UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
ESCOLA DE ENGENHARIA
ENGENHARIA DE CONTROLE E AUTOMAÇÃO

RICHARD SPIEWECK ADOLPHS - 218777

PROJETO: SISTEMA DE IRRIGAÇÃO AUTOMÁTICO SUPERVISIONADO PARA HORTAS

Porto Alegre
2020

RICHARD SPIEWECK ADOLPHS - 218777

PROJETO: SISTEMA DE IRRIGAÇÃO AUTOMÁTICO SUPERVISIONADO PARA HORTAS

Trabalho de Conclusão de Curso (TCC-CCA) apresentado à COMGRAD-CCA da Universidade Federal do Rio Grande do Sul como parte dos requisitos para a obtenção do título de *Bacharel em Eng. de Controle e Automação* .

ORIENTADOR:
Prof. Dr. Renato Ventura Bayan Henriques

Porto Alegre
2020

RICHARD SPIEWECK ADOLPHS - 218777

PROJETO: SISTEMA DE IRRIGAÇÃO AUTOMÁTICO SUPERVISIONADO PARA HORTAS

Este Trabalho de Conclusão de Curso foi julgado adequado para a obtenção dos créditos da Disciplina de TCC do curso *Engenharia de Controle e Automação* e aprovado em sua forma final pelo Orientador e pela Banca Examinadora.

Orientador: _____
Prof. Dr. Renato Ventura Bayan Henriques, UFRGS
Doutor pela Universidade Federal de Minas Gerais, Belo Horizonte, Brasil

Banca Examinadora:

Prof. Dr. Renato Ventura Bayan Henriques, UFRGS
Doutor pela Universidade Federal de Minas Gerais, Belo Horizonte, Brasil

Prof. Dr. Marcelo Götz, UFRGS
Doutor pela Universität Paderborn, Paderborn, Alemanha e pela Universidade Federal do Rio Grande do Sul, Porto Alegre, Brasil

Prof. Dr. Alcy Rodolfo dos Santos Carrara, UFRGS
Doutor pela Mcmaster University, Hamilton, Canadá

Marcelo Götz
Coordenador de Curso
Engenharia de Controle e Automação

Porto Alegre, novembro de 2020.

DEDICATÓRIA

Dedico este trabalho aos meus pais, que fizeram o impossível para que nunca me faltasse estudo, comida, lar e amor.

AGRADECIMENTOS

Agradeço, em primeiro lugar, aos meus pais Doris e Carlos, por todo apoio que recebi ao longo da vida para lutar pelos meus sonhos e por respeitarem meu tempo.

À minha irmã, Caroline, por ser a melhor amiga que já tive e me acompanhar nas trapalhadas da vida.

Ao meu companheiro, Gabriel, por todo suporte emocional, carinho e compreensão que foi essencial para que eu saísse da inércia e crescesse profissionalmente.

À minha *Alma Mater*, Universidade Federal do Rio Grande do Sul, pela oportunidade de estudar gratuitamente e me tornar Engenheiro em um local que é referencia no país.

Aos professores que me acompanharam nesta trajetória, formando meu senso crítico e minha base de conhecimento.

Ao meu orientador, Professor Renato Henriques, por me guiar em momentos de dúvida durante a concepção deste trabalho.

Aos colegas, pelos grupos de estudo e trocas de ajuda, muito importantes para que este momento chegasse.

A todos meus amigos mais queridos, cujos nomes não citarei para não ser injusto, por serem a base do meu ser e me ajudarem a relaxar em momentos de estresse com boas risadas e conversas.

À minha revisora, Maria Amália, pela amizade de muitos anos e pelo empenho na revisão deste trabalho.

A todos os 128 respondentes da minha pesquisa de mercado, a opinião sincera de vocês foi essencial.

Às empresas que me acolheram no início da minha vida profissional, onde tive a oportunidade de entrar em contato com profissionais excelentes e de aprender coisas que a universidade não ensina.

RESUMO

Este trabalho demonstra a implementação de um sistema automático de irrigação para hortas orgânicas utilizando o microcontrolador NodeMCU e o protocolo de comunicação MQTT (*Message Queue Telemetry Transport*). Durante a metodologia, serão apresentados os componentes utilizados, como foram selecionados, a topologia do sistema e os algoritmos de implementação do controle. Será também apresentado o *Dashboard* gerado para acompanhamento dos indicadores. Ao final, espera-se obter um sistema confiável, customizável, de simples implementação e baixo custo para cuidados remotos às plantas.

Palavras-chave: Engenharia, controle e automação, automação, MQTT, eletrônica, instrumentação, dashboard, horta automatizada.

ABSTRACT

This work shows the implementing of an automatic irrigation system for organic gardens using the NodeMCU microcontroller and the MQTT (Message Queue Telemetry Transport) communication protocol. During the methodology will be presented the components used, the selection process, the system topology and the algorithms of the control's implementing. The Dashboard for tracking the indicators will be also presented. By the end, it is expected to obtain a reliable, customizable and easily implementable low budget system to take remote care of one's plants.

Keywords: engineering, control and automation, automation, MQTT, electronics, instrumentation, dashboard, organic garden.

SUMÁRIO

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE ABREVIATURAS

E/S	Entrada/Saída
I/O	*Input/Output*
IoT	*Internet of Things*
IDE	*Integrated Development Environment*
GPIO	*General Purpose I/O*
PWM	*Pulse Width Modulation*
HTTP	*Hypertext Transmission Protocol*
MQTT	*Message Queue Telemetry Transport*
DIY	*Do It Yourself*

1 INTRODUÇÃO

Cuidar de plantas em casa, nem sempre é uma tarefa fácil. Por vezes temos que pesquisar sobre como irrigá-las, que tipo de solo utilizar, quanta luz de sol elas precisam, entre outros fatores que podem afetar seu crescimento e desenvolvimento saudável.

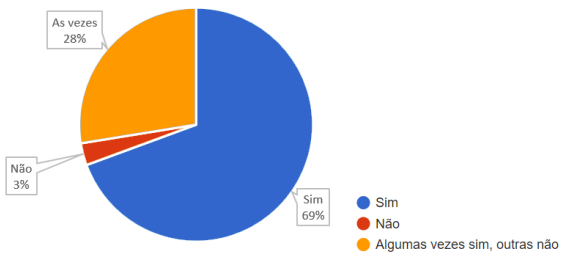
Aprender a cuidar de plantas não é algo simples, mas torna-se ainda mais complicado quando viajamos. Em viagens longas, é comum que criadores de plantas as deixem sob os cuidados de outrem, a fim de não deixá-las morrer durante sua ausência, mas nem sempre isto é possível.

Diante deste cenário, uma ideia surgiu: criar um sistema que permitisse irrigar de forma automática as plantas nos momentos de ausência. Com essa ideia em mente, tornou-se interessante realizar uma breve pequisa que analisasse a necessidade real de tal sistema.

Com a utilização da plataforma *Google Forms*, foi montada e posteriormente divulgada nas principais redes sociais (Whatsapp, Facebook e Instagram), uma pesquisa perguntando algumas características e hábitos das pessoas que criam plantas. Com base nestes dados, algumas informações relevantes puderam ser observadas:

Das 128 pessoas que responderam ao formulário, 98 dizem criar plantas em casa, e destas, 96,9% (95 pessoas) disseram se preocupar sempre ou às vezes com o que acontecerá com suas plantas durante a viagem (Figura 1). 69,4% (68 pessoas) disseram já ter deixado suas plantas sob os cuidados de outras pessoas (Figura 2, gráfico à esquerda).

Figura 1: *Preocupação ao viajar.*

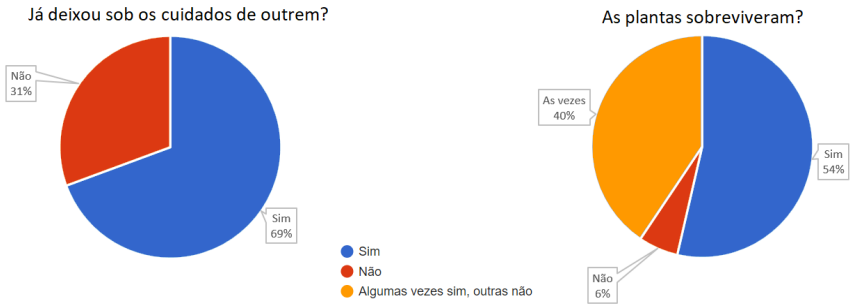


Fonte: Autor.

Entre as que deixaram suas plantas com outros, 46,4% (32 pessoas) revelaram que suas plantas por vezes não sobreviviam aos seus períodos de ausência, como denota a Figura 2 no gráfico à direita, e, mesmo os que disseram que sobreviviam sempre (37 pessoas), também demonstraram interesse no sistema, 94,6% (35 pessoas) deles responderam ter interesse e 89,2% (33 pessoas) comprariam um sistema assim, a Figura 3 demonstra os resultados desta estratificação.

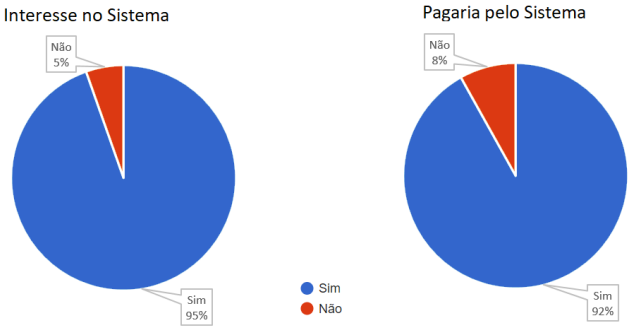
Apesar do grande interesse geral revelado pela entrevista, 11,2% (11 pessoas) dos respondentes marcaram que não pagariam por este sistema. Contudo, os dados validam

Figura 2: *Cuidados durante a viagem.*



Fonte: Autor.

Figura 3: *Interesse entre aqueles cujas plantas sobreviveram aos cuidados de outrem.*



Fonte: Autor.

o nicho de mercado para o desenvolvimento. Os respondentes da pesquisa eram de sete estados do Brasil e se dividiram em seis faixas etárias distintas.

O desenvolvimento deste projeto passará pela análise da bibliografia, apontando trabalhos relevantes na área, e pela seleção de materiais, protocolos de comunicação e sistemas de armazenamento de dados. Busca-se obter um sistema que irrigue as plantas de forma automática e permita o monitoramento remoto dos indicadores que serão definidos no decorrer deste trabalho. Os resultados serão demonstrados com base no atendimento dos objetivos traçados nas próximas seções deste.

2 REVISÃO DA LITERATURA

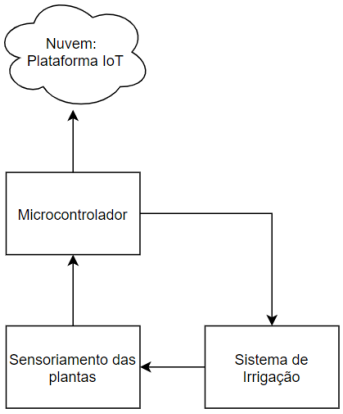
Para a construção deste trabalho, serão analisados os elementos eletrônicos com acesso à internet de acesso mais fácil, em termos monetários, e de utilização mais simples, e também os principais protocolos de comunicação em termos de IoT. Com este intuito, os seguintes estudos foram analisados e levados em conta para a construção do presente trabalho:

O manual do Prof. Dr. Renato Ventura Bayan Henriques (HENRIQUES, 2019), orientador do autor deste trabalho, descreve uma série de métodos e protocolos utilizados em IoT. O trabalho de conclusão do colega Ethar André Sirena Teixeira (TEIXEIRA, 2019), que trata de uma aplicação diferente do mesmo protocolo escolhido. O trabalho de mestrado de Mariel de Los Ángeles Proz (LOS ÁNGELES PROZ, 2020), que comenta sobre características do ambiente para cultivo de temperos em ambiente doméstico, com foco em salsa e manjericão. Além destes trabalhos, serão ainda consultados *Datasheets* e manuais de fabricantes, a fim de comparar componentes por suas características técnicas.

3 METODOLOGIA

O objetivo deste trabalho é trazer uma solução de baixo custo e simples implementação para pessoas que tenham interesse em IoT possam cuidar de suas plantas de onde quer que estejam. A arquitetura proposta para este sistema está descrita na Figura 4.

Figura 4: *Arquitetura pretendida.*



Fonte: Fonte: Autor.

Para criar o sistema desejado, é necessário realizar os seguintes passos:

- Definir os componentes utilizados;
- Definir o protocolo de comunicação;
- Estabelecer a comunicação entre os componentes e a internet;
- Configurar a troca de dados e as ações do microcontrolador;
- Criar um *dashboard* que permita acompanhar os dados e ações do sistema.

3.1 Definições de Componentes

Esta seção aborda o processo de escolha dos componentes eletrônicos do sistema, focando nos sensores e no atuador utilizados.

3.1.1 Microcontrolador

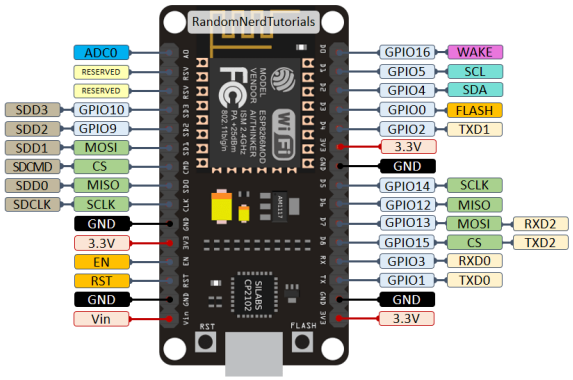
O microcontrolador é o cérebro do sistema, este componente será responsável por fazer a ligação entre os sensores, os acionadores e o supervisório do sistema (*Dashboard*). É importante que o microcontrolador selecionado seja capaz de acessar a internet diretamente (sem módulos adicionais) ou indiretamente (com módulo wifi ou ethernet, por exemplo).

Como se trata de um projeto focado em baixo custo e simplicidade de utilização, é conveniente que o microcontrolador seja muito bem documentado e de fácil acesso, neste contexto, a família de controladores Arduíno seria ideal. Contudo, as placas mais acessíveis de Arduíno não contêm componentes de acesso à rede, tomando como base o Arduíno Uno, Arduíno Mega e Arduíno Nano, todos estes com os quais o autor já teve contato necessitam de módulo extra para aceso a rede, gerando custos adicionais de aquisição.

Surge, neste cenário, uma placa de desenvolvimento em ascensão, o NodeMCU. Com circuito integrado de rede wifi, esta placa baseada no controlador ESP8266 12E se prova bastante versátil para aplicações de IoT. (HENRIQUES, 2019) Discute brevemente em seu manual sobre o porquê de esta placa ser uma boa escolha. Por ser de pequeno dimensional e programável na IDE Arduino através da linguagem C++, esta placa é muito bem documentada e há uma quantidade considerável de bibliotecas para sua programação, além de ser ideal para projetos que necessitem de economia de espaço. Ademais, este componente acessa os protocolos mais básicos de rede, funcionando como ponto de acesso, cliente e até mesmo como ambos ao mesmo tempo.

Um ponto negativo, porém, é que se trata de um controlador que, sem um sistema operacional dedicado, só pode executar uma *thread* por vez, o que impede a execução de tarefas simultâneas dentro do escopo deste trabalho.

Figura 5: *Mapeamento de GPIOs da placa NodeMCU.*



Fonte: https://randomnerdtutorials.com/esp8266-pinout-reference-gpios/, Acesso em: 27 set. 2020.

A Figura 5 mostra o endereçamento das conexões deste controlador. A placa é constituída de 16 canais digitais, sendo 9 com funcionamento PWM e de um canal com conversor analógio/digital de 10 bits. Todos estes canais operam como entrada e saída. O nível de tensão de alimentação para esta placa é entre 7 e 12V, com tensão de operação de 3,3V e suas GPIOs aceitam uma corrente máxima de 12mA.

3.1.2 Seleção dos sensores

Como o objetivo deste desenvolvimento é alcançar uma horta automatizada com sensoriamento e irrigação, os componentes selecionados estão diretamente relacionados às variáveis que se deseja utilizar no controle.

Mariel Proz discute, em seu trabalho, algumas das varáveis mais importantes para o desenvolvimento saudável de Salsa e Manjericão, sendo estas plantas as que serão utilizadas neste estudo como base para a escolha de sensores. Entre as variáveis críticas descritas por ela no trecho em que fala sobre a cultura destas plantas em ambientes domésticos, estão temperatura, umidade do ar, iluminação e irrigação (LOS ÁNGELES PROZ, 2020).

Com base nestas variáveis, os sensores selecionados foram:

3.1.2.1 DHT22

Embora tenha um custo maior que seu irmão DHT11, o sensor DHT22 possui maior precisão nas leituras da temperatura e umidade do ar, conforme a Tabela 1, ademais, aceita uma faixa de 0 a 100% na leitura da umidade do ar, contra 20 a 90% do DHT11, conforme a Tabela 2, ambas montadas a partir dos *datasheets* dos componentes.

Tabela 1: *Tabela de comparação das precisões dos sensores.*

	DHT11	DHT22
Temperatura	$\pm 2^{o}C$	$\pm 0,5^{o}C$
Umidade do Ar	$\pm 5\%$	$\pm 2\%$

Fonte: Autor.

Tabela 2: *Tabela de comparação das faixas de operação dos sensores.*

	DHT11	DHT22
Temperatura	$0\ a\ 50^{o}C$	$-40^{o}C\ a\ 80^{o}C$
Umidade do Ar	$20\%\ a\ 90\%$	$0\ a\ 100\%$

Fonte: Autor.

Desta forma, o sensor DHT22 foi selecionado porque uma maior precisão dentro de uma faixa maior de valores de leitura implica um controle mais sensível e mais abrangente. Este sensor não requer condicionamento adicional de sinal, podendo ser diretamente conectado à placa de desenvolvimento.

Endossando a escolha do sensor DHT22 para o projeto, temos o Trabalho de Conclusão de Ethar Teixeira (TEIXEIRA, 2019), que empregou o mesmo sensor para a leitura destas variáveis. Uma imagem deste sensor pode ser vista na Figura 6.

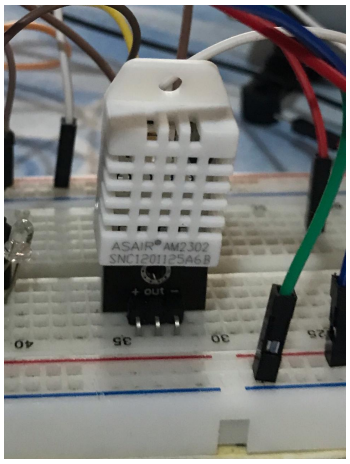
3.1.2.2 HW-080

Este sensor de baixo custo, denotado na Figura 7, representa uma excelente alternativa para medir a umidade do solo. Existe um grande volume de projetos na internet que aplica este sensor.

Um cuidado necessário ao aplicar este sensor é que sempre deve haver calibragem para o solo utilizado. Sua medição é resistiva, uma vez que é composto de dois eletrodos que, através da umidade e sais minerais presentes no solo, estabelecem um circuito. Por conta disso, a composição do solo pode afetar os valores mínimos e máximos alcançados de acordo com sua granulação e composição de sais minerais, por exemplo.

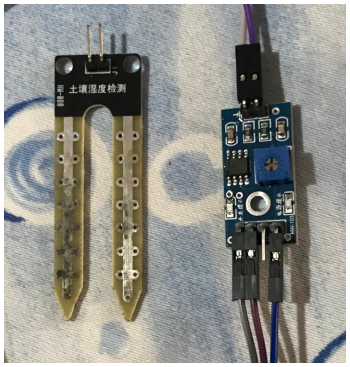
Este sensor acompanha um condicionador de sinais dedicado com regulagem de sensibilidade e possui duas saídas de sinal: uma analógica, que retornará o valor lido de umidade, e outra digital, que retornará sinal alto (3V3) ou baixo (0V) de acordo com o

Figura 6: *Sensor de temperatura e umidade do ar DHT22.*



Fonte: Autor.

Figura 7: *HW-080 e condicionador de sinal.*



Fonte: Autor.

setpoint de umidade regulado. Para este projeto, optou-se por utilizar a saída analógica, embora um resultado satisfatório pudesse igualmente ser alcançado caso a saída digital fosse empregada.

3.1.2.3 Fotoresistor

Este sensor foi escolhido para as leituras de luminosidade por ser barato e abundante em lojas de componentes. Sua aplicação requer pouco condicionamento (apenas baseado em resistores).

Após a calibração, é possível obter os valores mínimo e máximo de leitura do componente e utilizá-lo para medir a incidência solar na planta. Embora sua curva de resistência em função da luminosidade não seja linear, para a aplicação em questão é razoável considerar que é, uma vez que se deseja identificar apenas quando a incidência é demasiada grande, como será exposto nos algoritmos presentes nas próximas sessões deste trabalho.

3.1.3 Mini Bomba d'água RS385

Trata-se de uma bomba de 12VDC de alto fluxo (até 2l/min). Seu controle será realizado por meio de um módulo de relé. Foi selecionada principalmente por seu baixo custo e simplicidade de utilização, sendo uma bomba bastante comum nas principais lojas online de eletrônica e também empregada com frequência por entusiastas de "faça você mesmo" (DIY).

Figura 8: *Bomba RS385.*

Fonte: Autor.

3.1.4 Demais componentes

Os demais componentes utilizados para a montagem do projeto foram uma fonte 12VDC (Figura 9a) para a bomba e um relé de acionamento (Figura 9b), além dos fios para conexões e das mangueiras. As mangueiras integravam o kit da bomba adquirida. Fios, conectores, relé e fonte já compunham o material do autor.

3.2 Protocolo

Hoje em dia, existem diversos protocolos de comunicação baseados em internet. Entre eles encontram-se por exemplo: Sigfox, MQTT, HTTP, WiFi, entre outros. Para a realização deste trabalho, foi dado o enfoque no protocolo MQTT, por conta de sua implementação simples, com a possibilidade de criação de *dashboards* nos *brokers* mais comuns e por ser um protocolo bastante documentado.

Figura 9: *Fonte e módulo relé utilizados.*

(a) *Fonte 12VDC.*

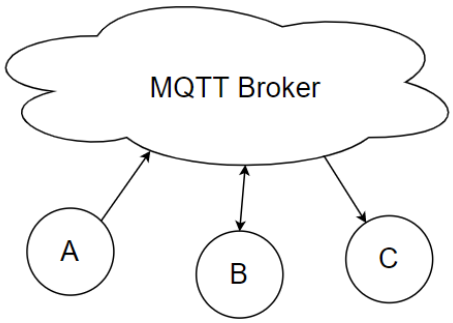
(b) *Módulo Relé.*

Fonte: Autor.

O MQTT foi desenvolvido no final da década de 90 Andy Stanford-Clark, da IBM, e Arlen Nipper, da Eurotech (HENRIQUES, 2019). É um protocolo que se baseia na arquitetura publicação/inscrição, "voltado para dispositivos restritos e redes inseguras, com baixa largura de banda e alta latência"(HENRIQUES, 2019).

Ao estabelecer comunicação com o *broker* (gerenciador da comunicação), o cliente pode realizar duas tarefas: publicar e inscrever-se em um tópico. Tópicos funcionam como canais para a recepção de dados, ao publicar em um tópico, este receberá a informação e opcionalmente poderá mantê-la. Ao inscrever-se em um tópico, o cliente passará a “ouvir” as publicações que ocorrerem nele. Desta forma, é possível operar de forma desacoplada, ou seja, os clientes não precisam de uma conexão direta entre si e podem receber dados de qualquer lugar do mundo que acessem os tópicos registrados.

Figura 10: *Exemplo de topologia do sistema em MQTT.*



Fonte: Autor.

No exemplo denotado na Figura 10, o cliente A publica dados em um tópico no qual os clientes B e C estão inscritos, por sua vez, o cliente B também publica no mesmo tópico. A troca de dados pode ocorrer em qualquer sentido, permitindo que, por exemplo, possa existir um cliente mestre que processa as informações e controla os clientes escravos através da nuvem do *broker*. Neste mesmo exemplo, A pode representar um sensor, B o mestre que recebe o dado do sensor e processa, e C pode representar um atuador.

3.3 *Broker* MQTT

Durante o desenvolvimento, foram realizados testes com alguns *brokers* MQTT. Boa parte deles não suporta mais contas gratuitas, de forma que o número de opções se afunilou para apenas um que fornecia gratuitamente uma solução que contemplava o autor, o Adafruit IO.

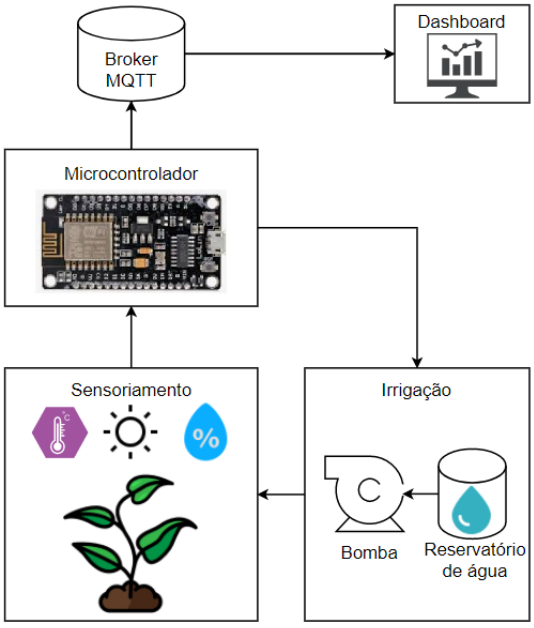
Este *broker* suporta, em sua versão gratuita, até dez tópicos (*feeds*) e até cinco *dashboards*, a uma taxa de transações (publicação/inscrição) máxima de 30 por minuto. Os dados publicados nos tópicos deste *broker* ficam armazenados por até 30 dias em nuvem.

Uma vez que não é necessária uma tomada de leituras dos sensores com uma taxa tão grande, considera-se esta solução mais do que suficiente para atender os objetivos do projeto. Ademais, os *dashboards* criáveis neste *broker* são customizáveis e comportam os mais comuns elementos esperados, como gráficos, botões, *streams*, chaves, entre outros.

3.4 Topologia e Algoritmos

A Figura 11 denota o sistema concebido para implementar a lógica de controle da horta automatizada. Os sensores estão acoplados ao NodeMCU, que publica nos tópicos de cada variável o valor desta cerca de duas a três vezes por minuto, este tempo varia em função da rotina de leitura do DHT22, que pode levar até dois segundos para efetuar a leitura de cada uma de suas grandezas medidas (LIU, 2020).

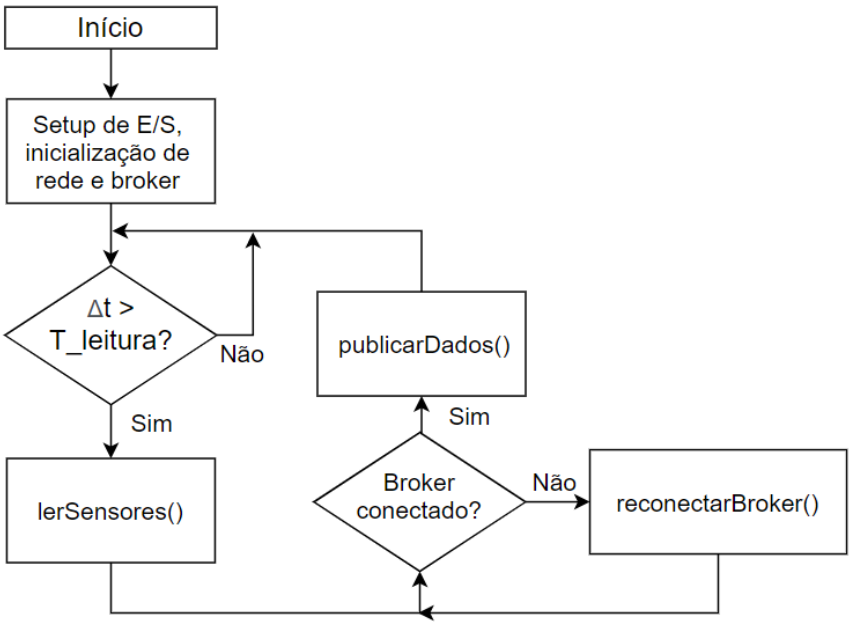
Figura 11: *Topologia do sistema concebido.*



Fonte: Autor.

A Figura 12 demonstra o algoritmo empregado para a leitura e publicação dos valores dos sensores. Após a inicialização das E/Ss do microcontrolador, da rede e do *broker* MQTT, a cada intervalo predefinido é efetuada a leitura dos sensores e então é checada se a conexão com o *broker* está ativa, caso negativo a conexão é reestabelecida, caso positivo os valores dos sensores são enviados ao *broker*, onde são adicionados ao *dashboard*. O *broker* em si não realiza processamentos de dados, estes são feitos no próprio microcontrolador,

Figura 12: *Algoritmo de publicação com pseudocódigo.*



Fonte: Autor.

que possui rotina de funcionamento automático.

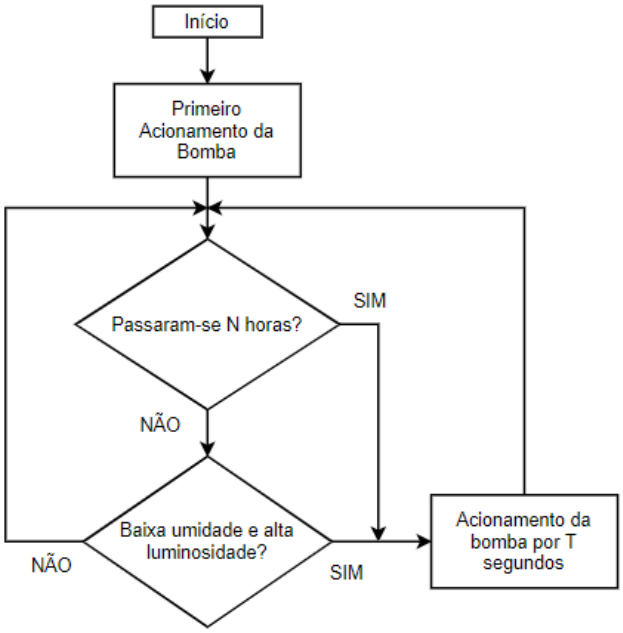
Na Figura 13, está denotada a rotina de funcionamento da bomba. Ao ligar o sistema, a bomba irriga as plantas uma vez e depois volta a irrigar de N em N horas por um período de T segundos. Os valores padrão do sistema para N e T são de 12 horas e três segundos, respectivamente, mas podem ser alterados por condições de temperatura, umidade e luminosidade. Quando a bomba é acionada, há publicação em tópico dedicado para rastrear as irrigações.

Como demonstra a Figura 14, o período T (bomba ligada) pode ser acrescido em até dois segundos conforme as condições de temperatura, luminosidade e umidade do solo, para manter as plantas saudáveis de acordo com as condições do ambiente. Estas leituras junto ao ajuste no tempo de acionamento da bomba funciona, de certa forma, como um sistema rudimentar em malha fechada. Neste caso, não há um *setpoint* específico, mas o sinal de controle - aqui representado pelo tempo de acionamento da bomba - é ajustado de acordo os dados dos sensores.

Os parâmetros foram definidos com base em observações realizadas pelo autor, de maneira empírica, com base na aparência das folhas de suas plantas. O algoritmo desenvolvido é ideal para as culturas do autor, mas não necessariamente para qualquer planta, uma vez que cada espécie conta com diferentes necessidades de irrigação (LOS ÁNGELES PROZ, 2020).

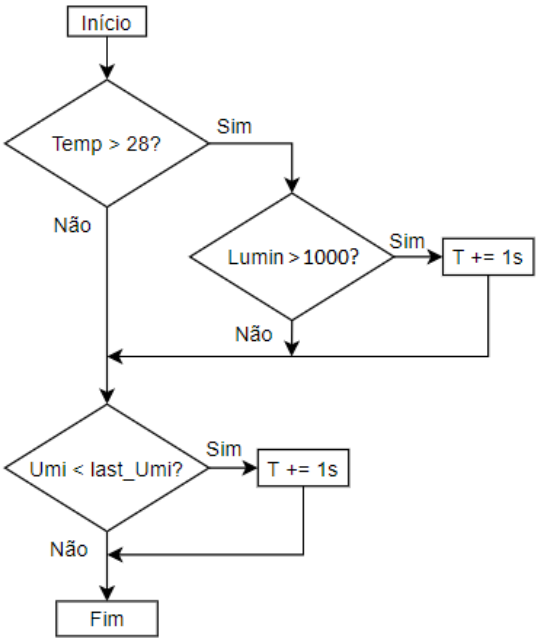
Embora uma parametrização mais eficiente possa ser gerada a partir de estudos específicos espécie a espécie e seja desejável que um produto baseado neste sistema contenha um banco de dados para cada uma, este tema foge o escopo deste trabalho, por isso a parametrização empírica foi considerada como suficiente para a prova de conceito.

Figura 13: *Algoritmo de Acionamento da Bomba.*



Fonte: Autor.

Figura 14: *Algoritmo de Adição de Tempo de Acionamento.*

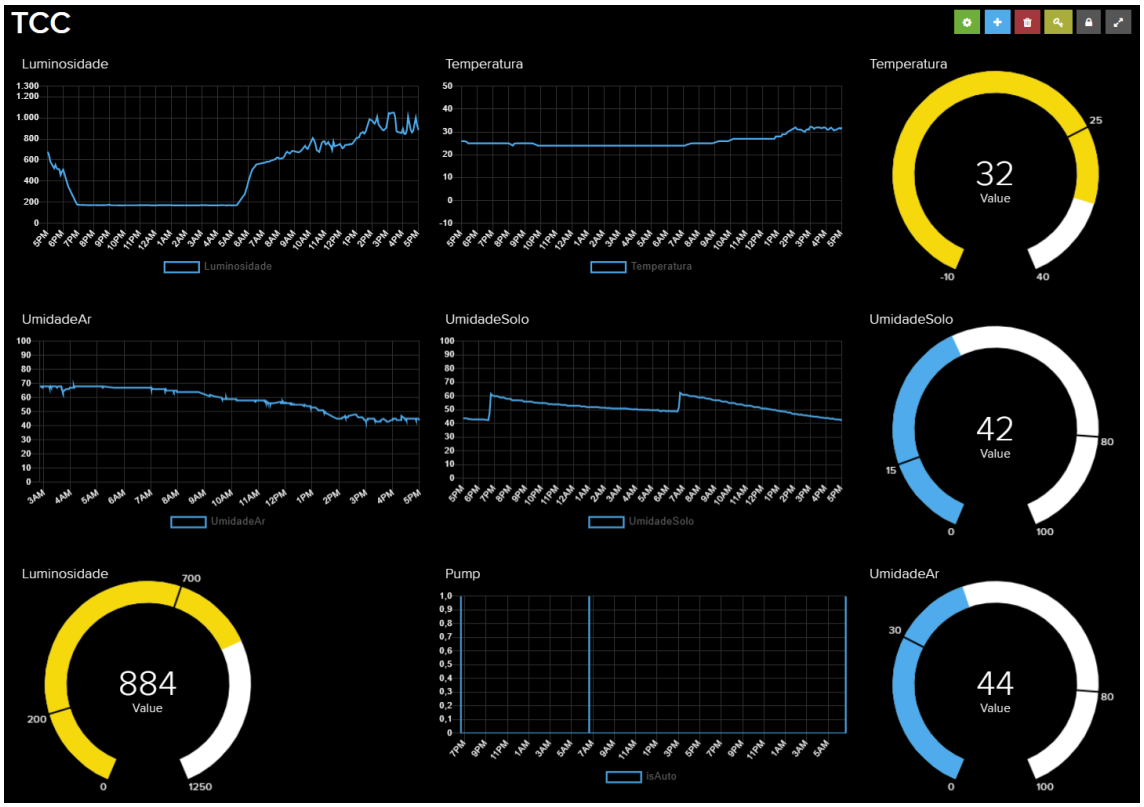


Fonte: Autor.

3.5 *Dashboard*

Para acompanhar os indicadores de temperatura, umidade, luminosidade e os acionamentos da bomba, o próprio *broker* escolhido para receber os dados conta com a funcionalidade de criação de *dashboards*. Contando com uma interface amigável a quem não tem experiência com programação, é possível gerar-se gráficos, *streams* de publicações, botões, entre outros blocos, que permitem uma visualização customizada dos dados.

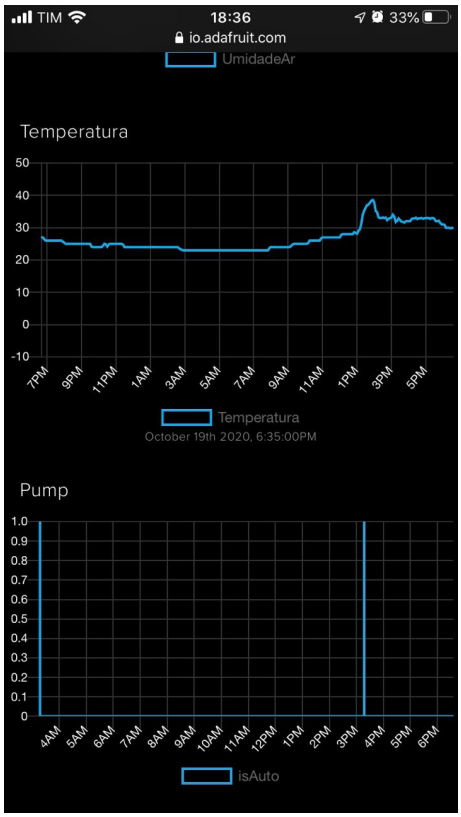
Figura 15: *Dashboard criado para o acompanhamento.*



Fonte: *Dashboard* gerado pelo *broker* Adafruit.io.

Além do visual apresentado na Figura 15, o *broker* ainda conta com versão *mobile*, de forma que o mesmo *dashboard* pode ser acessado do celular do usuário em qualquer lugar do mundo, bastando entrar com suas credenciais. Na Figura 16, pode-se verificar uma parte do *dashboard* visualizado a partir de um celular. No gráfico da bomba (gráfico inferior na Figura 16), cada acionamento representa um pico.

Figura 16: *Dashboard acessado via celular, gráficos de temperatura (acima) e acionamentos da bomba (abaixo).*



Fonte: *Dashboard* gerado pelo *broker* Adafruit.io.

4 RESULTADOS E DISCUSSÕES

Durante duas semanas, o sistema foi deixado em funcionamento como única fonte de irrigação para as plantas testadas (Manjericão e Salsa). Após os primeiros dias, notou-se que as folhas de uma das plantas estavam murchando. A partir disso, uma nova parametrização foi ajustada e seu resultado foi analisado durante os dias que se seguiram, resultando na recuperação das folhas (Figura 17).

Figura 17: *Aspecto do Manjericão saudável.*

Fonte: Autor.

Este ajuste passou a levar em consideração parâmetros mais baixos de temperatura e umidade do solo, que antes necessitavam de condições mais extremas para regulagem. Um exemplo foi o valor da temperatura, que antes passava a atuar para temperaturas superiores a 30^{o}C e agora atua a 28^{o}C.

Ademais, em ventura de uma viagem de lazer do autor, que viria a se ausentar por alguns dias, um teste foi realizado com uma terceira cultura: Tomates Cereja. Com o uso de um conector do tipo T, duas mangueiras foram conectadas à mangueira principal da saída da bomba, objetivando irrigar tanto os Tomates quanto o pote de Manjericão e Salsa simultaneamente.

Deste teste, alguns pontos foram observados:

1. Para irrigar plantas com necessidades de irrigação distintas, alguma medida deve ser tomada para gerar fluxos diferentes para cada planta. Neste caso, uma restrição foi adicionada à saída do pote de Manjericão e Salsa, que passou a receber uma parcela menor de água.

2. O controle de mais de uma planta em potes diferentes é dificultado no caso da irrigação com uma única bomba. Isso porque os parâmetros de acionamento são os mesmos e, caso não haja válvulas de controle em cada saída da mangueira nem sensores de umidade do solo dedicados, não há como controlar de maneira eficiente e totalmente segura os parâmetros de ambas.

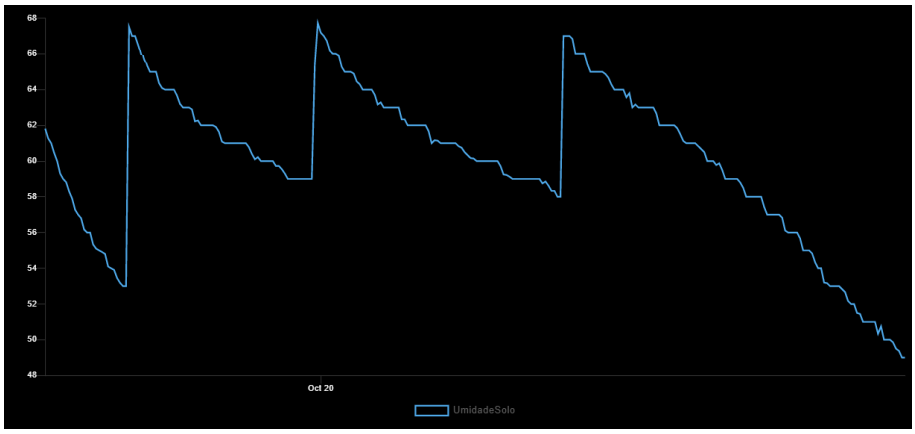
3. A lógica de controle pode ser modificada de maneira a irrigar apenas durante o dia (atualmente as irrigações padrão são de T em T horas mesmo que isso resulte em irrigações noturnas). O reconhecimento de pontos chave do dia para a realização das irrigações pode ser realizado com base em contadores de tempo e luminosidade, contudo, a parametrização empírica dos valores de luminosidade ideais pode variar de acordo com condições climáticas, como em dias nublados, por exemplo. Por isso, optou-se por apenas mencionar este ponto como oportunidade de melhoria do sistema.

Para o desenvolvimento de um produto comercializável a fim de atender o nicho de mercado identificado na pesquisa apresentada na introdução deste trabalho, os pontos acima apresentados devem ser levados em consideração no redesenho da lógica do sistema.

Durante os testes, ainda, alguns problemas relacionados ao *broker* e biblioteca de acesso do NodeMCU foram identificados. Por vezes, o aviso de acionamento da bomba não chegava ao *broker* de forma completa. Este aviso foi programado para ser disparado o valor 1 quando do acionamento da bomba, e o valor 0 quando do desligamento desta, contudo, por vezes apenas o valor 1 era recebido, ou apenas o valor 0. Inicialmente trabalhou-se em mitigar este problema passando por rotinas de checagem de conexão, no qual se percebeu que muitas vezes o problema era oriundo da desconexão do *broker*.

Mesmo com a adição das rotinas supracitadas, ainda houve casos de não recebimento dos valores de aviso de acionamento. Contudo, o problema não foi considerado prejudicial à prova de conceito por ser pouco frequente (ocorria cerca de 10% das vezes) e pelo fato de o gráfico de umidade do solo apresentar um salto de patamar quando do acionamento da bomba, conforme mostra a Figura 18, que representa três acionamentos.

Figura 18: *Evolução da umidade do solo ao longo do tempo.*



Fonte: *Dashboard* gerado pelo *broker* Adafruit.io

Com relação à adequação dos elementos utilizados na construção do projeto, observou-se nos testes iniciais de sensoreamento do solo que o sensor de umidade apresentava oxidação. Ocorre que o sensor nada mais é do que dois eletrodos em meio a sais minerais

e umidade, logo, ocorrem reações de oxidação no material. O resultado deste efeito pode ser observado no eletrodo esquerdo do sensor, apresentado anteriormente na Figura 7. É importante que sejam tomadas precauções quanto a este ponto, porque no caso de um sensor à base de cobre, por exemplo, o precipitado resultante desta reação será nocivo à planta (MCEVOY, 2020).

Para mitigar os efeitos da oxidação do sensor utilizado, foi empregada uma rotina que aciona a alimentação dele apenas quando for realizada uma leitura, desta forma, a reação é dificultada. A solução ideal seria a substituição do sensor por um à base de grafite, uma vez que este é mais estável no meio aplicado e não produz componentes prejudiciais à planta. (MCEVOY, 2020)

5 CONCLUSÃO

Com base nos objetivos traçados, ferramentas e materiais disponíveis e testes realizados, foi possível gerar um sistema customizável de controle de umidade do solo para hortas domésticas. O sistema é também expansível, ou seja, existe a possibilidade de se adicionar mais sensores e mais bombas para aumentar sua capacidade de irrigação.

Embora cada planta exija cuidados diferentes, o sistema não está amarrado a parametrizações fixas. Permeiam como possibilidades para trabalhos futuros, desenvolver a opção de acionamento remoto da bomba, de regular a parametrização também de forma remota e, inclusive, de gerar um banco de dados para os tipos mais comuns de hortaliças domésticas.

Apesar de o foco deste trabalho ter sido hortas domésticas, as possibilidades de aplicação deste tipo de sistema não se limita a esta. A exemplo, a viticultura de precisão - ou plantação de uvas viníferas - tem implantado cada vez mais sistemas de sensoreamento climático (mas também robôs, máquinas de seleção e drones) (GRIZZO, 2020). O resultado são processos produtivos agrícolas cada vez mais controlados e, no caso da viticultura, vinhos mais refinados.

APÊNDICE A - CÓDIGO DO NODEMCU

```
#include <ESP8266WiFi.h>
#include <PubSubClient.h>
#include <DHT.h>


//Parametros constantes
#define PARAMETROLUZFORTE 1000
#define ANALOGINPUT0 A0
#define SOILMOISTURECTRL 14 //D5
#define LIGHTCTRL 2 //D4
#define PUMPCTRL 13 //D7
#define IO_USERNAME  "*******" //omitido por segurança
#define IO_KEY       "*******" //omitido por segurança

//informações da rede WIFI
const char* ssid = "*******";      //SSID da rede WIFI,
                                     omitido por segurança
const char* password =  "*******";    //senha da rede wifi,
                                     omitido por segurança

const char* mqttServer = "io.adafruit.com"; //server
const char* mqttUser = IO_USERNAME;         //user
const char* mqttPassword = IO_KEY;          //password
const int mqttPort = 1883;                  //port

//Tópicos
const char* mqttTopicSub = "richadolphs/feeds/UmidadeSolo";
const char* mqttTopicSubDHT = "richadolphs/feeds/Temperatura";
const char* mqttTopicSubTEMP = "richadolphs/feeds/UmidadeAr";
const char* mqttTopicSubAuto = "richadolphs/feeds/Automatico";
const char* mqttTopicSubLumi = "richadolphs/feeds/Luminosidade";

const int hoursToPump = 12;

WiFiClient espClient;
PubSubClient client(mqttServer, mqttPort, espClient);
```

```
#define DHTPIN 4 //PINO DIGITAL UTILIZADO PELO DHT22
#define DHTTYPE DHT22 //DEFINE O MODELO DO SENSOR (DHT22 / AM2302)
DHT dht(DHTPIN, DHTTYPE); //PASSA OS PARÂMETROS PARA A FUNÇÃO

float leituraUmiSolo = 0, leituraDHTumi = 0,leituraDHTtemp = 0,
leituraLumi = 0, last_Umi = 0;
char str1[8], str2[8], str3[8], str4[8];
unsigned long t = 0, t2 = 0, t_last = 0, t_luz = 0;
bool state = false, toggleIf = true;
bool ligou = true;
int TBOMBA = 3;

//setup de portas e configurações de redes
void setup() {
  Serial.begin(115200);
  pinMode(ANALOGINPUT0, INPUT);
  pinMode(SOILMOISTURECTRL, OUTPUT);
  pinMode(LIGHTCTRL, OUTPUT);
  pinMode(16, OUTPUT);
  pinMode(PUMPCTRL, OUTPUT);

  WiFi.mode(WIFI_AP_STA);
  WiFi.begin(ssid, password);
  Serial.println("Iniciando DHT");
  dht.begin();
  Serial.println("DHT Iniciado");
  Serial.println("Conectando na rede WiFi");
  while (WiFi.status() != WL_CONNECTED) {
    delay(1000);
    Serial.print(".");
  }
  Serial.println("Conectado com sucesso na rede WiFi");

  client.setKeepAlive(10);
  client.setSocketTimeout(10);

  ClientConnect();
  client.publish("richadolphs/feeds/isAuto", "0", true);
  delay(2000);
  digitalWrite(LIGHTCTRL,HIGH);
}



//método pra reconectar ao servido MQTT
void ClientConnect() {
  while (!client.connected()) {
```

```
    delay(2000);
    Serial.println("Conectando ao Broker MQTT...");
    if (client.connect("ESP8266Client", mqttUser, mqttPassword)) {
      Serial.println("Conectado");
      client.subscribe("richadolphs/feeds/Automatico");
    } else {
      Serial.print("Falha estado ");
      Serial.println(client.state());
      delay(5000);
    }
  }
}

//método de leitura dos sensores
void sensorsRead(){
     for (int i = 0; i <= 10; i++)
     {
        leituraLumi = leituraLumi + analogRead(ANALOGINPUT0);
        delay(100);
     }
     digitalWrite(LIGHTCTRL,LOW);
     leituraLumi = leituraLumi/10.0;
     leituraLumi = 1250 - leituraLumi;
     itoa(leituraLumi, str4, 10);
     leituraDHTumi = dht.readHumidity();
     itoa(leituraDHTumi, str2, 10);
     leituraDHTtemp = dht.readTemperature();
     itoa(leituraDHTtemp, str3, 10);
     digitalWrite(SOILMOISTURECTRL,HIGH);
     delay(100);
     for (int i = 0; i <= 10; i++)
     {
        leituraUmiSolo = leituraUmiSolo + analogRead(ANALOGINPUT0);
        delay(10);
     }
     digitalWrite(SOILMOISTURECTRL,LOW);
     leituraUmiSolo = leituraUmiSolo/10.0;
     leituraUmiSolo = map(leituraUmiSolo, 200, 1250, 100, 0);
     itoa(leituraUmiSolo, str1, 10);
     digitalWrite(LIGHTCTRL,HIGH); // estabilizar o sensor
                                          de luminosidade
     if(leituraLumi > PARAMETROLUZFORTE && toggleIf){
      t_luz = millis();
      toggleIf = false;
     }
     else if(leituraLumi < PARAMETROLUZFORTE){
      t_luz = 0;
      toggleIf = false;
```

```
    }
}

// loop principal
void loop() {
  if(millis() - t > 30000)
  {
     sensorsRead();
     if (!client.connected()) { client.connect("ESP8266Client",
                                mqttUser, mqttPassword);  }
     client.publish("richadolphs/feeds/UmidadeSolo", str1, true);
     client.publish("richadolphs/feeds/UmidadeAr", str2, true);
     client.publish("richadolphs/feeds/Temperatura", str3, true);
     client.publish("richadolphs/feeds/Luminosidade", str4, true);
     delay(350);
     Serial.println("Sent Data.");
     t = millis();
  }
  if(millis() - t_last > hoursToPump * 60 * 60 * 1000 ||
                        (leituraUmiSolo < 15 && t_luz > 3600000)
                        || ligou)
                        //Rega se tiverem passado 12h da última
                        irrigação ou se o solo estiver
                        demasiado seco sob luz forte
  {
    if(leituraDHTtemp > 28 && leituraLumi > 1000) TBOMBA += 1;
    if(leituraUmiSolo < last_Umi) TBOMBA += 1;
    ligou = false;
    Serial.println("PUMP THEIR BUMP");
    if (!client.connected()) { client.connect("ESP8266Client",
                                    mqttUser, mqttPassword); }
    client.publish("richadolphs/feeds/isAuto", "1", true);
    digitalWrite(PUMPCTRL, HIGH);
    delay(TBOMBA * 1000);
    if (!client.connected()) { client.connect("ESP8266Client",
                                    mqttUser, mqttPassword); }
    client.publish("richadolphs/feeds/isAuto", "0", true);
    digitalWrite(PUMPCTRL, LOW);
    last_Umi = leituraUmiSolo;
    TBOMBA = 3; //Reset
    t_last = millis();
  }
  delay(150);
}
```

REFERÊNCIAS

GRIZZO, A. *Como funciona a viticultura de precisão?* Disponível em: <https://revistaadega.uol.com.br/artigo/como-funciona-viticultura-de-precisao_12060.html>. Acesso em: 27 out. 2020.

HENRIQUES, R. V. B. *Laboratório Maker - Introdução à Internet das Coisas (IoT)*. Porto Alegre, 2019. p. 94.

LIU, T. *DHT22 Datasheet*. Disponível em: <https://www.sparkfun.com/datasheets/Sensors/Temperature/DHT22.pdf>. Acesso em: 28 set. 2020.

LOS ÁNGELES PROZ, M. DE. *Compostos bioativos em Salsa (Petroselinum crispum) e Manjericão (Ocimum basilicum) Produzidos sob diferentes sistemas de cultivo*. 2020. f. 69. Tese (Mestrado em engenharia) – Programa de Pós-Graduação em Ciência e Tecnologia de Alimentos, Instituto de Ciência e Tecnologia de Alimentos, Universidade Federal do Rio Grande do Sul, Porto Alegre.

MCEVOY, B. *SENSING SOIL MOISTURE: YOU'RE DOING IT WRONG!* Disponível em: <https://hackaday.com/2017/11/16/sensing-soil-moisture-youre-doing-it-wrong/>. Acesso em: 17 out. 2020.

TEIXEIRA, E. A. S. *Arquitetura IoT para aplicação em automação residencial*. 2019. f. 52. Trabalho de Conlusão de Curso – Escola de Engenharia, Universidade Federal do Rio Grande do Sul, Porto Alegre.