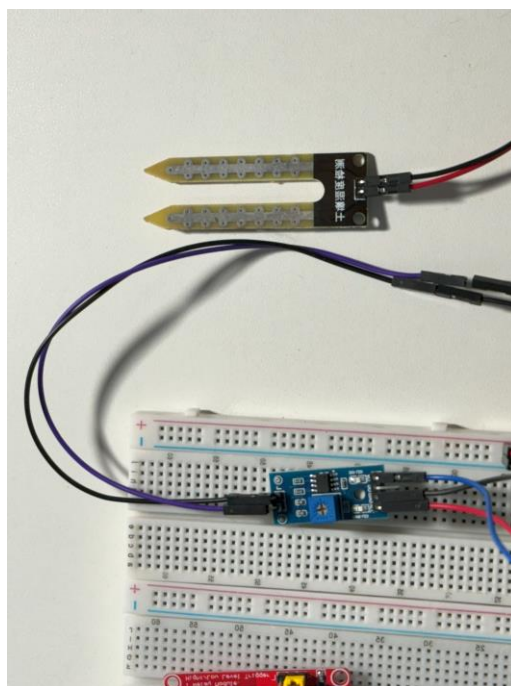
**Figura 23. Visão de Cima do Projeto Montado. Fonte: Autoria Própria**



##### 3.1.2. Imagem 2: Sensor de Umidade

Descrição: Close-up do sensor de umidade conectado ao NodeMCU ESP32. O sensor mede a umidade do solo e envia os dados para o ESP32 para posterior envio ao broker MQTT.

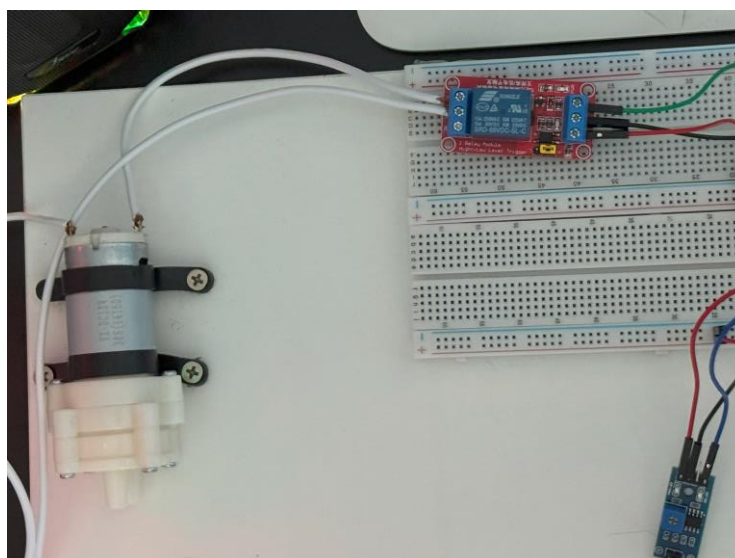
**Figura 24. Close-Up Sensor de Umidade. Fonte: Autoria Própria**



### 3.1.3. Imagem 3: Relé e Atuador

Descrição: Detalhe do módulo relé conectado ao NodeMCU ESP32. O relé é responsável por controlar a bomba d'água, que é acionada quando a umidade do solo cai abaixo de um certo limite.

**Figura 25. Relé e Bomba de Água. Fonte: Autoria Própria**



### 3.1.4. Imagem 4: Interface do Aplicativo Móvel

Figura 26. Dashboard via MQTT. Fonte: Autoria Própria



Descrição: Captura de tela do aplicativo móvel utilizado para monitorar e controlar o protótipo. O aplicativo mostra o gráfico de gauge de umidade, um indicador de LED para monitorar o estado do motor, um botão para desbloquear a bomba, e uma indicação do estado do reservatório (vazio ou normal).

### 3.2. Vídeo-Demonstração

No vídeo sobre o projeto de irrigação automática (disponível aqui: <https://youtu.be/C2EdTCbD1KY>), é apresentado o projeto de forma breve e visual. O vídeo cobre a configuração do projeto, incluindo os componentes utilizados (sensor de umidade, bomba d'água, relé, etc.), a estrutura do código e a lógica de funcionamento. São demonstradas a conexão Wi-Fi e MQTT, a lógica de monitoramento da umidade do solo, e o acionamento da bomba. Além disso, há uma demonstração prática do sistema em funcionamento, mostrando a integração dos componentes e a comunicação via aplicativo.

### 3.3. Medições de Tempo de Resposta

Abaixo está a tabela com os tempos médios de resposta dos sensores e atuadores. Foram realizadas quatro medições para cada sensor e atuador para calcular o tempo médio.

Núm. Medida	Sensor/Atuador	Tempo de Resposta (ms)
1	Sensor de Umidade	150
2	Sensor de Umidade	145
3	Sensor de Umidade	152
4	Sensor de Umidade	148
<b>Média</b>	<b>Sensor de Umidade</b>	<b>148.75</b>
1	Relé (Atuador)	200
2	Relé (Atuador)	195
3	Relé (Atuador)	202
4	Relé (Atuador)	198
<b>Média</b>	<b>Relé (Atuador)</b>	<b>198.75</b>

### 3.4. Link para o Repositório GitHub

O repositório GitHub contém a documentação completa do projeto, incluindo código fonte, esquemas de hardware, e instruções detalhadas para reprodução do projeto.

<https://github.com/Laryfernandes/NodeMCU-ESP32>

## 4. Conclusões

### 4.1. Alcanço dos Objetivos

Os objetivos propostos para este projeto foram alcançados com sucesso. O sistema desenvolvido integra um sensor de umidade do solo e um atuador (relé) para controlar uma bomba de água, utilizando o protocolo MQTT para comunicação com a internet.



A solução permite monitorar e controlar a umidade do solo remotamente através de um aplicativo móvel, garantindo a automação do processo de irrigação.

## 4.2. Principais Problemas Enfrentados e Soluções

Durante o desenvolvimento do projeto, alguns problemas significativos foram enfrentados e solucionados:

1. **Limitação de Mensagens MQTT:** Para evitar a sobrecarga do broker MQTT e do aplicativo móvel, foi necessário implementar uma lógica de limitação de mensagens. Utilizou-se uma variável de controle (`limita_mensagem`) para garantir que mensagens de status fossem enviadas somente quando necessário.
2. **Gerenciamento de Memória na Lista Encadeada:** A adição contínua de leituras de umidade à lista encadeada poderia causar problemas de memória. Implementamos uma função para limpar a lista quando necessário e limitar o tamanho da lista a cinco leituras, garantindo um gerenciamento eficiente da memória.
3. **Integração do Botão Virtual no Aplicativo Móvel:** Houve um desafio na integração do botão virtual do aplicativo móvel com o sistema para desbloquear a bomba de água. Foi solucionado ao configurar corretamente a função de callback MQTT para responder a mensagens específicas do tópico "botao\_pressao".

## 4.3. Vantagens e Desvantagens do Projeto

### 4.3.1. Vantagens:

- **Automação e Controle Remoto:** O projeto permite a automação da irrigação e o controle remoto da bomba de água, proporcionando conveniência e eficiência.
- **Monitoramento em Tempo Real:** O uso do protocolo MQTT permite o monitoramento em tempo real da umidade do solo e o status da bomba através de um aplicativo móvel.
- **Manutenção Fácil:** A modularidade do código, com a separação de funcionalidades em diferentes arquivos, facilita a manutenção e futuras expansões do sistema.

#### 4.3.2. Desvantagens:

- **Dependência de Conectividade WiFi:** O sistema depende da conectividade WiFi para funcionar corretamente. Em locais com sinal fraco ou ausência de WiFi, o sistema pode apresentar falhas.
- **Consumo de Energia:** O uso contínuo da ESP32 e do sensor de umidade pode resultar em um consumo significativo de energia, especialmente em aplicações onde a energia deve ser conservada.
- **Ausência de Sensor de Nível de Água:** Sem um sensor específico para monitorar o nível de água no reservatório, o sistema depende da lógica de estagnação de umidade, que pode não ser 100% precisa em todas as situações.

#### 4.4. Melhorias Futuras

Para aprimorar o projeto, algumas melhorias futuras podem ser implementadas:

1. **Adição de Funcionalidades de Economia de Energia:** Implementar modos de economia de energia na ESP32 e utilizar sensores de umidade com menor consumo de energia pode aumentar a eficiência do sistema.
2. **Implementação de Backups de Conectividade:** Integrar um módulo de comunicação celular (GSM) para garantir que o sistema continue funcionando em locais sem WiFi.
3. **Integração com Mais Sensores:** Adicionar sensores adicionais, como sensores de temperatura e luminosidade, pode proporcionar uma visão mais abrangente das condições ambientais, melhorando a precisão do sistema de irrigação.
4. **Aprimoramento da Interface do Usuário:** Melhorar a interface do aplicativo móvel para incluir gráficos mais detalhados e opções de configuração avançadas pode aumentar a usabilidade do sistema.
5. **Desenvolvimento de Algoritmos de Irrigação Inteligentes:** Incorporar algoritmos de aprendizado de máquina para prever as necessidades de irrigação com base em dados históricos e condições ambientais pode tornar o sistema mais inteligente e eficiente.
6. **Implementação de Sensor de Nível de Água:** Adicionar um sensor de nível de água no reservatório para monitorar diretamente a quantidade de água

disponível, evitando a dependência exclusiva da lógica de estagnação de umidade.

Essas melhorias podem tornar o sistema mais robusto, eficiente e adaptável a diferentes condições e requisitos de uso, proporcionando uma solução ainda mais eficaz para o monitoramento e controle da umidade do solo.

## Referências

BASSOI, Luís Henrique. Intensificação e sustentabilidade dos sistemas de produção agrícolas. Embrapa, 2024. Disponível em: [https://www.embrapa.br/olhares-para-2030/intensificacao-e-sustentabilidade-dos-sistemas-de-producao-agricolas/-/asset\\_publisher/MpEPEYHn8qxt/content/luis-henrique-bassoi?inheritRedirect=true](https://www.embrapa.br/olhares-para-2030/intensificacao-e-sustentabilidade-dos-sistemas-de-producao-agricolas/-/asset_publisher/MpEPEYHn8qxt/content/luis-henrique-bassoi?inheritRedirect=true). Acesso em: 3 maio. 2024.

DIB, Ana Emilia Hernandes. Desenvolvimento de um sistema de irrigação automatizado para plantas utilizando Arduino e Blynk. 2020. 103 f. Dissertação (Mestrado em Engenharia Elétrica) - Universidade de São Paulo, São Paulo, 2020. Disponível em: <https://repositorio.usp.br/directbitstream/83b99ccd-efb3-443d-ada0-36da436cd39b/ANA%20EMILIA%20HERNANDES%20DIB%20TCC20.pdf>. Acesso em: 24 mar. 2024.

ENCICLOPÆDIA BRITANNICA. Evaporation and Seepage Control. Disponível em: <https://www.britannica.com/technology/irrigation/Evaporation-and-seepage-control>. Acesso em: 07 maio. 2024.

ENCICLOPÆDIA BRITANNICA. Egypt. In: Britannica, The Editors of Encyclopaedia Britannica. Disponível em: <https://www.britannica.com/place/Egypt>. Acesso em: 07 maio. 2024.

HISTÓRIA DA IRRIGAÇÃO. [S.l.: s.n.], [s.d.]. Apresentação em PowerPoint. Disponível em:

[https://www.ufrb.edu.br/neas/images/TALES\\_CCA039/Tales\\_Historia\\_da\\_Irigacao\\_v4.ppt](https://www.ufrb.edu.br/neas/images/TALES_CCA039/Tales_Historia_da_Irigacao_v4.ppt). Acesso em: 23 fev. 2024.

KUONGSHUN ELECTRONIC SHOP. Tutoriais. Disponível em: <https://kuongshun.com/pages/tutorials>. Acesso em: 06 maio 2024.

MANEJO DE RECURSOS HÍDRICOS. [S.l.: s.n.], 2000. Apostila IT 157. Disponível em: <http://www.ufrj.br/institutos/it/deng/leonardo/downloads/APOSTILA/Apostila%20IT%20157/it157-Manejo2000.pdf>. Acesso em: 23 fev. 2024.

MEDIEIROS, Pedro Henrique Silva. Sistema de irrigação automatizado para plantas caseiras. João Monlevade: Universidade Federal de Ouro Preto, Instituto de Ciências Exatas e Aplicadas, Departamento de Computação e Sistemas, 2018. 94 f. Monografia.

RELATÓRIO MUNDIAL DAS NAÇÕES UNIDAS SOBRE O DESENVOLVIMENTO DOS RECURSOS HÍDRICOS 4. [S.l.: s.n.], 2012. Disponível em: [https://www.icmbio.gov.br/educacaoambiental/images/stories/biblioteca/rio\\_20/wwdr4-fatos-e-dados.pdf](https://www.icmbio.gov.br/educacaoambiental/images/stories/biblioteca/rio_20/wwdr4-fatos-e-dados.pdf). Acesso em: 23 fev. 2024.

SRITU HOBBY. How to make a plant watering system with ESP32 board and Blynk app. 2023. Disponível em: <https://srituhobby.com/how-to-make-a-plant-watering-system-with-esp32-board-and-blynk-app/>. Acesso em: 24 mar. 2024.

STRAUB, Matheus Gebert. Projeto Arduino de irrigação automática: sua planta sempre bem cuidada. Usina Info, 17 jul. 2019. Disponível em: <https://www.usinainfo.com.br/blog/projeto-arduino-de-irrigacao-automatica-sua-planta-sempre-bem-cuidada/>. Acesso em: 23 fev. 2024.

TAVARES, Renato. Irrigação Automático Arduino. GitHub, [s.d.]. Disponível em: [https://github.com/renatotvs/irrigacao\\_automatgico\\_arduino\\_nodemcu\\_esp8266\\_iot\\_al](https://github.com/renatotvs/irrigacao_automatgico_arduino_nodemcu_esp8266_iot_al) exa. Acesso em: 23 fev. 2024.

UN. Objetivos de Desenvolvimento Sustentável. Disponível em: <https://brasil.un.org/pt-br/sdgs>. Acesso em: 07 maio. 2024.