UNIVERSIDADE FEDERAL DE SANTA CATARINA
CENTRO TECNOLÓGICO, DE CIÊNCIAS EXATAS E EDUCAÇÃO
DEPARTAMENTO DE ENG. DE CONTROLE, AUTOMAÇÃO E COMPUTAÇÃO
CURSO DE GRADUAÇÃO EM ENGENHARIA DE CONTROLE E AUTOMAÇÃO

Cícero Augusto Weber

ALIMENTADOR AUTOMATIZADO DE BAIXO CUSTO PARA PISCICULTURA

Blumenau
2022

Cícero Augusto Weber

ALIMENTADOR AUTOMATIZADO DE BAIXO CUSTO PARA PISCICULTURA

Trabalho de Conclusão de Curso de Graduação em Engenharia de Controle e Automação do Centro Tecnológico, de Ciências Exatas e Educação da Universidade Federal de Santa Catarina como requisito para a obtenção do título de Engenheiro de Controle e Automação.
Orientador: Prof. Dr. Tiago Davi Curi Busarello

Blumenau
2022

Ficha de identificação da obra elaborada pelo autor,
através do Programa de Geração Automática da Biblioteca Universitária da UFSC.

Weber, Cícero Augusto
ALIMENTADOR AUTOMATIZADO DE BAIXO CUSTO PARA PISCICULTURA / Cícero Augusto Weber ; orientadora, Tiago Davi Curi Busarello, 2022.
84 p.

Trabalho de Conclusão de Curso (graduação) - Universidade Federal de Santa Catarina, Campus Blumenau, Graduação em Engenharia de Controle e Automação, Blumenau, 2022.

Inclui referências.

1. Engenharia de Controle e Automação. 2. Desenvolvimento de um alimentador de baixo custo para piscicultura . I. Busarello, Tiago Davi Curi. II. Universidade Federal de Santa Catarina. Graduação em Engenharia de Controle e Automação. III. Título.

Cícero Augusto Weber

ALIMENTADOR AUTOMATIZADO DE BAIXO CUSTO PARA PISCICULTURA

Este Trabalho de Conclusão de Curso foi julgado adequado para obtenção do Título de "Engenheiro de Controle e Automação" e aprovado em sua forma final pelo Curso de Graduação em Engenharia de Controle e Automação.

Blumenau, 05 de 08 de 2022.

Banca Examinadora:

Documento assinado digitalmente
Tiago Davi Curi Busarello
Data: 05/08/2022 10:50:41-0300
CPF: 319.950.098-65
Verifique as assinaturas em https://v.ufsc.br

Prof. Dr. Tiago Davi Curi Busarello
Universidade Federal de Santa Catarina

Documento assinado digitalmente
Ciro Andre Pitz
Data: 05/08/2022 11:43:58-0300
CPF: 055.728.089-38
Verifique as assinaturas em https://v.ufsc.br

Prof. Dr. Ciro André Pitz
Universidade Federal de Santa Catarina

Documento assinado digitalmente
ALEX FABIANO BUENO
Data: 05/08/2022 12:50:39-0300
CPF: 561.810.780-53
Verifique as assinaturas em https://v.ufsc.br

Prof. Dr. Alex Fabiano Bueno
Universidade Federal de Santa Catarina

Dedico este trabalho a todos aqueles que, de alguma forma, auxiliaram para a concretização desta etapa.

AGRADECIMENTOS

Agradeço primeiramente a minha mãe Maria Susete Tavares, que enquanto pôde fez o máximo para eu receber a melhor educação possível e me tornar a pessoa que sou hoje, a meu pai Jacir Norberto Weber que não mediu esforços e me deu todo o suporte necessário em tudo que precisei, ao meu orientador Tiago Davi Curi Busarello que me ajudou na elaboração deste projeto, a todos os meus amigos e familiares que passaram pela minha vida e fizeram parte dessa trajetória e a todos os professores que me auxiliaram de todas as formas necessárias.

RESUMO

O Brasil é excepcionalmente favorecido em relação às suas fontes hídricas, possuindo a maior quantidade de água doce do mundo, contando com 12% do total existente no mesmo, e sendo inclusive mais abundante do que continentes inteiros como a Europa. Segundo estudos da Universidade Federal do Pará, o aquífero Alter do Chão, que é a maior reserva subterrânea de água doce do mundo, está localizado no Brasil, mais especificamente, sob os estados do Amapá, Pará e Amazonas, em uma área total de 437 mil km², sendo composto por 86 mil km² de água doce. Porém, mesmo com todo este potencial aquático, o Brasil ocupa hoje a 8ª posição no ranking de maiores produtores de peixes de água doce no mundo. Mesmo sendo um país rico em água, a maior parte da população brasileira enfrenta dificuldades financeiras, por esta razão, a inserção da tecnologia na criação de pequenos e médio produtores é baixa e lenta. Este fato provoca um atraso nas produções, que acabam por não atingir seu máximo potencial, consequentemente, não obtêm o lucro que poderiam. Perante esta situação, este trabalho visa o desenvolvimento de um alimentador automatizado de baixo custo, com a utilização de um microcontrolador. O objetivo é propor uma plataforma open source, onde o proprietário pode seguir os passos deste documento e confeccionar seu próprio alimentador, possibilitando aos criadores que não possuem muito capital para investir, uma potencialização nas suas criações. Todo o processo estrutural desenvolvido será descrito neste documento, tanto quanto a montagem passo a passo do circuito, além de todo o código elaborado com todas as linhas comentadas.

Palavras-chave: Alimentador, Piscicultura, Produtores, Microcontrolador, Lucro, Fontes Hídricas.

ABSTRACT

Brazil is exceptionally favored in relation to its water sources, possessing the largest amount of fresh water in the world, accounting for 12% of the total existing in the same, and being even more abundant than continents such as Europe. According to studies by the Federal University of Pará, the Alter do Chão aquifer, which is the largest underground freshwater reserve in the world, is located in Brazil, more specifically, under the states of Amapá, Pará and Amazonas, in a total area of 437 thousand km², comprising 86 thousand km² of fresh water. However, even with all this aquatic potential, Brazil currently occupies the 8th position in the ranking of the largest producers of freshwater fish in the world. Despite being a water-rich country, the most of the Brazilian population faces financial difficulties, for this reason, the insertion of technology in the creation of small and medium producers is low and slow. This fact causes a delay in productions, which end up not reaching their maximum potential, consequently, they do not obtain the profit they could. Given this situation, the purpose of this work is to propose an open source platform, where the owner can follow the steps in this document and make his own feeder, allowing creators who do not have much capital to invest, a potentialization in their creations. The entire structural process developed will be described in this document, as well as the step-by-step assembly of the circuit, in addition to all the code elaborated with all the lines commented.

Keywords: Feeder, Pisciculture, Producers, Microcontroller, Profit, Water Sources.

LISTA DE FIGURAS

LISTA DE TABELAS

SUMÁRIO

1 INTRODUÇÃO

1.1 MOTIVAÇÃO E CONTEXTO

A partir do ano de 1950, com o fim da Segunda Guerra, a população mundial começou a expandir-se rapidamente. Em 1950 o planeta continha cerca de 2,5 bilhões de habitantes, cresceu para 7,5 bilhões em 2017, e provavelmente deve atingir a marca de 11,2 bilhões em 2100 (EBC, EMPRESA BRASIL DE COMUNICAÇÃO, 2021). Esse rápido crescimento populacional provocou uma crescente demanda por recursos, principalmente por alimentos.

Com a busca por novas fontes de proteínas e a constante evolução no desenvolvimento da alimentação do ser humano, a aquicultura vem se tornando uma ótima opção de cultivo, pois é possível implementar a criação em um pequeno espaço quando comparada com os ramos de criação mais comuns no país como a pecuária de corte, a suinocultura e a avicultura.

O Brasil já foi considerado o país com maior potencial para o desenvolvimento da pesca e aquicultura. Hoje ocupa apenas a 13ª posição na produção de peixes em cativeiro, e é o 8º na produção de peixes de água doce (FOGAÇA, 2020). A produção de peixes cultivados no Brasil é a atividade zootécnica que mais vem crescendo no país nos últimos dez anos, movimenta cerca de R$4 bilhões/ano, gera um milhão de empregos diretos e indiretos e cresce a taxas superiores a 10% ao ano (EBC, EMPRESA BRASIL DE COMUNICAÇÃO, 2020).

Uma das atividades mais importantes da piscicultura é a alimentação diária dos peixes, pois além de ser determinante para o resultado do cultivo, o fornecimento de ração tem um grande impacto financeiro, representando, geralmente, cerca de 60% a 70% do custo total da produção dos peixes. Esta prática consome cerca de 900 mil toneladas de rações e movimenta cerca de R$1,2 bilhão/ano (**senar2019**).

Com a alimentação influenciando diretamente e majoritariamente na lucratividade da produção, um tratador automatizado se torna fundamental para maximizar a efetividade da produção. A implementação da automação na indústria é um grande exemplo, pois foi crucial para uma forte expansão operacional e um grande crescimento nos lucros. Isto ocorre pois, grande parte das empresas e indústrias lidam diariamente com processos repetitivos, que realizados de forma automatizada, implicam, na diminuição de tempo na produção e evitam erros operacionais quando comparados com a mão de obra humana. O mesmo se aplica à piscicultura, pois os processos de criação se repetem diariamente e constantemente.

A metrologia também tem papel fundamental e se faz hoje indispensável para diversas fases de um processo de produção, como na medição de nível, temperatura, densidade e umidade de produtos, visando sempre evitar desperdícios de matérias-primas, gastos com manutenções desnecessárias e erros que poderiam ser evitados, procurando

realizar um processo na sua máxima eficiência.

1.2 DELIMITAÇÃO DO PROBLEMA

O constante crescimento da piscicultura no país nos últimos dez anos, demonstra o grande potencial que o Brasil possui para se tornar líder na criação e exportação de peixes. Sendo possível alcançar um crescimento de 5,93% comparando os anos de 2019 e 2020, gerando cerca de 802.930 toneladas produzidas, e um total de U$11,690 milhões referentes a exportação (LIMA FERREIRA *et al.*, 2021), obtendo crescimento até mesmo durante a pandemia de Coronavírus, onde a economia mundial foi afetada e a exportação muito prejudicada pela escassez de voos. A Figura 1 apresenta um gráfico referente à exportação de produtos da área de piscicultura por trimestre nos anos de 2019 e 2020 em milhares de dólares.

Figura 1 – Gráfico da exportação de produtos da área de piscicultura por trimestre.
Fonte: (EBC, EMPRESA BRASIL DE COMUNICAÇÃO, 2020)

Esse forte crescimento nos números, referentes à piscicultura na última década, não foi acompanhado pelo crescimento tecnológico na área. O Brasil é o país do mundo que detém a maior quantidade de água doce em seu território, contando com 12% do total existente no planeta. Possuindo mais que todo o continente europeu que detém 7%, e todo o continente africano que detém 10% (MADEIRO, 2015). Porém, mesmo com toda essa vantagem hídrica, o país se encontra apenas na oitava posição de maior produtor de peixes de água doce, como citado anteriormente.

Atualmente a maior parte dos criadores no Brasil praticam uma criação totalmente manual, sem utilizar aparelhos de instrumentação ou qualquer automação aplicada no processo, forçando assim, uma alimentação de forma variada e desregulada, ocasionando em perdas de rações e consequentemente em um retardo no crescimento dos peixes, gerando um prejuízo desnecessário.

A maioria dos alimentadores automatizados encontrados no mercado nacional, possuem um controle muito básico, geralmente contendo apenas temporizadores no projeto, sem utilizar qualquer tipo de indicação visual ou auditiva perante ao controle de nível da ração ou a outros indicativos. O valor desses alimentadores também é um ponto negativo, com preços variando de R$700 a R$2500, sendo inviável para muitos criadores de pequeno

porte, atrasando assim a evolução da piscicultura no país. Este trabalho visa a criação de um alimentador automatizado de baixo custo, utilizando um controlador básico, que será responsável por todo o controle do projeto.

O projeto descrito no presente documento apresenta o seu desenvolvimento baseado em seis etapas metodológicas fundamentais, sendo elas:

1. A detecção automática do nível da ração através de um sensor ultrassônico;
2. A detecção da temperatura da água através de um termistor NTC.
3. A leitura dos sinais emitidos através de um microcontrolador Atmel AVR.
4. A transmissão de um sinal de saída para dois LEDs.
5. A transmissão de um sinal de saída para um módulo LCD.
6. A rotação de uma mola helicoidal de arame galvanizado realizada por um motor síncrono de 60 Hz.

1.3 OBJETIVOS

Nesta seção são apresentados os objetivos gerais e específicos para a execução do presente Trabalho de Conclusão de Curso.

1.3.1 Objetivo Geral

Desenvolver um alimentador automatizado de baixo custo, que atenda as especificações do cenário atual de criação de peixes em cativeiro, com um custo de construção acessível ao criador e uma melhor eficiência na execução da alimentação.

1.3.2 Objetivos Específicos

Propor uma plataforma open source, onde o proprietário pode seguir os passos deste documento e confeccionar seu próprio alimentador, possibilitando aos criadores que não possuem muito capital para investir, uma potencialização nas suas criações Proporcionar conhecimentos na área da piscicultura com ênfase na alimentação e temperatura. Configurar uma programação eficiente e ajustada às necessidades do ecossistema proposto.

1.4 ESTRUTURA DO DOCUMENTO

O presente documento foi dividido em cinco capítulos de modo a apresentar a base teórica, os materiais e métodos utilizados além dos resultados obtidos. No Capítulo 1, uma breve introdução e caracterização do problema é apresentada. Neste capítulo, também são apresentados os objetivos gerais e específicos do trabalho. No Capítulo 2 é realizada uma breve revisão bibliográfica apresentando as principais abordagens encontradas na literatura sobre os materiais utilizados para o desenvolvimento do alimentador. No capítulo

3 é apresentado todos os itens necessários para a construção da estrutura do alimentador, uma ideia de mecanismo para auxílio na distribuição da ração para diversos ecossistemas, dentre outros itens. No Capítulo 4 são apresentados os principais resultados obtidos como o código desenvolvido para o ecossistema hipotetizado, os valores dos itens utilizados para a construção do alimentador, além de um comparativo com itens semelhantes do mercado atual. Por fim, o Capítulo 5 faz o encerramento do estudo, apresentando as principais conclusões obtidas, uma breve análise do cenário nacional e as perspectivas futuras para o país.

2 REVISÃO DE LITERATURA

2.1 ARDUINO

Em termos práticos, um Arduino é um pequeno computador que você pode programar para processar entradas e saídas entre o dispositivo e os componentes externos conectados a ele, como demonstrado na Figura 2. O Arduino é o que chamamos de plataforma de computação física ou embarcada, ou seja, um sistema que pode interagir com seu ambiente por meio de hardware e software. Por exemplo, um uso simples de um Arduino seria para acender uma luz por um determinado intervalo de tempo após o acionamento do botão. Nesse exemplo, o Arduino teria uma lâmpada e um botão conectados a ele. O Arduino aguardaria até que o botão fosse pressionado; uma vez pressionado, ele acenderia a lâmpada e iniciaria a contagem. Depois de contados 30 segundos, apagaria a lâmpada e aguardaria um novo apertar do botão. Poderia-se utilizar essa configuração para controlar uma lâmpada em um closet, por exemplo. Esse conceito também poderia ser estendido pela conexão de um sensor, como um sensor de movimento PIR, para acender a lâmpada quando ele fosse disparado. Esses são alguns exemplos simples de como você poderia utilizar um Arduino (MCROBERTS, 2018).

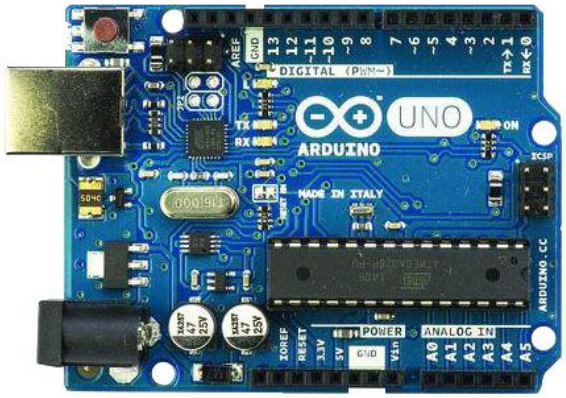

Figura 2 – Ilustração de um microcontrolador Arduino Uno.
Fonte: (NEWARK, 2021)

A maior vantagem do Arduino sobre outras plataformas de desenvolvimento de microcontroladores é a facilidade de sua utilização; pessoas que não são da área técnica podem, rapidamente, aprender o básico e criar seus próprios projetos em um intervalo de tempo relativamente curto. Há uma grande comunidade de pessoas utilizando Arduinos, compartilhando seus códigos e diagramas de circuito para que outros os copiem e modifiquem. A maioria dessa comunidade também está muito disposta a auxiliar outros desenvolvedores (MCROBERTS, 2018).

O Arduino se popularizou pela sua tecnologia versátil e sua fácil utilização, devido a sua plataforma eletrônica de código aberto baseada em hardware e software de simples

utilização, e de seu baixo custo, quando comparado com outros controladores (MOREIRA *et al.*, 2018).

O hardware e o software do Arduino são ambos de fonte aberta, o que significa que o código, os esquemas, o projeto etc. podem ser utilizados livremente por qualquer pessoa e com qualquer propósito. Dessa forma, há muitas placas-clone e outras placas com base no Arduino disponíveis para compra, ou que podem ser criadas a partir de um diagrama (MCROBERTS, 2018). A Figura 3 ilustra uma placa de circuito impresso.

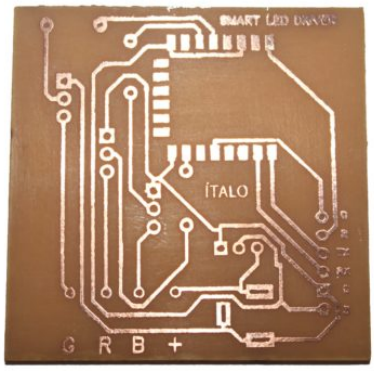

Figura 3 – Ilustração de uma placa de circuito impresso.
Fonte: (**Placa**)

Pode-se estender o Arduíno com a utilização de shields (escudos), que são placas de circuito contendo outros dispositivos (por exemplo, receptores GPS, displays de LCD, módulos de Ethernet etc), que você pode simplesmente conectar ao seu Arduino para obter funcionalidades adicionais (MCROBERTS, 2018).

O microcontrolador pode ser conectado a LEDs, displays (mostradores) de matriz de pontos, botões, interruptores, motores, sensores de temperatura, sensores de pressão, sensores de distância, receptores GPS, módulos Ethernet ou qualquer outro dispositivo que emita dados ou possa ser controlado (MCROBERTS, 2018).

O Arduino pode ser utilizado para desenvolver objetos interativos independentes, ou pode ser conectado a um computador, a uma rede, ou até mesmo à Internet para recuperar e enviar dados do Arduino e atuar sobre eles. Em outras palavras, ele pode enviar um conjunto de dados recebidos de alguns sensores para um site, dados estes que poderão, assim, ser exibidos na forma de um gráfico (MCROBERTS, 2018).

Pode-se ler dados de um sensor e controlar, componentes como luzes, motores, termostatos e portas de garagem. Ele foi desenvolvido principalmente para fins de prototipagem (JAVED, 2017).

A placa Arduino é composta de um microprocessador Atmel AVR, um cristal ou oscilador (um relógio simples que envia pulsos de tempo em uma frequência especificada para permitir sua operação na velocidade correta) e um regulador de voltagem de 5 volts

(alguns Arduinos podem usar um regulador de voltagem chaveado e outros, como o Due, não são compatíveis com 5 volts) (MCROBERTS, 2018).

Para programar o Arduino (fazer com que ele faça o que você deseja) você utiliza o IDE do Arduino, um software livre no qual você escreve o código na linguagem que o Arduino compreende (baseada na linguagem C). O IDE permite que você escreva um programa de computador, que é um conjunto de instruções passo a passo, das quais você faz o upload para o Arduino. Seu Arduino, então, executará essas instruções, interagindo com o que estiver conectado a ele. No mundo do Arduino, programas são conhecidos como sketches (rascunho, ou esboço) (MCROBERTS, 2018).

2.2 LEDS

O LED é um componente que conta com dois terminais, nomeados anodo e catodo, sendo considerado assim, um componente bipolar. Dependendo da forma da polarização, permite ou não a passagem da corrente elétrica, o que acarreta consequentemente, na geração ou não de luz (RANGEL; SILVA, P. B.; GUEDE, 2009).

A energia gerada em um LED é dissipada como luz e calor. A luz é emitida a partir do chip semicondutor e irradiada em todas as direções, porém, não irradiam calor como em uma lâmpada convencional. O calor gerado é retido no interior do LED e deve ser eliminado através do dissipador de calor evitando falhas no mesmo. Os LED's não emitem radiação IV (Infravermelho) ou UV (Ultravioleta) na luz visível (RANGEL; SILVA, P. B.; GUEDE, 2009).

Atualmente, existem LEDs que chegam a atingir a marca de 120 lumens de fluxo luminoso, disponíveis em diferentes cores, como demonstrado na Figura 4, responsáveis pelo aumento considerável na substituição de alguns tipos de lâmpadas em várias aplicações luminosas (OLIVEIRA; SANTOS, 2020).

Figura 4 – Ilustração com diferentes opções de cores de LEDs.
Fonte: (ELETRONIK REICHELT, 2019)

2.3 SENSOR DE DISTÂNCIA ULTRASSÔNICO

O HC-SR04 é um dispositivo ultrassônico para medição de distância, bastante utilizado em equipamentos eletromecânicos. Nele, há um circuito de controle, um transmissor

e um receptor ultrassônico, como demonstrado na Figura 5. O sensor fornece medidas de 20 mm a 4000 mm, cuja precisão pode chegar a 3mm (ELEC FREAKS, 2018).

Figura 5 – Ilustração do sensor de distância HC-SR04.
Fonte: (ELEC FREAKS, 2018)

O funcionamento do HC-SR04 se baseia no envio de sinais ultrassônicos do sensor, que aguarda o retorno do sinal nomeado de Echo, e com base no tempo entre envio e retorno, consegue calcular a distância entre o sensor e o objeto detectado. Primeiramente é enviado um pulso de 10µs, indicando o início da transmissão de dados. Depois disso, são enviado 8 pulsos de 40 KHz e o sensor então aguarda o retorno (em nível alto/high) (ADILSON THOMSEN, 2011). A Figura 6 ilustra esse comportamento.

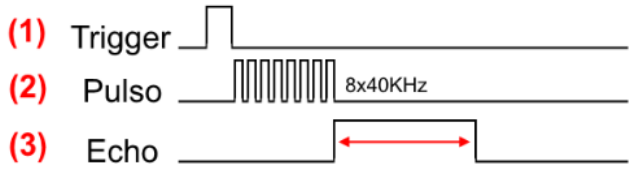

Figura 6 – Ilustração do cálculo de tempo do envio do sinal.
Fonte: (ADILSON THOMSEN, 2011)

Para determinar a distância entre o sensor e o objeto, utiliza-se a equação:

$$Distancia = (TempoEchoNivelAlto * VelocidadeSom)/2 \quad (1)$$

A Figura 7 ilustra esse procedimento (ADILSON THOMSEN, 2011).

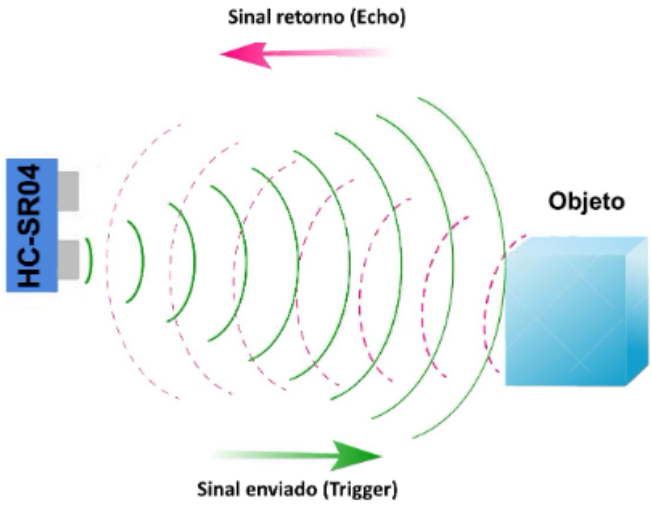

Figura 7 – Ilustração de envio do sinal ultrassônico.
Fonte: (ADILSON THOMSEN, 2011)

2.4 SENSOR DE TEMPERATURA

Termistor, termo derivado da combinação das palavras termo (temperatura) e resistor. Essa denominação passou a ser atribuída à classe de componentes eletrônicos semicondutores, cuja resistência elétrica é fortemente influenciada pela variação da temperatura, que altera a concentração de portadores de carga no dispositivo. Tal como as termo resistências (RTD), o termistor é também um resistor sensível à temperatura, que prima por sua altíssima sensibilidade, e dessa forma, traduz pequenas variações de temperatura em grandes variações na resistência elétrica do dispositivo, seguindo uma relação é exponencial. Em termometria, é consenso que os termopares se destacam pela simplicidade e pela versatilidade; as RTDs, pela estabilidade; os termistores, pela sensibilidade. Nos termistores, o ganho de sensibilidade tem como contraponto uma forte não linearidade na curva de resposta (CARVALHO FILHO *et al.*, 2021). Termistores são dispositivos aplicados numa faixa de temperatura que começa em valores criogênicos, da ordem de -260C, e podem ser utilizados até 300C. Entretanto, dificuldades construtivas, decorrentes do próprio princípio operacional, comercialmente limita a maioria das aplicações à faixa de -60C e 160C (ANALÓGICA INSTRUMENTAÇÃO E CONTROLE, 2013).

Os sensores de temperatura empregados neste projeto são os termistores NTC 3950, como demonstrado na Figura 8. Há várias vantagens e desvantagens em utilizar um termistor NTC. As vantagens dos termistores incluem seu pequeno tamanho e alto grau de estabilidade. Os NTCs também são duradouros e muito precisos. As desvantagens incluem sua não linearidade e sua inadequação para uso em temperaturas extremas (CRISTIANO BERTULUCCI SILVEIRA, 2018).

Figura 8 – Ilustração de um sensor de temperatura NTC 3950.
Fonte: (USINAINFO, 2019)

A resistência elétrica dos termistores pode variar tanto de forma proporcional ou inversamente proporcional com o aumento de temperatura ao qual o sensor for exposto. Por essa característica é feita uma classificação dos termistores, sendo NTC(negative temperature coeficiente) e PTC(positive temperature coeficiente) (CRUZ; CORREA; SILVA, W. L., s.d.), a Figura 9 demonstra graficamente essa diferença.

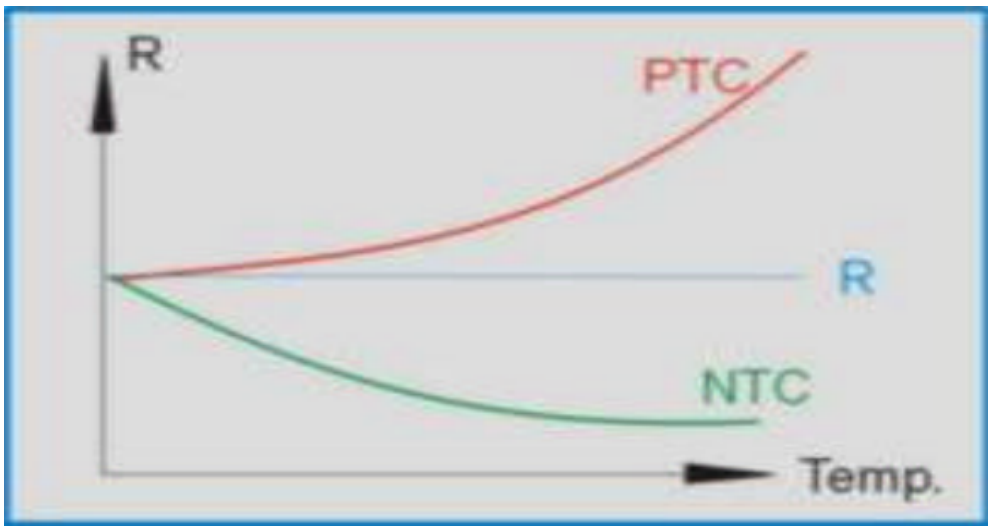

Figura 9 – Curvas sensores PTC e NTC.
Fonte: (FREITAS, L. M. d. *et al.*, 2019)

O NTC é mais utilizado do que o PTC, devido a maior facilidade de ser manufaturado. O PTC tem como sua peculiaridade possuir um ponto de transição, somente a partir de uma determinada temperatura exibirá uma variação ôhmica com a variação da temperatura (FREITAS, L. M. d. *et al.*, 2019). Os sensores NTC são mais utilizados na coleta dos dados da temperatura, já os sensores PTC são normalmente utilizados como dispositivos de proteção.

2.5 MOTOR ELÉTRICO

O motor é um equipamento que transforma energia elétrica em energia mecânica. Possibilitando assim movimentar máquinas e transmitir rotação para outros componentes. Como eixos, polias e engrenagens. Dessa forma, é o principal componente de muitos equipamentos industriais (VENTURI, 2019).

São muitos os tipos de motores elétricos que você encontra no mercado, mas, pode-se separá-los em 3 grandes grupos: Motores de corrente alternada (CA), Motores de corrente contínua (CC) e Motores Universais (CA e CC). A corrente elétrica desses motores diz respeito ao fluxo de elétrons que neles ocorrem e o sentido deste fluxo é o que determina se a corrente é alternada ou contínua. Nos motores de corrente alternada, um ímã é colocado para girar a partir de um campo magnético girante, produzido por bobinas percorridas por correntes elétricas alternadas. Na corrente contínua (CC), o fluxo de elétrons percorre um único sentido, do polo negativo (que exerce repulsão) para o polo positivo (que exerce atração)(FREITAS, L. B., s.d.). A Figura 10 apresenta os principais tipos de motores utilizados nas indústrias atualmente.

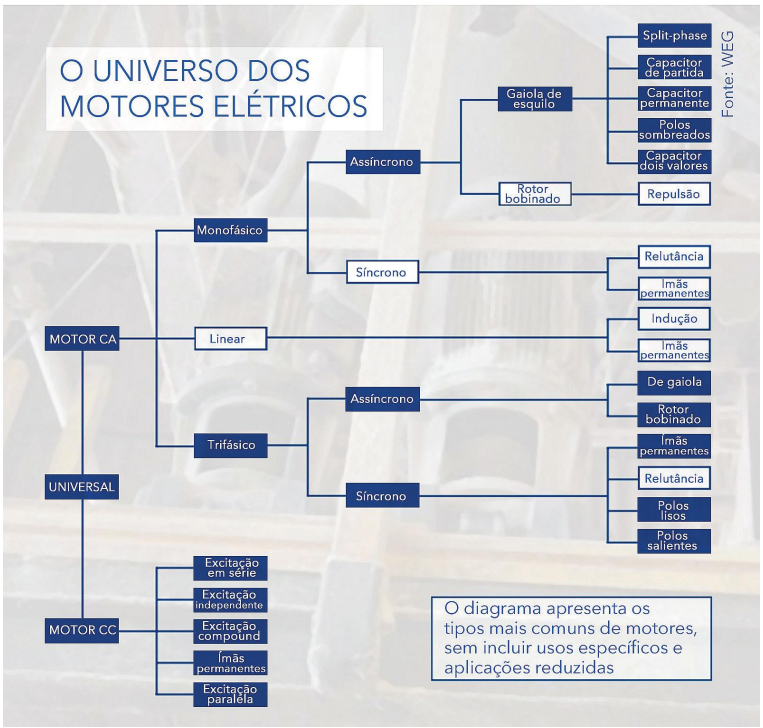

Figura 10 – Subdivisões de Motores.
Fonte: (PETRUZELLA, 2013)

O motor elétrico utilizado nesse trabalho é um motor Motor Síncrono Monofásico de ímã permanente de corrente alternada, demonstrado na Figura 11, muito utilizado em fornos micro-ondas para a rotação do prato. Este motor possui: tensão de funcionamento de 220V - 240V, potência 4W, rotação 5/6 RPM, frequência 50/60Hz e material do eixo composto por plástico (LOMBAS *et al.*, 2020). É um motor de baixo custo que apresenta

uma ótima resposta para a tarefa proposta.

Figura 11 – Motor Síncrono Monofásico.
Fonte: (LOMBAS *et al.*, 2020)

2.6 RELÉ

O relé, é um interruptor eletromecânico que foi projetado por Michael Faraday na década de 1830, com inúmeras aplicações possíveis em comutação de contatos elétricos, servindo para ligar ou desligar dispositivos (WERNEC; LAMEU, 2018).

Existem dois tipos principais de relé, são eles: eletromecânico e de estado sólido. O relé de estado sólido é utilizado no chaveamento de cargas indutivas e resistivas, já o relé eletromecânico contém uma parte mecânica de contato e seu acionamento ocorre através da corrente elétrica em uma bobina (AMÉRICO, s.d.).

Quando uma corrente circula pela bobina, ela cria um campo magnético que atrai um ou uma série de contatos fechando ou abrindo circuitos. Ao interromper essa corrente o campo magnético também será interrompido, fazendo com que os contatos voltem para a posição original. Os relés podem ter algumas configurações referentes aos seus contatos: eles podem ser NA (normalmente aberto), NF (normalmente fechado) ou ambos (WANZELER, 2015). Como demonstrado na Figura 12.

Devem ser observadas as limitações dos relés quanto a corrente e tensão máxima admitida entre os terminais. Se não forem observados estes fatores a vida útil do relé estará comprometida, ou até mesmo a do circuito controlado (JESUS, 2018).

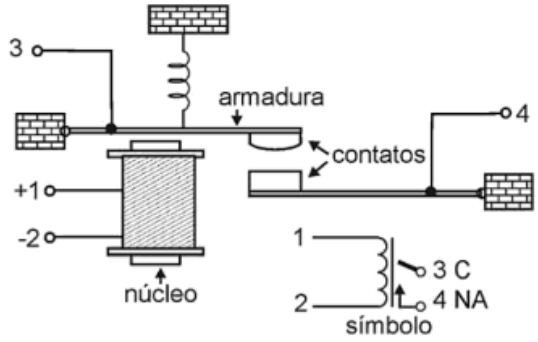

Figura 12 – Ilustração dos componentes de um relé.
Fonte: (BRAGA, 2017)

2.7 MÓDULO LCD

Os módulos LCD são interfaces de saída muito úteis em sistemas microprocessados. Estes módulos podem ser gráficos e a carácter. Os módulos LCD gráficos são encontrados com resoluções de 122x32, 128x64, 240x64 e 240x128 DPI, e geralmente estão disponíveis com 20 pinos para conexão. Os LCD comuns são especificados em número de linhas por colunas e são encontrados nas configurações previstas na Tabela 1 (BARBACENA; FLEURY, 1996).

Tabela 1. Módulos LCD disponíveis

Número de Colunas	**Número de Linhas**	**Quantidade de pinos**
8	2	14
12	2	14/15
16	1	14/16
16	2	14/16
16	4	14/16
20	1	14/16
20	2	14/16
20	4	14/16
24	2	14/16
24	4	14/16
40	2	16
40	4	16

Os módulos LCD são projetados para conectar-se com a maioria das CPU's disponíveis no mercado, bastando para isso que esta CPU atenda as temporizações de leitura e escrita de instruções e dados, fornecido pelo fabricante do módulo (BARBACENA; FLEURY, 1996).

Para utilizar os displays de cristal líquido é necessário a utilização de um controlador específico. Um dos controladores mais comuns no mercado é o Hitachi HD44780, presente em diversos modelos de LCD de vários tamanhos diferentes. Cada elemento (célula) pode

ser acessado e alterado via programação, utilizando funções específicas (TANNUS, 2018). A Figura 13 ilustra este componente.

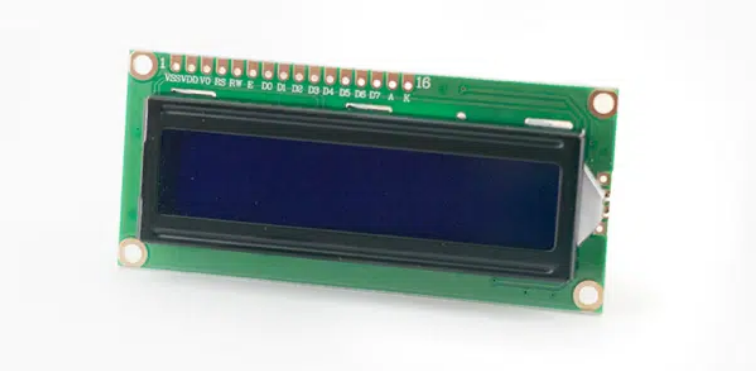

Figura 13 – Módulo LCD.
Fonte: (TANNUS, 2018)

2.8 POTENCIÔMETRO

O potenciômetro pode ser mais claramente definido como sendo um divisor de tensão variável. Ele é composto por uma faixa de material resistivo (geralmente grafite) ligada entre seus dois terminais externos. Nesse material, desliza um cursor, ligado diretamente ao terminal central do potenciômetro. Esse cursor pode ser movimentado através de um eixo rotativo ou um pino de plástico ou metal. Quando altera-se a posição do cursor, altera-se a resistência entre o terminal central e os dois terminais externos do potenciômetro (PATSKO, 2006).

O potenciômetro pode ser utilizado em aplicações que envolvam deslocamentos, movimentos e outros fenômenos puramente mecânicos. Através desse componente é possível que a mudança de uma variável mecânica, como um ângulo ou uma altura, seja transformada numa mudança de uma característica elétrica (PATSKO, 2006).

O potenciômetro também possui a capacidade de utilização na variação de resistência e pode ser definido como um tipo especial de resistor, sendo uma resistência elétrica variável. O Potenciômetro também serve para ajustar os valores de tensão e corrente de um circuito e dessa forma controlar o valor entrada, amplificação ou atenuação (COSTA GOMES *et al.*, s.d.). A Figura 14 ilustra o interior de um potenciômetro.

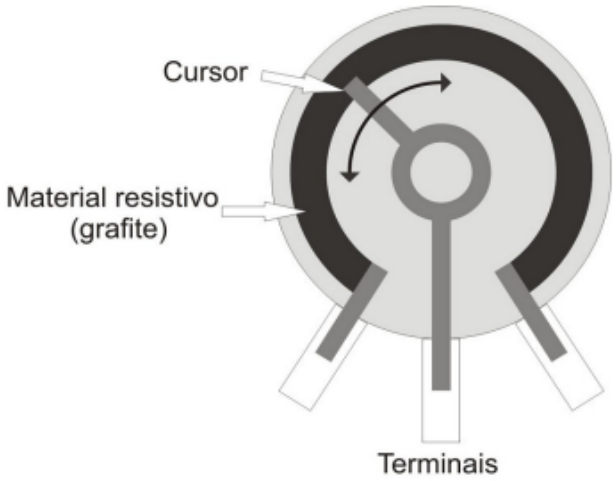

Figura 14 – Interior do potenciômetro com suas partes principais destacadas.
Fonte: (PATSKO, 2006)

3 SISTEMA PROPOSTO

A realização deste trabalho de conclusão de curso foi dividida em cinco etapas. Inicialmente, uma introdução à alimentação da piscicultura é apresentada, bem como a motivação e os objetivos deste trabalho. Em seguida, é feita uma revisão da literatura a fim de descrever todos os dispositivos essenciais utilizados para a construção do alimentador automatizado. Apresentados esses conceitos, parte-se para o projeto do alimentador.

O projeto do alimentador automatizado de baixo custo para piscicultura descreve detalhadamente o cálculo da alimentação e a influência que a temperatura possui para o ecossistema. Além disso, no projeto é apresentada a estrutura física completa do alimentador, a programação desenvolvida, dentre outros aspectos.

Após exemplificada a etapa do projeto do alimentador, são apresentados os resultados obtidos para a validação do projeto. Tais como a amostragem da alimentação, os valores totais do projeto, o código de programação ajustada a este ecossistema, dentre outros.

3.1 PROJETO DO ALIMENTADOR

Para projetar o alimentador, precisa-se primeiramente entender o cálculo da quantidade de ração necessária e se há algum aspecto natural que exerça influência no ecossistema, necessitando assim de um controle específico. Posteriormente precisa-se definir como será realizada a estrutura do alimentador; definir um tanque para armazenamento da ração; Projetar a estrutura de saída da ração; definir como será realizado o movimento da ração até o destino desejado; onde posicionar o microcontrolador e como proteger sua estrutura.

3.1.1 Cálculo da Alimentação

A quantidade de ração necessária para alimentar os peixes é calculada com base na taxa de arraçoamento da criação. Estudos mostraram que essa taxa varia de 2% a 5% da biomassa total do ecossistema por dia. Para alevinos e juvenis, pode-se considerar de 4% a 5% do peso vivo por dia. Para adultos pode-se considerar 2% a 3% da biomassa total por dia (ZIMMERMANN; FITZSIMMONS, 2004). Por exemplo, em um ecossistema com 100 tilápias de 0.5 kg cada, têm-se 50 kg de biomassa (peso vivo). Com taxa de arraçoamento de 3%, esses peixes precisarão comer 1500 gramas de ração por dia. A quantidade de ração fornecida por dia (QTD) é dada plea equação (3.1).

$$QTD = (PV) * 3\% \tag{2}$$

então:

$$50kg * 3\% = 1,5kg \tag{3}$$

Devido aos peixes não possuírem a capacidade de regular a temperatura do próprio corpo, quando a água esfria, seu metabolismo é reduzido, e quando a temperatura está alta demais, os animais entram em estresse, levando assim a redução de apetite. Para os peixes tropicais, é considerado baixas as temperaturas quando a água atinge de 14 a 18 °C, e altas temperaturas quando ultrapassa os 32 °C, como demonstrado na Figura 15 (FABÍOLA DE LUCA COIMBRA BOMTEPO, MARCELO DE SOUSA NUNES, VALÉRIA GEDANKEN, WENDERSON ARAÚJO, TONY OLIVEIRA, PLÍNIO QUARTIM, 2019). Portanto, sempre que a temperatura da água estiver muito alta ou muito baixa faz-se necessário a redução do fornecimento de ração, para evitar desperdícios.

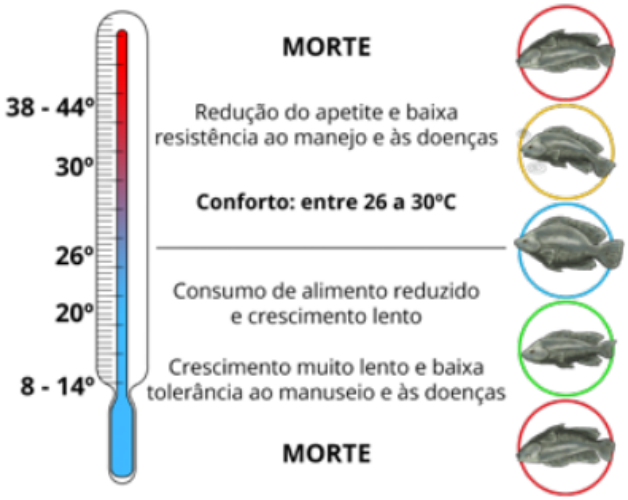

Figura 15 – Indicativo da influência da temperatura no ecossistema dos peixes.
Fonte: (FABÍOLA DE LUCA COIMBRA BOMTEPO, MARCELO DE SOUSA NUNES, VALÉRIA GEDANKEN, WENDERSON ARAÚJO, TONY OLIVEIRA, PLÍNIO QUARTIM, 2019)

3.1.2 Desenvolvimento da Estrutura do Alimentador

A estrutura do alimentador foi desenvolvida visando o maior custo/benefício possível, buscando a economia máxima do produtor, com uma eficiência satisfatória na execução da tarefa.

3.1.2.1 Tanque de Armazenamento

O tanque escolhido para o armazenamento da ração foi um balde plástico alimentício, tendo como exemplo uma bombona de 50 L, pois possui uma boa vedação, um bom espaço interno e um custo acessível. A Figura 16 apresenta diferentes opções de tanques de armazenamento com suas respectivas medidas, sendo essas necessárias para a programação do Arduino.

Figura 16 – Possíveis recipientes para o armazenamento da ração.
Fonte: (SILVA, O. H. da *et al.*, 2017)

3.1.2.2 Estrutura de Saída da Ração (Servidor)

A estrutura de saída da ração proposta, baseia-se em canos de PVC, redutor PVC e um conector T PVC, itens que possuem um custo bastante razoável e uma boa durabilidade perante chuvas e tempo úmido. O cano inferior possui duas aberturas de saída para o despejo da ração, uma em cada lado, além de possuir um mola helicoidal na parte interna acoplada no motor. Na parte superior há a conexão com o recipiente de armazenagem. A Figura 17 indica o funcionamento deste mecanismo.

3.1.2.3 Movimentação da Ração

A ração chegará a estrutura de saída através da gravidade e será movimentada em direção às aberturas pela mola helicoidal. Uma das extremidades da mola é fixada no motor que está acoplado na tampa de PVC. Conforme a rotação do motor acontece, a ração se movimenta até alcançar alguma das saídas e ser despejada no ecossistema. A Figura 18 ilustra esse mecanismo.

3.1.2.4 Posicionamento e Proteção do Arduino

O microcontrolador é posicionado na parte lateral do recipiente de armazenamento da ração e é revestido por uma case de acrílico como proteção primária, demonstrada na Figura 19.

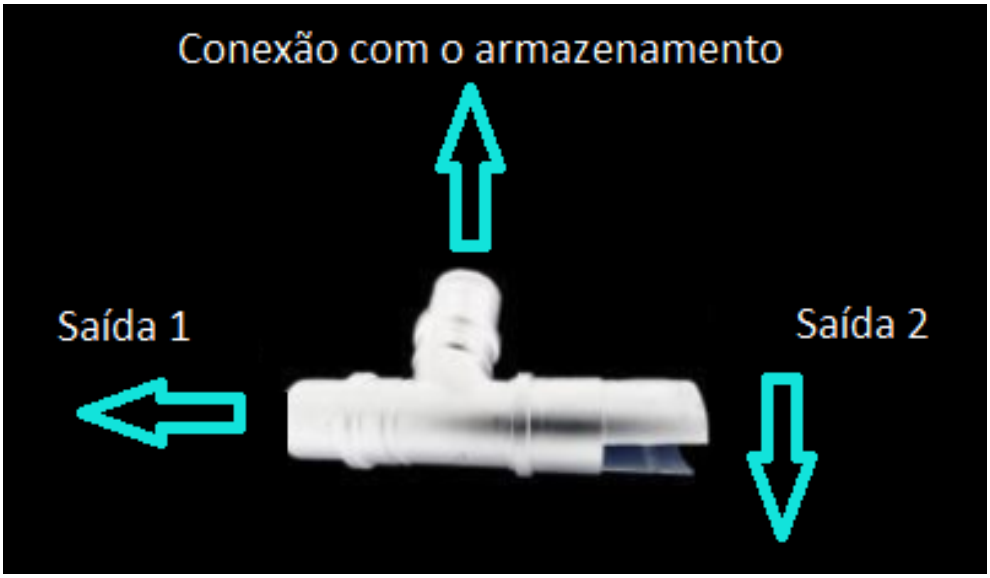

Figura 17 – Estrutura de saída da ração (servidor).
Fonte: Autor

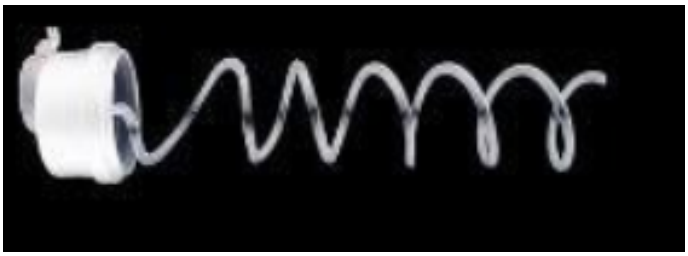

Figura 18 – Mola helicoidal produzida com arame galvanizado.
Fonte: Autor

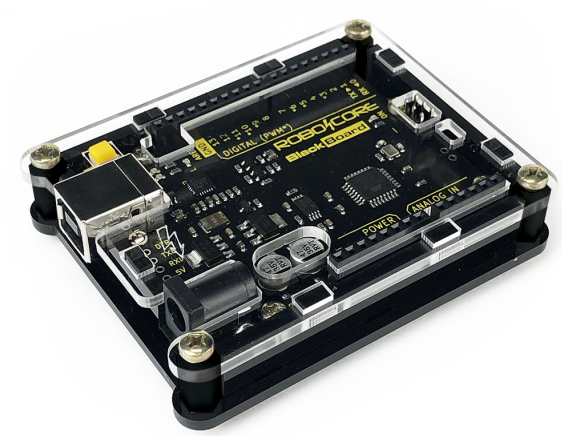

Figura 19 – Case protetora para Arduino.
Fonte: (ROBOCORE, 2022)

O Arduino Uno possui dimensões de 53x68mm como demonstrado na Figura 20. Já a case protetora para o microcontrolador, possui dimensões de 82x65x22mm (EVANS; NOBLE; HOCHENBAUM, 2013).

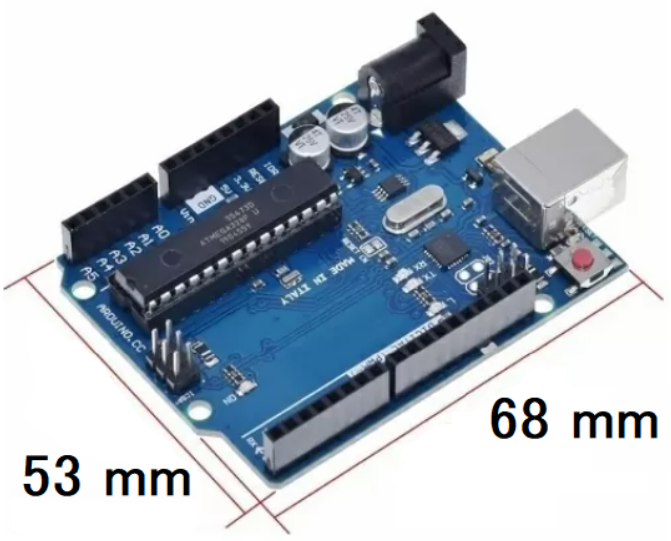

Figura 20 – Dimensões do Arduino Uno.
Fonte: (ROBOCORE, 2022)

Por ser tratar de um protótipo utiliza-se uma protoboard 400 pinos, esta placa possui dimensões 85x55mm como demonstrado na Figura 21. Porém, uma placa de circuito impressa seria a melhor opção.

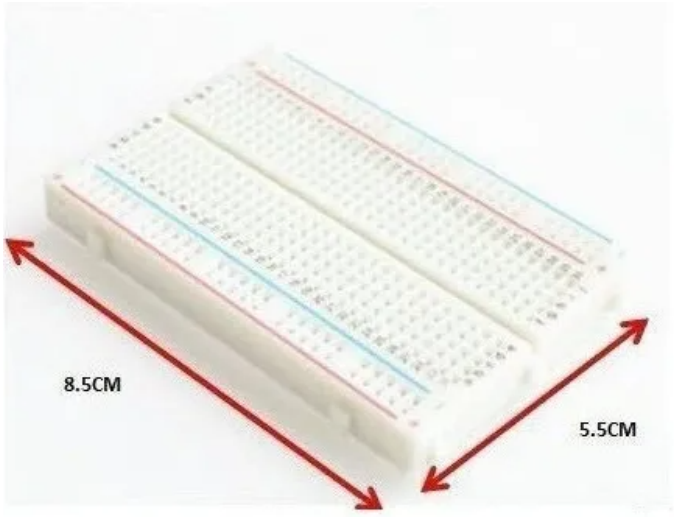

Figura 21 – Dimensões protoboard 400 pinos.
Fonte: (MEKANUS ROBÓTICA EDUCACIONAL, 2016)

Uma segunda caixa de dimensões 280x180x140mm é utilizada como proteção final do Arduino e da protoboard, este equipamento é demonstrado na Figura 22. Constituída de plástico, possui uma boa proteção perante tempo úmido, chuvas e sol.

Figura 22 – Caixa utilizada para proteção do Arduino Uno.
Fonte: (ALFABOT, 2022)

3.1.3 Posicionamento do Sensor de Distância Ultrassônico com Referencial para os LEDs

O sensor de distância ultrassônico será posicionado na tampa do recipiente de armazenamento, pois desta forma só há a necessidade da utilização de um sensor, caso fosse colocado nas laterais seria necessário a utilização de mais sensores, encarecendo assim o custo do projeto. A Figura 23 ilustra esta aplicação do sensor e demonstra como serão programadas as distâncias para o acionamento dos LEDs. De uma distância de zero a 30 cm o nível de ração é considerado cheio. De 30 a 45 cm, o nível de ração é considerado médio e o LED amarelo precisa ser acionado. Com uma distância maior que 45 cm, o nível da ração é considerado baixo e o LED vermelho precisa ser acionado.

3.2 AMPLIAÇÃO DO CONTROLE

O mecanismo elaborado utiliza basicamente sete periféricos para o Arduino, sendo um sensor de distância ultrassônico, dois LEDs, um sensor de temperatura, um relé, um potenciômetro e uma módulo LCD. O controle do mecanismo para uma criação convencional é tranquilamente efetuado, porém, caso o proprietário esteja investindo em uma criação mais sensível na qual necessite de um controle ainda maior há a possibilidade da utilização de mais componentes.

Primeiramente é possível acrescentar um maior número de LEDs caso o proprietário se sinta mais confortável com uma maior sinalização visual, podendo ser utilizado para mais marcações do nível de controle da ração e também para uma melhor visualização.

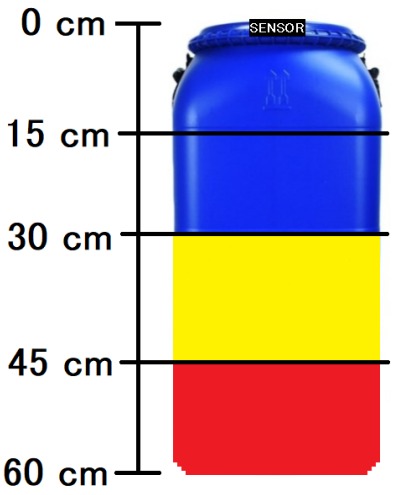

Figura 23 – Ilustração de posicionamento do sensor ultrassônico e referencial dos LEDs.
Fonte: Autor

Outra opção é a utilização de um maior número de sensores de distância ultrassônicos, possibilitando uma exatidão no nível de ração, sendo possível a transmissão praticamente em tempo real.

Também há a possibilidade da inserção de um buzzer caso o proprietário queira incluir uma sinalização sonora. O buzzer é um componente eletrônico que converte um sinal elétrico em onda sonora. Este dispositivo é utilizado em computadores, despertadores, carros, entre outros. O buzzer é composto por duas camadas de metal, uma terceira camada de cristal piezoelétrico, envolvidas em um invólucro de plástico, e dois terminais para ligação elétrica (PINTO, 2016). Este componente é demonstrado na Figura 24.

Figura 24 – Buzzer.
Fonte: (PINTO, 2016)

3.3 DISTRIBUIÇÃO DA RAÇÃO

O sistema de distribuição da ração foi desenvolvido para englobar todo e qualquer tipo de criação, sendo ela grande ou pequena, comercial ou recreativa. Possui três fortes características: fácil instalação, baixo custo e uma grande abrangência na adaptação aos mais variados ecossistemas de criação.

O mecanismo é desenvolvido com a interligação de tubos PVC em um compartimento onde a ração é despejada pelo alimentador, sendo uma caixa ou funil, de preferência plásticos pela questão de valores e peso. Aplicando uma queda nos tubos PVC, consegue-se deslocar a ração até o ecossistema pela força da gravidade. Dependendo do tamanho da criação é necessário a utilização de um maior número de tubos, assim como a utilização de uma queda maior. Também se faz necessária a utilização de escoras de sustentação, como tanto a estrutura de tubos PVC quanto o alimento transportado dentro deles são muito leves, não há a necessidade de uma grande estrutura de sustentação, sendo possível a utilização de ferro, madeira ou concreto. Um esboço do mecanismo é demonstrado em um esquemático 3D nas Figuras 25, 26 e 27.

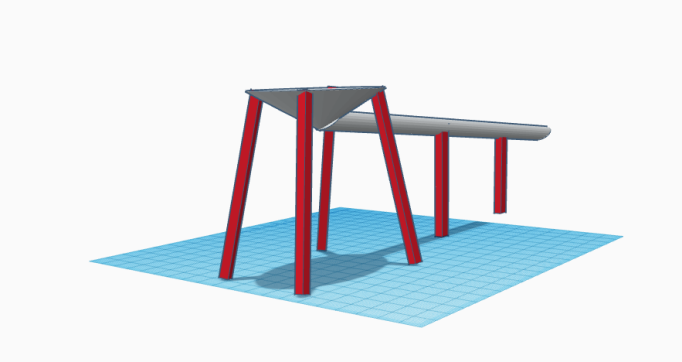

Figura 25 – Esquemático 3D do mecanismo de distribuição da ração lado 1.
Fonte: Autor

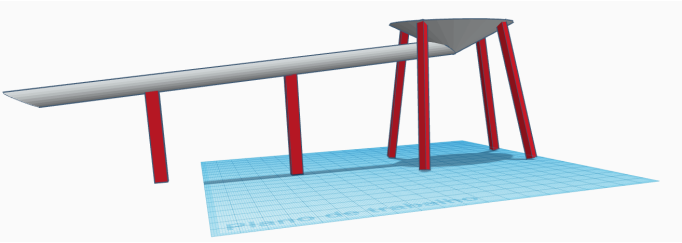

Figura 26 – Esquemático 3D do mecanismo de distribuição da ração lado 2.
Fonte: Autor

3.4 ESQUEMÁTICO 3D DO ALIMENTADOR

O esquemático 3D, retrata um esboço de como seria utilizado o alimentador automatizado e o mecanismo de distribuição da ração. Demonstrado nas Figuras 28, 29 e 30.

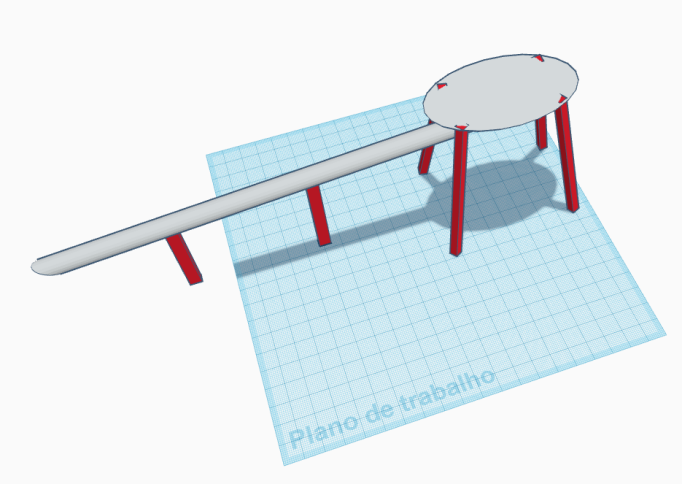

Figura 27 – Esquemático 3D do mecanismo de distribuição da ração lado3.
Fonte: Autor

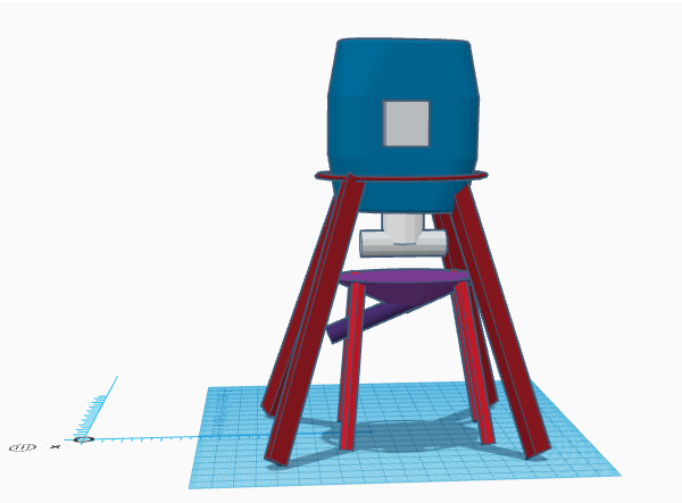

Figura 28 – Esquemático 3D do mecanismo de distribuição da ração lado um.
Fonte: Autor

3.5 ELABORAÇÃO DO CIRCUITO

Para a elaboração do circuito utilizou-se o programa Tinkercad, que é um programa de modelagem tridimensional online gratuito que roda no navegador da web, possui uma simplicidade e facilidade de uso, no qual pode-se montar e testar circuitos rapidamente.

3.5.1 Sequência de Montagem

Primeiramente necessita-se de uma protoboard e de um microcontrolador. Estes componentes são demonstrados na Figura 31.

A primeira ligação necessária é o positivo e o negativo do Arduino na protoboard para poder energizá-la. Com as devidas ligações, pôde-se conectar o sensor de distância ultrassônico. Essa montagem é demonstrada na Figura 32.

Com o sensor de distância devidamente posicionado, insere-se os LEDs e as devidas

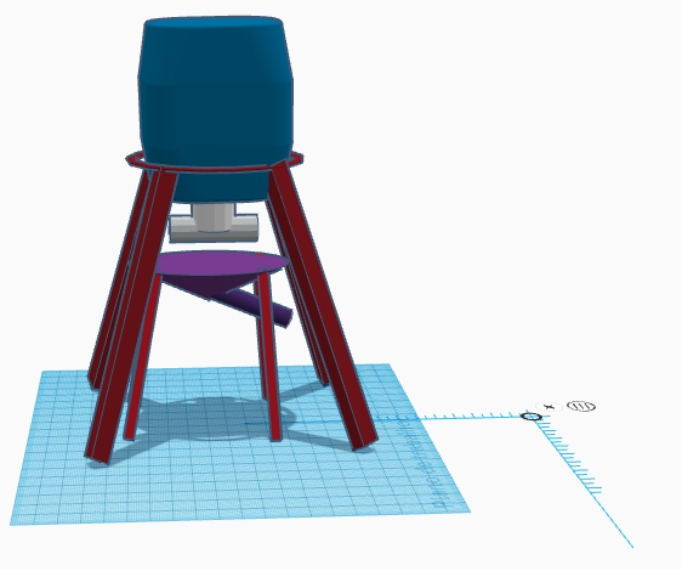

Figura 29 – Esquemático 3D do mecanismo de distribuição da ração lado dois.
Fonte: Autor

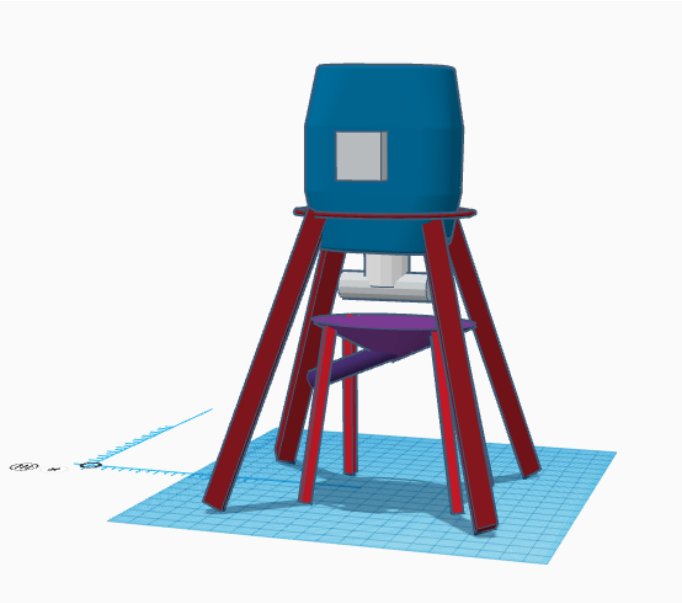

Figura 30 – Esquemático 3D do mecanismo de distribuição da ração lado três.
Fonte: Autor

resistências, os LEDs serão os responsáveis por transformar o sinal enviado pelo sensor ultrassônico em informação visual. Essa montagem é demonstrada na Figura 33.

Posteriormente, é necessário inserir o sensor de temperatura. Conectando-se seus terminais no positivo e negativo da protoboard, além de conectar com o pino do microcontrolador. Essas montagem é demonstrado na Figura 34.

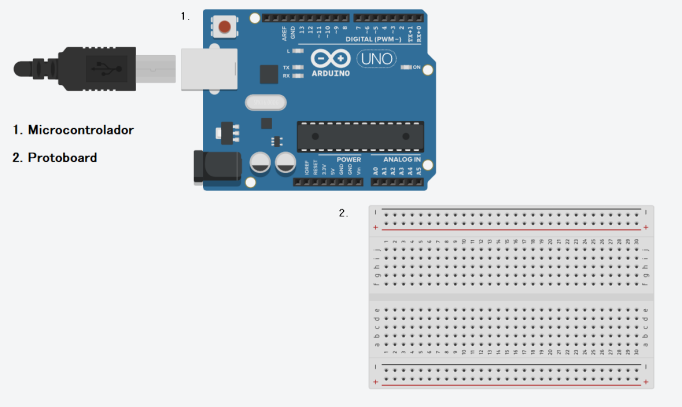

Figura 31 – Protoboard e microcontrolador.
Fonte: Autor

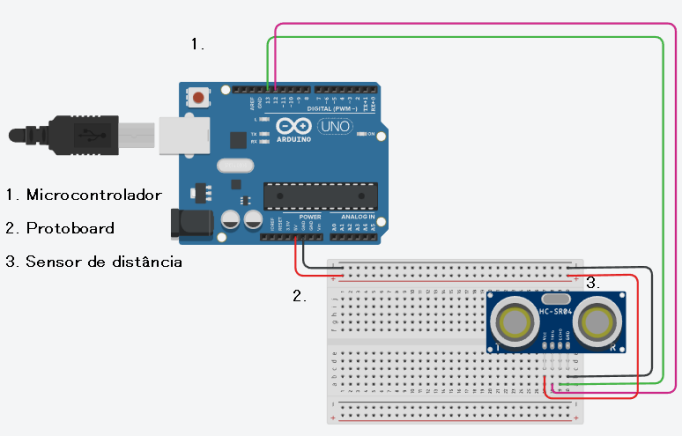

Figura 32 – Sensor ultrassônica e devidas conexões aplicadas.
Fonte: Autor

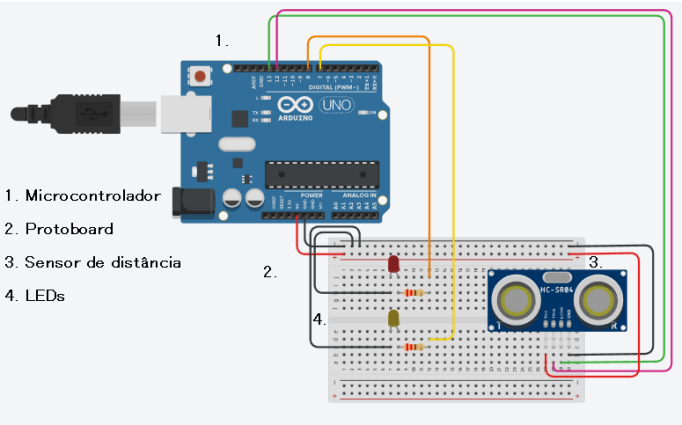

Figura 33 – LEDs e resistências aplicadas.
Fonte: Autor

Com os componentes aplicados pode-se inserir o motor, sem esquecer de inserir um módulo relé necessário pelas características do motor utilizado. Essa montagem é demonstrada na Figura 35.

Há a necessidade da aplicação de uma fonte de alimentação para o funcionamento do motor. Essa montagem é demonstrada na Figura 36.

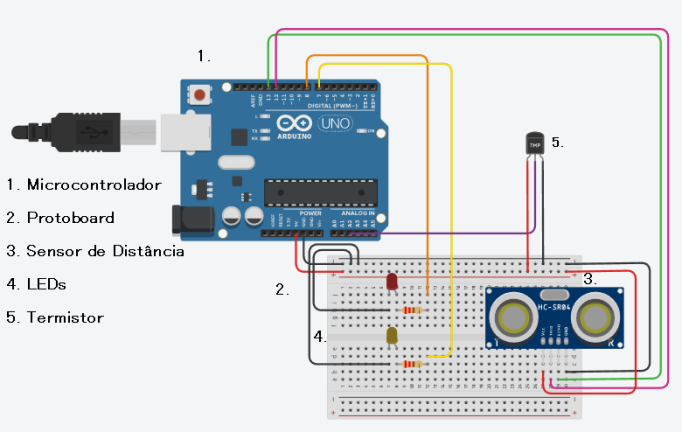

Figura 34 – Sensor de temperatura aplicado.
Fonte: Autor

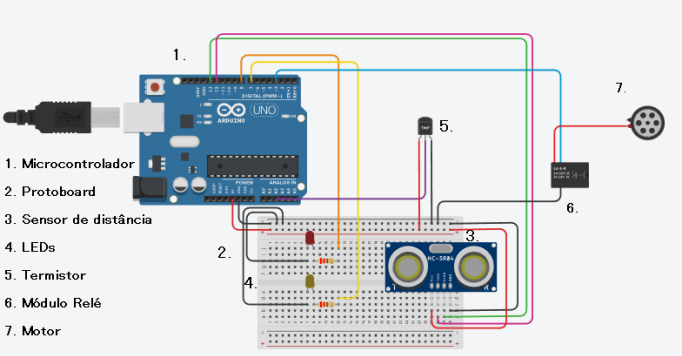

Figura 35 – Motor e módulo relé aplicados.
Fonte: Autor

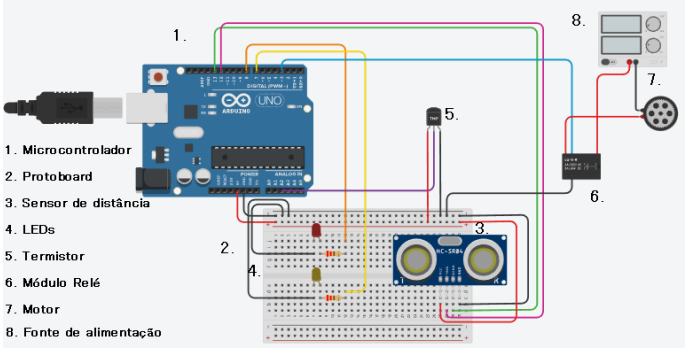

Figura 36 – Fonte de alimentação aplicada no circuito.
Fonte: Autor

Por fim, introduz-se um módulo LCD ao circuito. Uma segunda protoboard foi utilizada com o intuito de proporcionar uma melhor visualização do circuito, também foi aplicado um termistor utilizado como uma resistência elétrica variável, sendo possível assim controlar a quantidade de energia que o LCD irá receber, aumentando ou diminuindo o contraste da imagem reproduzida pelo LCD. Essa montagem é demonstrada na Figura 37.

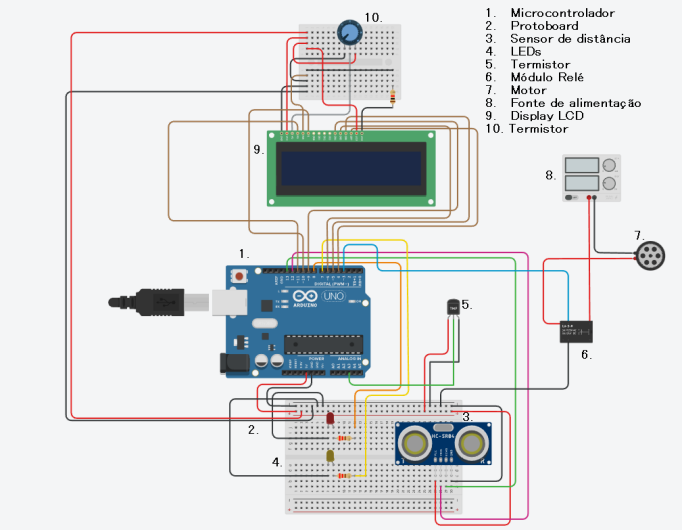

Figura 37 – Módulo LCD e potenciômetro aplicados no circuito.
Fonte: Autor

3.6 PROTÓTIPO ESTRUTURAL DESENVOLVIDO

Um protótipo estrutural simplificado foi construído fisicamente. O objetivo é obter uma amostragem na quantidade de ração fornecida, analisar a eficiência do motor proposto, e poder observar como reagiria a mola helicoidal. Neste protótipo utilizou-se um cano PVC 100mm como armazenador da ração (utilizado como substituto da bombona, pois para fins de testes não seria necessário grandes espaços), uma redução PVC 100mm X 75mm conectado ao armazenador, uma conexão T de PVC 75mm conectado a redução, e um cano PVC 75mm conectado a conexão T. Como demonstrado na Figura 38. A mola helicoidal acoplada no motor é demonstrada na Figura 39. O motor utilizado, conectado a uma das tampas de PVC, é demonstrado na Figura 40.

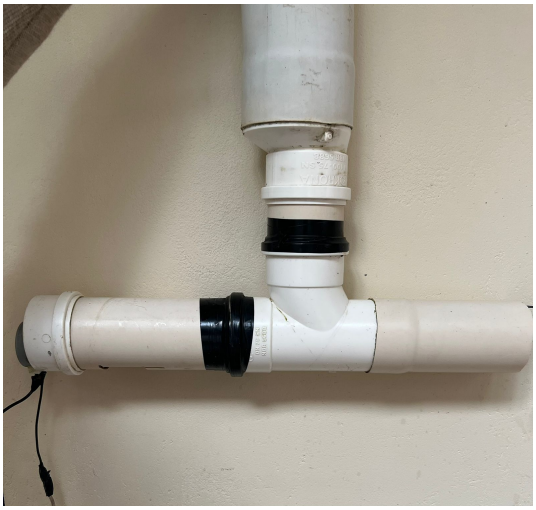

Figura 38 – Protótipo estrutural simplificado.
Fonte: Autor

Figura 39 – Mola helicoidal.
Fonte: Autor

Figura 40 – Motor conectado em uma das extremidades.
Fonte: Autor

3.7 AMOSTRAGEM DE ALIMENTAÇÃO

Foram efetuados quatro testes com o alimentador, marcando a quantidade de ração despejada por minuto. Os resultados são demonstrados nas Figuras 41, 42, 43 e 44. Com os testes, pôde-se observar que o alimentador despeja cerca de 200 gramas de ração por minuto. E que tanto o motor escolhido, como a mola helicoidal, suprem a tarefa proposta.

3.8 PROGRAMAÇÃO DESENVOLVIDA

Para desenvolver a programação da alimentação de um ecossistema, necessita-se primeiramente definir a quantidade de peso vivo presente. E para obter uma maior exatidão na distribuição da ração, é necessário saber qual a espécie de peixe criada, para adaptar a

Figura 41 – Quantidade de ração fornecida primeiro teste.
Fonte: Autor

Figura 42 – Quantidade de ração fornecida segundo teste.
Fonte: Autor

Figura 43 – Quantidade de ração fornecida terceiro teste.
Fonte: Autor

programação da forma mais eficiente possível a esse ecossistema. Para realizar a simulação foi hipotetizado um ecossistema com 100 tilápias contendo 0.3 kg cada, totalizando um

Figura 44 – Quantidade de ração fornecida quarto teste.
Fonte: Autor

peso vivo de 30,00 kg. Recentes estudos demonstram que é mais vantajoso alimentar as tilápias três vezes ao dia (**tilapiamanejo**). Com um peso vivo de 30,00 kg e considerando uma taxa de arraçoamento de 2%, necessita-se de um total de 600 gramas de ração diária, dividida entre três alimentações de 200 gramas cada.

3.8.1 Código Desenvolvido

3.8.1.1 Declaração das Variáveis

Primeiramente inicia-se as variáveis e seus respectivos pinos de conexão com o microcontrolador, seguindo a montagem do circuito construído previamente. Estas linhas são demonstrado na Figura 45.

```
////////////////////////////////////////////////////////////////////////////////////////
// DECLARAÇÃO DAS VARIÁVEIS E SEUS RESPECITVOS PINOS DE CONEXÃO COM O MICROCONTROLADOR //
////////////////////////////////////////////////////////////////////////////////////////

//É NECESSÁRIO A INCLUSÃO DA BLIBLIOTECA LIQUIDCRYSTAL UTILIZADA PARA O LCD

#include<LiquidCrystal.h>

LiquidCrystal lcd(11,10,9,6,5,4); //INICIA UM OBJETO LCD

const int sensorPin = A3; //PINO ANALÓGICO UTILIZADO PELO SENSOR DE TEMPERATURA

int trigger = 12; //SENSOR ULTRASSÔNICO

int echo = 13; //SENSOR ULTRASSÔNICO

int led = 8; //LED VERMELHO

int led2 = 7; //LED AMARELO

int motor = 3; //MOTOR ESPIRAL

int comida= 10; //VARIAVEL QUE INDICARÁ O NÍVEL DE RAÇÃO EXIBIDO NO DISPLAY LCD

int botao= 2; //BOTÃO
```

Figura 45 – Iniciação das variáveis.
Fonte: Autor

Posteriormente inicia-se as variáveis auxiliares que serão utilizadas no decorrer da programação. estas linhas são demonstradas na Figura 46.

```
27
28   ////////////////////////////////////////////////////////////////////////////////////////
29  // DECLARAÇÃO DAS VARIAVEIS QUE UTILIZAREMOS NAS FUNÇÕES DO PROGRAMA E PARA AUXILIO DOS SENSORES //
30 ////////////////////////////////////////////////////////////////////////////////////////
31
32 float temp = 0; //VARIÁVEL GLOBAL DO TIPO FLOAT UTILIZADO PARA A TEMPERATURA
33
34 float duracao = 0; //VARIÁVEL DE DURAÇÃO
35
36 float cm = 0; //VARIÁVEL EM CM
37
38 bool tem_comida; //VARIÁVEL GLOBAL UTILIZADO PARA INDICAR SE HÁ OU NÃO RAÇÃO NO RESERVATÓRIO
39
40 long int tempo_racao=0; //VARIÁVEL QUE RECEBE A SOMA DE TODO O TEMPO DECORRIDO NA ALIMENTAÇÃO
41
42 int estadobotao = 0; //VARIÁVEL UTILIZADA PARA O ESTADO DO BOTÃO
43
44 int contador_dia=0; //VARIÁVEL UTILIZADA COMO CONTADOR DIÁRIO SENDO RESETADO A CADA 7 DIAS
45
46 float tempo_para_desconto=0; //VARIÁVEL QUE ARMAZENA O RESULTADO DO TEMPO NO FORMATO DE BLOCOS DE 5 SEGUNDOS
47
48 long int tempo_racao_reset=0; //VARIÁVEL QUE ARMAZENA A SOMA DOS TEMPOS DE ALIMENTAÇÃO DE CADA DIA
```

Figura 46 – Iniciação das variáveis auxiliares.
Fonte: Autor

3.8.1.2 Configuração dos Pinos do Microcontrolador

A seguir configura-se os pinos do Arduíno como entradas e saídas. Estas linhas são demonstrado na Figura 47.

```
51   ////////////////////////////////////////////////////////////////////////////
52  // PROCEDIMENTO PARA CONFIGURAR AS ENTRADAS E AS SAÍDAS DO MICROCONTROLADOR //
53 ////////////////////////////////////////////////////////////////////////////
54
55 void setup() {
56
57   Serial.begin(9600); //INICIA A PORTA SERIAL E CONFIGURA A TAXA DE DADOS PARA 9600 BPS
58
59   pinMode(trigger, OUTPUT); //CONFIGURA O PINO ESPECÍFICO PARA FUNCIONAR COMO SAÍDA
60
61   pinMode(botao,INPUT); //CONFIGURA O PINO ESPECÍFICO PARA FUNCIONAR COMO ENTRADA
62
63   pinMode(echo, INPUT); //CONFIGURA O PINO ESPECÍFICO PARA FUNCIONAR COMO ENTRADA
64
65   pinMode(led, OUTPUT); //CONFIGURA O PINO ESPECÍFICO PARA FUNCIONAR COMO SAÍDA
66
67   pinMode(led2, OUTPUT); //CONFIGURA O PINO ESPECÍFICO PARA FUNCIONAR COMO SAÍDA
68
69   pinMode(motor, OUTPUT); //CONFIGURA O PINO ESPECÍFICO PARA FUNCIONAR COMO SAÍDA
70
71   pinMode(sensorPin, INPUT); //CONFIGURA O PINO ESPECÍFICO PARA FUNCIONAR COMO ENTRADA
72
73   lcd.begin(16, 2); //CONFIGURA O ESPAÇO DE TRABALHO DO DISPLAY
74
75 }
```

Figura 47 – Configuração dos pinos do microcontrolador.
Fonte: Autor

3.8.1.3 Procedimento Main

Após a declaração de todas as variáveis e a configuração dos seus respectivos pinos, deu-se início ao loop principal do código, onde todos os procedimentos serão listados e ocorrerão na ordem programada. Primeiramente é aplicado o procedimento do sensor de temperatura ("medir_temperatura"), responsável pela coleta da temperatura do ecossistema. Posteriormente é aplicado o procedimento do sensor de distância ("sensor"), responsável por captar o nível de ração do recipiente. a seguir, o procedimento da alimentação ("alimentar") é realizado, neste procedimento há a variação da quantidade de ração despejada ao ecossistema mediante a variação da temperatura. Posteriormente o procedimento da primeira hibernação ("timer_de_8_horas_descontado") é realizado, este procedimento é responsável por atualizar o contador diário de tempo, além de aplicar um descanso de oito horas ao mecanismo. a seguir, o procedimento de alimentação é executado novamente, seguido do procedimento da segunda hibernação ("timer_de_8_horas"), neste procedimento o mecanismo descansa por mais oito horas. Posteriormente o procedimento

de alimentação é executado pela terceira e ultima vez no dia. Por fim, o procedimento da terceira hibernação ("timer_de_8_horas") é aplicado, gerando mais um descanso de oito horas ao mecanismo e totalizando assim as 24 horas diárias. Ao final do procedimento, insere-se um contador que acrescenta +1 ao final de cada loop. Estas linhas são demonstrado na Figura 48.

```
502   ////////////////////////////////////////////////////////////////////////////////////////
503  // PROCEDIMENTO PRINCIPAL ONDE O ARDUÍNO REALIZA AS TAREFAS LISTADAS NO LOOP INFINITAMENTE //
504 ////////////////////////////////////////////////////////////////////////////////////////
505
506 void loop() {
507
508   Serial.print("Inicio do dia: agora sao 6 da manha \n"); //CONFIGURA A MENSAGEM QUE O SERIAL EXIBIRÁ
509
510   medir_temperatura(); //APLICA O PROCEDIMENTO MEDIR TEMPERATURA
511
512   sensor(); //APLICA O PROCEDIMENTO SENSOR
513
514   alimentar(); //APLICA O PROCEDIMENTO ALIMENTAR
515
516   timer_de_8_horas_descontado (); //APLICA O PROCEDIMENTO TIMER_DE_8_HORAS
517
518   alimentar(); //APLICA O PROCEDIMENTO ALIMENTAR
519
520   timer_de_8_horas(); //APLICA O PROCEDIMENTO TIMER_DE_8_HORAS
521
522   alimentar(); //APLICA O PROCEDIMENTO ALIMENTAR
523
524   timer_de_8_horas(); //APLICA O PROCEDIMENTO TIMER_DE_8_HORAS
525
526   if(contador_dia==7){ //SE O CONTADORR DO DIA ESTIVER COM O VALOR 7
527
528     tempo_racao=0; //DEFINE A VARIÁVEL tempo_racao COMO ZERO
529
530     contador_dia=0; //DEFINE A VARIÁVEL contador_dia COMO ZERO
531
532     Serial.print("\nFIM DA SEMANA\n"); //CONFIGURA A MENSAGEM QUE O SERIAL DO ARDUINO PRINTARÁ
533
534   }
535
536   contador_dia++; //ACRESCENTA +1 NO VALOR DA VARIÁVEL AO FINAL DO PROCEDIMENTO LOOP
537 }
```

Figura 48 – Loop Principal.
Fonte: Autor

3.8.1.4 Procedimento Sensor de Temperatura

Para dar início ao o procedimento do sensor de temperatura é necessário a utilização da função "map", demonstrada na linha 209. Esta função aproxima o valor encontrado pelo sensor a um número inteiro convertido em graus Celsius, para poder então ser exibido na tela LCD e no serial do microcontrolador. Estas linhas são demonstradas na Figura 49.

```
203   ////////////////////////////////////////////////
204  // PROCEDIMENTO PARA MEDIÇÃO DA TEMPERATURA //
205 ////////////////////////////////////////////////
206
207 void medir_temperatura(){
208
209   temp = map(((analogRead(sensorPin) - 20) * 3.04), 0, 1023, -40, 125); //VARIÁVEL temp RECEBE A TEMPERATURA MEDIDA
210
211   mostrar_temp(); //EXIBE A TEMPERATURA NO DISPLAY
212 }
```

Figura 49 – Inicialização do sensor de temperatura.
Fonte: Autor

3.8.1.5 Procedimento Sensor de Distância Ultrassônico

A seguir inicia-se o procedimento para configurar o sensor de distância ultrassônico. Neste procedimento é necessária a introdução da fórmula utilizada pelo sensor de distância para calcular a distância percorrida pelo pulso e fazer sua conversão para cm, este procedimento é demonstrada na linha 142. Para a composição desta parte do código são necessários quatro "ifs, o primeiro é definido por uma distância maior que 1,00 e menor

ou igual a 29,90 cm, se esta condição for respeitada atribui verdadeiro para a condição de haver comida e valor "0"para a variável comida. Estas linhas são demonstradas na Figura 50.

O segundo é definido por uma distância maior que 30,00 e menor ou igual a 45,00 cm, se esta condição for respeitada, liga o LED amarelo, atribui verdadeiro para a condição de haver comida e valor "1"para a variável comida. Estas linhas são demonstradas na Figura 51.

O terceiro é definido por uma distância maior que 45,00 e menor que 55,00 cm, se esta condição for respeitada, liga o LED vermelho, atribui verdadeiro para a condição de haver comida e valor "2"para a variável comida. Estas linhas são demonstradas na Figura 52.

O quarto é definido por uma distância maior que 55,00 cm, se esta condição for respeitada, atribui falso para a condição de haver comida e valor "2"para a variável comida, além de exibir a mensagem "nível crítico"no display LCD. Estas linhas são demonstradas na Figura 53.

```
133   ////////////////////////////////////////
134  // PROCEDIMENTO DO SENSOR DE DISTÂNCIA //
135 ////////////////////////////////////////
136
137 void sensor() {
138
139   digitalWrite(trigger, LOW); //SETAMOS LOW PARA O PINO DETERMINADO
140
141   digitalWrite(trigger, HIGH); //SETAMOS HIGH PARA O PINO DETERMINADO
142
143   digitalWrite(trigger, LOW); //SETAMOS LOW PARA O PINO DETERMINADO
144
145   duracao = pulseIn(echo, HIGH); //CAPTURA A DURAÇÃO DE UM PULSO EM UM PINO ESPECÍFICO
146
147   cm = duracao*0.034/2; //FÓRMULA UTILIZADA PARA CÁLCULO DA DISTÂNCIA PERCORRIDA PELO PULSO
148
149   if(cm <=29.9&&cm>1){ //SE A DISTÂNCIA FOR MAIOR QUE 5 E MENOR OU IGUAL A 29.9 CM
150
151     tem_comida=true; //DEFINE A VARIÁVEL tem_comida COMO VERDADEIRA
152
153     Serial.print("\nReservatorio de comida cheio!\n"); //CONFIGURA A MENSAGEM QUE O SERIAL EXIBIRÁ
154
155     comida=0; //DEFINE A VARIÁVEL comida COM VALOR 0
156   }
```

Figura 50 – Sensor de distância ultrassônico primeira condição.
Fonte: Autor

```
158   if (cm <=45&&cm>30){ //SE A CONDIÇÃO DE DISTÂNCIA MAIOR QUE 30 E MENOR OU IGUAL A 45 CM FOR SATISFEITA
159
160     digitalWrite(led2, HIGH); //SETA HIGH PARA O PINO DETERMINADO
161
162     tem_comida=true; //A VARIÁVEL tem_comida É DEFINIDA COMO VERDADEIRA
163
164     comida=1; //DEFINE A VARIÁVEL comida COM VALOR 1
165
166   } else {   //SE A CONDIÇÃO NÃO FOR SATISFEITA
167
168     digitalWrite(led2, LOW); //SETA LOW PARA O PINO DETERMINADO
169   }
```

Figura 51 – Sensor de distância ultrassônico segunda condição.
Fonte: Autor

```
171   if (cm<55&&cm>45){ //SE A CONDIÇÃO DE DISTÂNCIA MAIOR QUE 45 CM FOR SATISFEITA
172
173     digitalWrite(led, HIGH); //SETA HIGH PARA O PINO DETERMINADO
174
175     tem_comida=true; //A VARIÁVEL tem_comida É DEFINIDA COMO VERDADEIRA
176
177     comida=2; //DEFINE A VARIÁVEL comida COM VALOR 2
178   }
179
180   else {  //SE A CONDIÇÃO NÃO FOR SATISFEITA
181
182     digitalWrite(led, LOW); //SETA LOW PARA O PINO DETERMINADO
183   }
```

Figura 52 – Sensor de distância ultrassônico terceira condição.
Fonte: Autor

```
185   if (cm>55){ //SE A CONDIÇÃO DE DISTÂNCIA MAIOR QUE 50 CM FOR SATISFEITA
186
187     digitalWrite(led, HIGH); //SETA HIGH PARA O PINO DETERMINADO
188
189     tem_comida=false; //A VARIÁVEL tem_comida É DEFINIDA COMO FALSO
190
191     comida=2; //A VARIÁVEL comida É DEFINIDA COM VALOR 2
192
193     lcd.clear(); //LIMPA O DISPLAY
194
195     lcd.setCursor(1,0); //CONFIGURA A POSIÇÃO QUE O DISPLAY EXIBIRÁ A MENSAGEM
196
197     lcd.print("Nivel Critico"); //CONFIGURA A MENSAGEM QUE O DISPLAY EXIBIRÁ
198   }
199
200 }
```

Figura 53 – Sensor de distância ultrassônico quarta condição.
Fonte: Autor

3.8.1.6 Procedimento Tela LCD

Posteriormente inicia-se o procedimento para configurar o funcionamento do display LCD, utilizado para informar o nível da ração e a temperatura medida. Aplica-se três "ifs"incluindo as constantes utilizadas no procedimento do sensor de distância ultrassônico, utilizado para o controle de nível da ração. Valor "0"se o nível estiver bom, "1"se o nível estiver mediano e "2"caso o nível esteja baixo. Dentro de cada "if"há o comando para cada mensagem que deve ser mostrada no LCD assim como o comando para mostrar a temperatura medida. Este procedimento é demonstrada nas Figuras 54, 55 e 56.

```
78   ////////////////////////////////////////////////////////////////
79  // PROCEDIMENTO PARA CONFIGURAR O FUNCIONAMENTO DO DISPLAY LCD //
80 ////////////////////////////////////////////////////////////////
81
82 void mostrar_temp(){
83
84     if(comida==0){ //SE A VARIÁVEL COMIDA ESTIVER COM O VALOR 0
85
86         lcd.clear(); //LIMPA O DISPLAY
87
88         lcd.setCursor(1,0); //CONFIGURA A POSIÇÃO QUE O DISPLAY EXIBIRÁ A MENSAGEM
89
90         lcd.print("Nivel Cheio"); //CONFIGURA A MENSAGEM QUE O DISPLAY EXIBIRÁ
91
92         lcd.setCursor(3,1); //CONFIGURA A POSIÇÃO QUE O DISPLAY EXIBIRÁ A MENSAGEM
93
94         lcd.print("Temp:"); //CONFIGURA A MENSAGEM QUE O DISPLAY EXIBIRÁ
95
96         lcd.print(temp); //PRINTA O VALOR DE TEMPERATURA OBTIDO PELO SENSOR
97
98     }
```

Figura 54 – Inicialização tela LCD.
Fonte: Autor

```
100     if(comida==1){ //SE A VARIÁVEL COMIDA ESTIVER COM O VALOR 1
101
102         lcd.clear(); //LIMPA O DISPLAY
103
104         lcd.setCursor(1,0); //CONFIGURA A POSIÇÃO QUE O DISPLAY EXIBIRÁ A MENSAGEM
105
106         lcd.print("Nivel Medio"); //CONFIGURA A MENSAGEM QUE O DISPLAY EXIBIRÁ
107
108         lcd.setCursor(3,1); //CONFIGURA A POSIÇÃO QUE O DISPLAY EXIBIRÁ A MENSAGEM
109
110         lcd.print("Temp:"); //CONFIGURA A MENSAGEM QUE O DISPLAY EXIBIRÁ
111
112         lcd.print(temp); //PRINTA O VALOR DE TEMPERATURA OBTIDO PELO SENSOR
113
114     }
```

Figura 55 – Inicialização tela LCD.
Fonte: Autor

```
116     if(comida==2){ //SE A VARIÁVEL COMIDA ESTIVER COM O VALOR 2
117
118         lcd.clear(); //LIMPA O DISPLAY
119
120         lcd.setCursor(1,0); //CONFIGURA A POSIÇÃO QUE O DISPLAY EXIBIRÁ A MENSAGEM
121
122         lcd.print("Nivel Baixo"); //CONFIGURA A MENSAGEM QUE O DISPLAY EXIBIRÁ
123
124         lcd.setCursor(3,1); //CONFIGURA A POSIÇÃO QUE O DISPLAY EXIBIRÁ A MENSAGEM
125
126         lcd.print("Temp:"); //CONFIGURA A MENSAGEM QUE O DISPLAY EXIBIRÁ
127
128         lcd.print(temp); //PRINTA O VALOR DE TEMPERATURA OBTIDO PELO SENSOR
129     }
130
131 }
```

Figura 56 – Inicialização tela LCD.
Fonte: Autor

3.8.1.7 Procedimento da Alimentação

Para o procedimento da alimentação é necessário a utilização de sete "ifs", cada "if"é utilizado para alimentar uma quantidade específica de ração em um dado intervalo de temperatura. Estes valores são demonstrados na Tabela 2 e foram retirados de um estudo realizado pela Fundação Estadual de Pesquisa Agropecuária (FEPAGRO).

Tabela 2. Porcentagens utilizadas para determinada temperatura.

Temperatura da Água	**Porcentagens utilizadas**
Mais que 29 °C	02
29 - 27 °C	100
Menos que 27 - 25 °C	90
Menos que 25 - 22 °C	70
Menos que 22 - 20 °C	40
Menos que 20 - 18 °C	10
Menos que 18 °C	02

Cada "if"possui um determinado tempo de funcionamento do motor mediante a temperatura verificada no momento. Sabe-se que este mecanismo despeja cerca de 200 gramas de ração por minuto e que este ecossistema necessita de três alimentações de 200 gramas diárias. Sendo assim, para uma temperatura maior que 29 graus, utiliza-se 2% de um minuto, totalizando 12 segundos de alimentação. Para uma temperatura de 29 a 27 graus, utiliza-se 100% de um minuto, totalizando 60 segundos de alimentação. Para uma temperatura de 27 - 25 graus, utiliza-se 90% de um minuto, totalizando 54 segundos de alimentação. Para uma temperatura de 25 - 22 graus, utiliza-se 70% de um minuto,

totalizando 42 segundos de alimentação. Para uma temperatura de 22 - 19 graus, utiliza-se 40% de um minuto, totalizando 24 segundos de alimentação. Para uma temperatura de 19 - 18 graus, utiliza-se 10% de um minuto, totalizando seis segundos de alimentação. para uma temperatura menor que 18 graus, utiliza-se 2% de um minuto, totalizando 12 segundos de alimentação. Após o processo de alimentação o motor é desligado e o serial do microcontrolador exibe a quantidade de tempo que o motor ficou ligado em segundos. As variáveis "tempo_racao"e "tempo_racao_reset"armazenam o tempo total alimentado em mili-segundos para serem utilizadas posteriormente pelos contadores. Este procedimento é demonstrada nas Figuras 57 - 70.

```
216   ////////////////////////////////////
217  // PROCEDIMENTO PARA A ALIMENTAÇÃO //
218 ////////////////////////////////////
219
220 void alimentar() {
221
222   if(tem_comida==true){ //SE A CONDIÇÃO DE HAVER COMIDA FOR VERDADEIRA
223
224     if (temp>29){ //SE A TEMPERATURAR FOR MAIOR QUE 29 GRAUS
225
226         Serial.print("\nTemperatura agora: "); //CONFIGURA A MENSAGEM QUE O SERIAL EXIBIRÁ
227
228         Serial.print(temp); //EXIBE A TEMPERATURA NO SERIAL DO ARDUINO
229
230         mostrar_temp(); //EXIBE A TEMPERATURA NO DISPLAY
231
232         digitalWrite(motor, HIGH); //SETA HIGH PARA O PINO DO MOTOR
```

Figura 57 – Alimentação primeira condição.
Fonte: Autor

```
233
234       delay(1200); //DELAY DE 1,2 SEGUNDOS APLICADO
235
236         digitalWrite(motor, LOW); //SETA LOW PARA O PINO DO MOTOR
237
238         Serial.print("\nPeriodo de alimentacao de: "); //CONFIGURA A MENSAGEM QUE O SERIAL EXIBIRÁ
239
240         Serial.print(1200/1000); //SERIAL EXIBE O TEMPO TOTAL DE FUNCIONAMENTO DO MOTOR
241
242         Serial.print(" segundos\n"); //CONFIGURA A MENSAGEM QUE O SERIAL EXIBIRÁ
243
244         tempo_racao=tempo_racao+1200; //TEMPO TOTAL ALIMENTADO UTILIZADO PARA O CONTADOR
245
246         tempo_racao_reset=tempo_racao_reset+1200; //TEMPO TOTAL ALIMENTADO UTIIZADO PARA FAZER O DESCONTO DIÁRIO
247   }
```

Figura 58 – Alimentação primeira condição.
Fonte: Autor

```
 if (temp>=27&&temp<=29){ //SE A TEMPERATURAR FOR MAIOR OU IGUAL A 27 E MENOR OU IGUAL A 29 GRAUS

     Serial.print("\nTemperatura agora: "); //CONFIGURA A MENSAGEM QUE O SERIAL EXIBIRÁ

     Serial.print(temp); //EXIBE A TEMPERATURA NO SERIAL DO ARDUINO

     mostrar_temp(); //EXIBE A TEMPERATURA NO DISPLAY

     digitalWrite(motor, HIGH); //SETA HIGH PARA O PINO DO MOTOR

     delay(60000); //DELAY DE 60 SEGUNDOS APLICADO
```

Figura 59 – Alimentação segunda condição.
Fonte: Autor

```
       digitalWrite(motor, LOW); //SETA LOW PARA O PINO DETERMINADO

       Serial.print("\nPeriodo de alimentacao de: "); //CONFIGURA A MENSAGEM QUE O SERIAL EXIBIRÁ

       Serial.print(60000/1000); //SERIAL EXIBE O TEMPO TOTAL DE FUNCIONAMENTO DO MOTOR

       Serial.print(" segundos\n"); //CONFIGURA A MENSAGEM QUE O SERIAL EXIBIRÁ

       tempo_racao=tempo_racao+60000; //TEMPO TOTAL ALIMENTADO UTILIZADO PARA O CONTADOR

       tempo_racao_reset=tempo_racao_reset+60000; //TEMPO TOTAL ALIMENTADO UTIIZADO PARA FAZER O DESCONTO DIÁRIO
}
```

Figura 60 – Alimentação segunda condição.
Fonte: Autor

```
 if( temp>=25&&temp<=26.9){ //SE A TEMPERATURAR FOR MAIOR OU IGUAL A 25 E MENOR OU IGUAL A 26.9 GRAUS

     Serial.print("\nTemperatura agora: "); //CONFIGURA A MENSAGEM QUE O SERIAL EXIBIRÁ

     Serial.print(temp); //EXIBE A TEMPERATURA NO SERIAL DO ARDUINO

     mostrar_temp(); //EXIBE A TEMPERATURA NO DISPLAY

     digitalWrite(motor, HIGH); //SETA HIGH PARA O PINO DO MOTOR

     delay(4000); //DELAY DE 54 SEGUNDOS APLICADO
```

Figura 61 – Alimentação terceira condição.
Fonte: Autor

```
       digitalWrite(motor, LOW); //SETA LOW PARA O PINO DO MOTOR

       Serial.print("\nPeriodo de alimentacao de: "); //CONFIGURA A MENSAGEM QUE O SERIAL EXIBIRÁ

       Serial.print(54000/1000); //SERIAL EXIBE O TEMPO TOTAL DE FUNCIONAMENTO DO MOTOR

       Serial.print(" segundos\n"); //CONFIGURA A MENSAGEM QUE O SERIAL EXIBIRÁ

       tempo_racao=tempo_racao+54000; //TEMPO TOTAL ALIMENTADO UTILIZADO PARA O CONTADOR

       tempo_racao_reset=tempo_racao_reset+54000; //TEMPO TOTAL ALIMENTADO UTIIZADO PARA FAZER O DESCONTO DIÁRIO
}
```

Figura 62 – Alimentação terceira condição.
Fonte: Autor

```
 if (temp>=22&&temp<=24.9){ //SE A TEMPERATURA FOR MAIOR OU IGUAL A 22 E MENOR OU IGUAL A 24.9 GRAUS

     Serial.print("\nTemperatura agora: "); //CONFIGURA A MENSAGEM QUE O SERIAL EXIBIRÁ

     Serial.print(temp); //EXIBE A TEMPERATURA NO SERIAL DO ARDUINO

     mostrar_temp(); //EXIBE A TEMPERATURA NO DISPLAY

     digitalWrite(motor, HIGH); //SETA HIGH PARA O PINO DO MOTOR
```

Figura 63 – Alimentação quarta condição.
Fonte: Autor

```
       digitalWrite(motor, LOW); //SETA LOW PARA O PINO DO MOTOR

       Serial.print("\nPeriodo de alimentacao de: "); //CONFIGURA A MENSAGEM QUE O SERIAL EXIBIRÁ

       Serial.print(42000/1000); //SERIAL EXIBE O TEMPO TOTAL DE FUNCIONAMENTO DO MOTOR

       Serial.print(" segundos\n"); //CONFIGURA A MENSAGEM QUE O SERIAL EXIBIRÁ

       tempo_racao=tempo_racao+42000; //TEMPO TOTAL ALIMENTADO UTILIZADO PARA O CONTADOR

       tempo_racao_reset=tempo_racao_reset+42000; //TEMPO TOTAL ALIMENTADO UTIIZADO PARA FAZER O DESCONTO DIÁRIO
}
```

Figura 64 – Alimentação quarta condição.
Fonte: Autor

```
if (temp>=20&&temp<=21.9){ //SE A TEMPERATURA FOR MAIOR OU IGUAL A 20 E MENOR OU IGUAL A 21.9 GRAUS

      Serial.print("\nTemperatura agora: "); //CONFIGURA A MENSAGEM QUE O SERIAL EXIBIRÁ

      Serial.print(temp); //EXIBE A TEMPERATURA NO SERIAL DO ARDUINO

      mostrar_temp(); //EXIBE A TEMPERATURA NO DISPLAY

      digitalWrite(motor, HIGH); //SETA HIGH PARA O PINO DO MOTORR

      delay(24000); //DELAY DE 24 SEGUNDOS APLICADO
```

Figura 65 – Alimentação quinta condição
Fonte: Autor

```
       digitalWrite(motor, LOW); //SETA LOW PARA O PINO DO MOTOR

       Serial.print("\nPeriodo de alimentacao de: "); //CONFIGURA A MENSAGEM QUE O SERIAL EXIBIRÁ

       Serial.print(24000/1000); //SERIAL EXIBE O TEMPO TOTAL DE FUNCIONAMENTO DO MOTOR

       Serial.print(" segundos\n"); //CONFIGURA A MENSAGEM QUE O SERIAL EXIBIRÁ

       tempo_racao=tempo_racao+24000; //TEMPO TOTAL ALIMENTADO UTILIZADO PARA O CONTADOR

       tempo_racao_reset=tempo_racao_reset+24000; //TEMPO TOTAL ALIMENTADO UTIIZADO PARA FAZER O DESCONTO DIÁRIO
}
```

Figura 66 – Alimentação quinta condição.
Fonte: Autor

```
if (temp>=18&&temp<19.9){ //SE A TEMPERATURA FOR MAIOR OU IGUAL A 18 E MENOR OU IGUAL A 19.9 GRAUS

      Serial.print("\nTemperatura agora: "); //CONFIGURA A MENSAGEM QUE O SERIAL EXIBIRÁ

      Serial.print(temp); //EXIBE A TEMPERATURA NO SERIAL DO ARDUINO

      mostrar_temp(); //EXIBE A TEMPERATURA NO DISPLAY

      digitalWrite(motor, HIGH); //SETA HIGH PARA O PINO DO MOTOR

      delay(6000); //DELAY DE 6 SEGUNDOS
```

Figura 67 – Alimentação sexta condição
Fonte: Autor

```
       digitalWrite(motor, LOW); //SETA LOW PARA O PINO DO MOTOR

       Serial.print("\nPeriodo de alimentacao de: "); //CONFIGURA A MENSAGEM QUE O SERIAL EXIBIRÁ

       Serial.print(6000/1000); //SERIAL EXIBE O TEMPO TOTAL DE FUNCIONAMENTO DO MOTOR

       Serial.print(" Segundos\n"); //CONFIGURA A MENSAGEM QUE O SERIAL EXIBIRÁ

       tempo_racao=tempo_racao+6000; //TEMPO TOTAL ALIMENTADO UTILIZADO PARA O CONTADOR

       tempo_racao_reset=tempo_racao_reset+6000; //TEMPO TOTAL ALIMENTADO UTIIZADO PARA FAZER O DESCONTO DIÁRIO
}
```

Figura 68 – Alimentação sexta condição.
Fonte: Autor

```
  if (temp<18){ //SE A TEMPERATURA FOR MENOR QUE 18 GRAUS

      Serial.print("\nTemperatura agora: "); //CONFIGURA A MENSAGEM QUE O SERIAL EXIBIRÁ

      Serial.print(temp); //EXIBE A TEMPERATURA NO SERIAL DO ARDUINO

      mostrar_temp(); //EXIBE A TEMPERATURA NO DISPLAY

      digitalWrite(motor, HIGH); //SETA HIGH PARA O PINO DO MOTOR

      delay(1200); //DELAY DE 12 SEGUNDOS
```

Figura 69 – Alimentação sétima condição.
Fonte: Autor

```

       digitalWrite(motor, LOW); //SETA LOW PARA O PINO DO MOTOR

       Serial.print("\nPeriodo de alimentacao de: "); //CONFIGURA A MENSAGEM QUE O SERIAL EXIBIRÁ

       Serial.print(1200/1000); //SERIAL EXIBE O TEMPO TOTAL DE FUNCIONAMENTO DO MOTOR

       Serial.print(" Segundos\n"); //CONFIGURA A MENSAGEM QUE O SERIAL EXIBIRÁ

       tempo_racao=tempo_racao+1200; //TEMPO TOTAL ALIMENTADO UTILIZADO PARA O CONTADOR

       tempo_racao_reset=tempo_racao_reset+1200; //TEMPO TOTAL ALIMENTADO UTIIZADO PARA FAZER O DESCONTO DIÁRIO
  }
 }
}
```

Figura 70 – Alimentação sétima condição.
Fonte: Autor

3.8.1.8 Procedimento Timers

A seguir iniciou-se o procedimento necessário para o funcionamento do timer. Responsável por organizar três alimentações diárias com um intervalo de oito horas entre cada alimentação. Para este procedimento utilizou-se um loop demonstrado na linha "412"do código. Este loop é dividido em 5760 segmentos de 5000 mili-segundos, pois 5760 vezes 5000 mili-segundos resulta em 28800000 mili-segundos, que equivale a oito horas (esta conversão é necessária pois a função "delay"do Arduino trabalho em mili-segundos). Dentro deste loop utiliza-se outro "if", responsável por verificar se o botão contador foi pressionado. Caso esta condição seja verdadeira a tela LCD mostrará a quantidade de ração fornecida até o momento. Para este cálculo utilizou-se uma regra de três básica. Sabe-se que o alimentador despeja 200 gramas de ração por minuto, sendo assim, há uma relação de 200/60, simplificando obtém-se uma relação de 20/6. A variável "tempo_racao"é utilizada para armazenar todo o tempo utilizado na alimentação em mili-segundos, para ser transformado para segundos basta dividir por 1000. Por fim encontra-se a relação onde a quantidade em gramas consumida (QGC) será igual a "tempo_racao"/1000 x 20/6, sendo assim:

$$QGC = "tempo\_racao" * 20/6000. \tag{4}$$

O valor proveniente deste cálculo é exibido na tela LCD e representa a totalidade de ração em gramas despejada até o momento. Esta fórmula é demonstrada na linha 426. Este procedimento é demonstrada nas Figuras 71 e 72.

```
////////////////////////////////////////
// PROCEDIMENTO PARA O TIMER DE 8 HORAS //
////////////////////////////////////////

void timer_de_8_horas(){

  sensor(); //APLICA O PROCEDIMENTO DO SENSOR

  Serial.print("comeco da hibernacao ate a proxima alimentacao\n"); //CONFIGURA A MENSAGEM QUE O SERIAL EXIBIRÁ

  for (int i=0;i<5760;i++){ //LOOP PARA O TIMER DE 8 HORAS E CONTADOR DE RAÇÃO

    delay(5000); //DELAY DE 5 SEGUNDOS

        sensor(); //APLICA O PROCEDIMENTO DO SENSOR

        estadobotao = digitalRead(botao); //FAZ A LEITURA DO PINO DO BOTÃO

        if(estadobotao == HIGH){ //SE A CONDIÇÃO DE O BOTÃO PRESSIONADO FOR VERDADEIRA
```

Figura 71 – Procedimento para configuração do tempo.
Fonte: Autor

```
            lcd.clear(); //LIMPA O DISPLAY

            lcd.setCursor(1,0); //CONFIGURA A POSIÇÃO QUE O DISPLAY EXIBIRÁ A MENSAGEM

            lcd.print(((tempo_racao/1000)*(20/6))); //CONFIGURA A MENSAGEM QUE O DISPLAY EXIBIRÁ

            lcd.setCursor(7,0); //CONFIGURA A POSIÇÃO QUE O DISPLAY EXIBIRÁ A MENSAGEM

            lcd.print("Gramas"); //CONFIGURA A MENSAGEM QUE O DISPLAY EXIBIRÁ

            lcd.setCursor(1,1); //CONFIGURA A POSIÇÃO QUE O DISPLAY EXIBIRÁ A MENSAGEM

            lcd.print("Consumidas"); //CONFIGURA A MENSAGEM QUE O DISPLAY EXIBIRÁ
    }

  }

  medir_temperatura(); //MEDE A TEMPERATURA

  Serial.print("Fim da hibernacao\n"); //CONFIGURA A MENSAGEM QUE O SERIAL DO ARDUINO EXIBIRÁ

}
```

Figura 72 – Procedimento para configuração do tempo.
Fonte: Autor

Posteriormente é iniciado um segundo procedimento pro timer, este é praticamente igual ao procedimento do timer exemplificado anteriormente, porém, com uma diminuição no tempo do loop, este processo é utilizado na primeira alimentação do dia para atualizar o tempo de operação do microcontrolador. Este procedimento é necessário pois não se sabe quanto tempo o motor ficará ligado, devido a variação no tempo de alimentação em diferentes temperaturas. Sendo assim, ao decorrer algum tempo o alimentador não estaria mais programado para alimentar as 6 da manhã, o intervalo de oito horas seria mantido entre as alimentações porém, acabaria desregulando os horários planejados inicialmente o que poderia vir a atrapalhar na criação de algumas espécies mais sensíveis. Ao final deste procedimento as variáveis "tempo_para_reset"e "tempo_para_desconto"são ambas resetadas para o valor zero para reiniciarem então o processo de contagem. Esse procedimento é demonstrada nas Figuras 73 e 74.

```
////////////////////////////////////////////////////////////////
// PROCEDIMENTO PARA O TIMER DE 8 HORAS COM UM DESCONTO DE TEMPO //
////////////////////////////////////////////////////////////////

void timer_de_8_horas_descontado(){

  int i;  //DECLARAÇÃO DA VARIÁVEL UTILIZADA NO LOOP FOR VARIANDO DE 0 A 5670

  sensor(); //APLICA O PROCEDIMENTO DO SENSOR

  tempo_para_desconto=tempo_racao_reset;  //IGUALA O VALOR DA VARIÁVEL tempo_para_desconto COM A VARIÁVEL tempo_racao_reset

  tempo_para_desconto=tempo_para_desconto/5000;  //DIVIDE POR 5000 A VARIÁVEL DO TIPO FLOAT Tempo_para_desconto

  Serial.print("Comeco da hibernacao ate a proxima alimentacao\n"); //CONFIGURA A MENSAGEM QUE O SERIAL EXIBIRÁ

  for (i=0;i<5760-(int)tempo_para_desconto;i++){ //LOOP PARA O TIMER DE 8 HORAS COM O DESCONTO PROVENIENTE DO CÁLCULO ANTERIOR

    Serial.print("\nNesta primeira hibernacao do dia foram descontados: "); //CONFIGURA A MENSAGEM QUE O SERIAL EXIBIRÁ

    Serial.print(tempo_para_desconto); //SERIAL EXIBE O VALOR DA VARIÁVEL tempo_para_desconto

    Serial.print(" Delay's correspondentes a 5 segundos cada \n"); //CONFIGURA A MENSAGEM QUE O SERIAL EXIBIRÁ

    delay(5000); //DELAY DE 5 SEGUNDOS

        sensor(); //APLICA O PROCEDIMENTO DO SENSOR

        estadobotao = digitalRead(botao); //FAZ A LEITURA DO PINO DIGITAL ESPECÍFICO

        if(estadobotao == HIGH){ //SE A CONDIÇÃO DE O BOTÃO PRESSIONADO FOR VERDADEIRA
```

Figura 73 – Procedimento para configuração do tempo com atraso.

```
            lcd.clear(); //LIMPA O DISPLAY
            lcd.setCursor(1,0); //CONFIGURA A POSIÇÃO QUE O DISPLAY EXIBIRÁ A MENSAGEM
            lcd.print(((tempo_racao/1000)*(20/6))); //CONFIGURA A MENSAGEM QUE O DISPLAY EXIBIRÁ
            lcd.setCursor(7,0); //CONFIGURA A POSIÇÃO QUE O DISPLAY EXIBIRÁ A MENSAGEM
            lcd.print("Gramas"); //CONFIGURA A MENSAGEM QUE O DISPLAY EXIBIRÁ
            lcd.setCursor(1,1); //CONFIGURA A POSIÇÃO QUE O DISPLAY EXIBIRÁ A MENSAGEM
            lcd.print("Consumidas"); //CONFIGURA A MENSAGEM QUE O DISPLAY EXIBIRÁ
    }
  }
  medir_temperatura(); //MEDE A TEMPERATURA
  Serial.print("Fim da hibernacao\n"); //CONFIGURA A MENSAGEM QUE O SERIAL DO ARDUINO EXIBIRÁ
  tempo_racao_reset=0;  //SETA A VARIÁVEL tempo_racao_reset PARA VALOR 0
  tempo_para_desconto=0; //SETA A VARIÁVEL tempo_para_desconto PARA VALOR 0
}
```

Figura 74 – Procedimento para configuração do tempo com atraso.
Fonte: Autor

3.9 OPEN SOURCE

Como mencionado nos objetivos, este documento visa o desenvolvimento de um alimentador de baixo custo para psicultura, que será disponibilizado como base para qualquer pessoa seguir os passos e construir o seu próprio alimentador.

Primeiramente é necessário a construção da estrutura física do alimentador. Os passos para esta construção já foram demonstrados nos itens anteriores. Posteriormente é necessário a montagem do circuito do microcontrolador, utilizando todos os componentes necessários, a montagem de todo o circuito também é demonstrada nos itens anteriores. Por fim, há a necessidade de inserir o código no microcontrolador. Primeiramente é necessário acessar o site do Arduino "arduino.cc"e clicar na aba de software, como demonstrado na Figura 75.

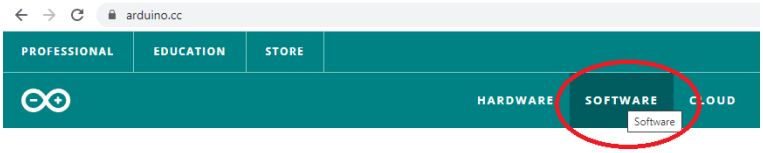

Figura 75 – Site Arduíno
Fonte: Autor

Após acessar a parte do software realiza-se o download, como demonstrado na Figura 76.

Figura 76 – Download software Arduíno
Fonte: Autor

Após o download e a instalação, o programa irá abrir a janela para a edição do código. Como demonstrado na Figura 77.

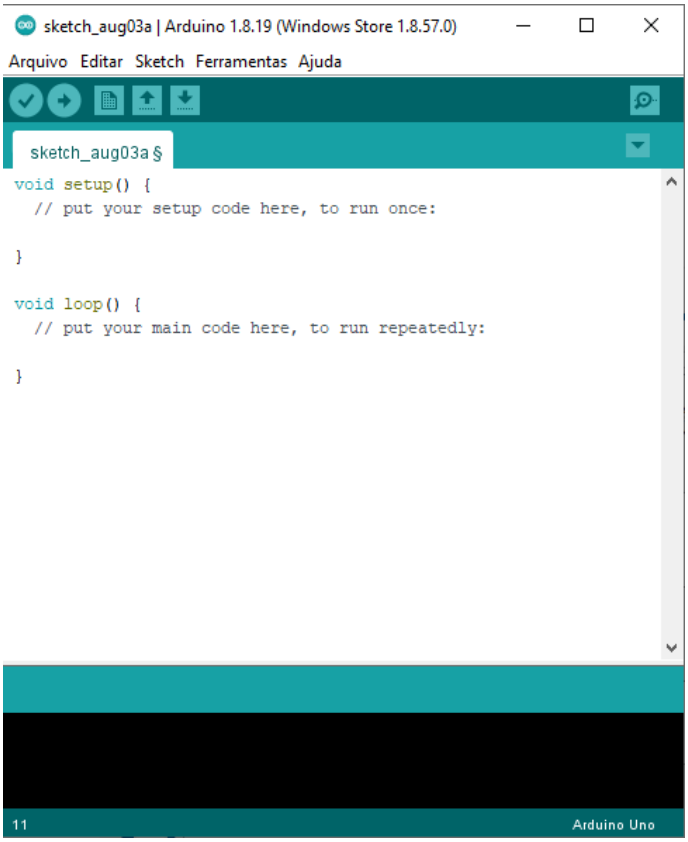

Figura 77 – Sketch primária do Arduíno
Fonte: Autor

Para a executar o código desenvolvido, basta copiar para o sketch do Arduíno todas as linhas apresentadas na parte de apêndice localizado no final do documento. Este procedimento é exemplificado na Figura 78.

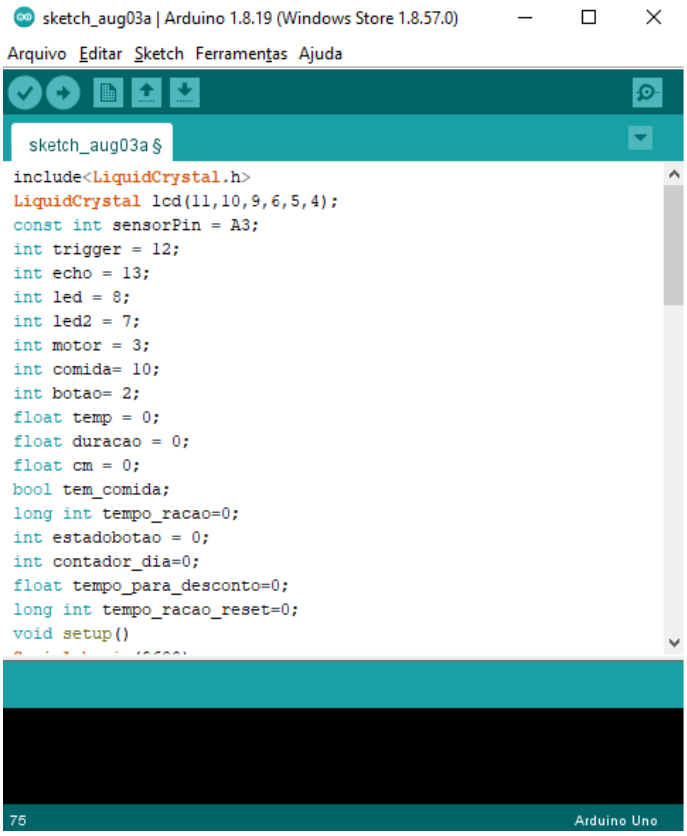

Figura 78 – Exemplo de sketch com o código inserido.
Fonte: Autor

4 RESULTADOS

4.1 FUNCIONAMENTO DO CIRCUITO

Primeiramente conecta-se o microcontrolador a uma fonte de energia e inicializa-se o código para observar se há a acusação de algum erro por parte do Arduíno ou dos periféricos utilizados. Pode-se observar que a tela LCD está funcionando corretamente assim como o potenciômetro. Este procedimento é demonstrada nas figuras 79 e 80.

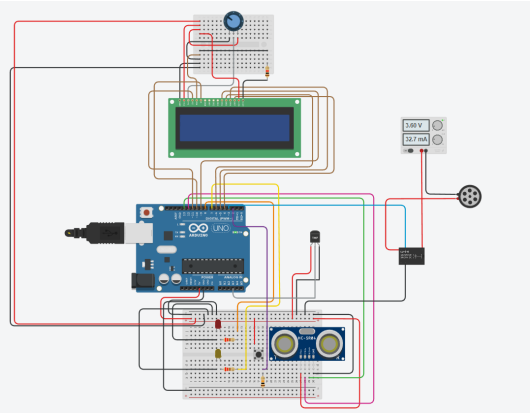

Figura 79 – Funcionamento do potenciômetro e LCD.
Fonte: Autor

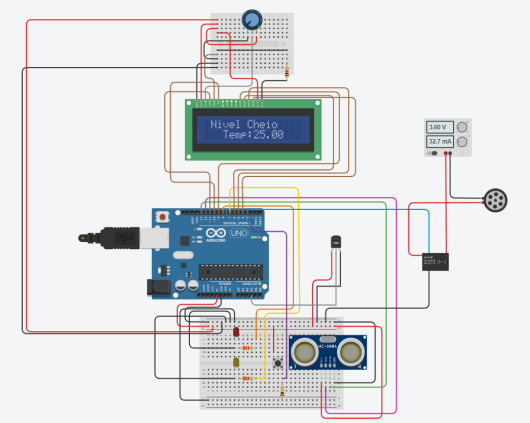

Figura 80 – Funcionamento do potenciômetro e LCD.
Fonte: Autor

Posteriormente altera-se a temperatura do ambiente e observa-se o comportamento do sensor de temperatura e a alteração ocorrendo corretamente na tela LCD. Este procedimento é demonstrada nas figuras 81 e 82.

A seguir altera-se a distância do sensor de distância ultrassônico para simular a alteração no nível da ração e poder observar as resposta dos LEDs e do monitor LCD. Pôde-se observar que ao captar uma distância de 22,40 cm, os dois LEDs permanecem desligados e o monitor LCD aponta de forma correta que o nível do recipiente está cheio. Com a alteração para uma distância de 35,60 cm, o LED amarelo acende e o monitor LCD

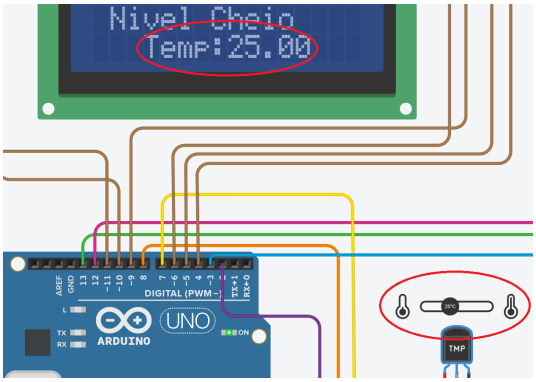

Figura 81 – Funcionamento do sensor de temperatura.
Fonte: Autor

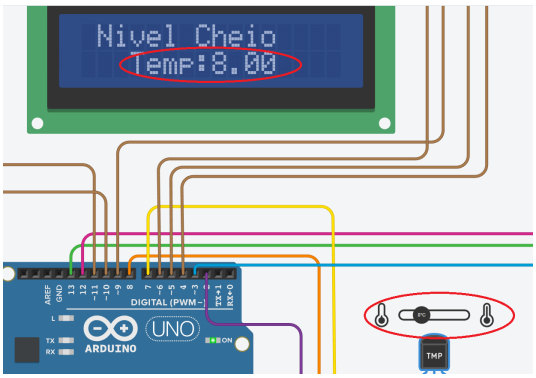

Figura 82 – Funcionamento do sensor de temperatura.
Fonte: Autor

aponta de forma correta que o recipiente contém um nível médio no momento. Alterando novamente a distância do sensor para 49,70 cm, o LED amarelo apaga e o vermelho acende, além de o monitor LCD apontar de forma correta que o nível do recipiente está baixo. Por fim, altera-se a distância para 58,10 cm e observa-se que o LED vermelho continua aceso e o amarelo continua apagado, além de o monitor LCD apontar de forma correta que o nível de ração no recipiente está crítico. Este procedimento é demonstrado em sequência nas figuras 83 - 86.

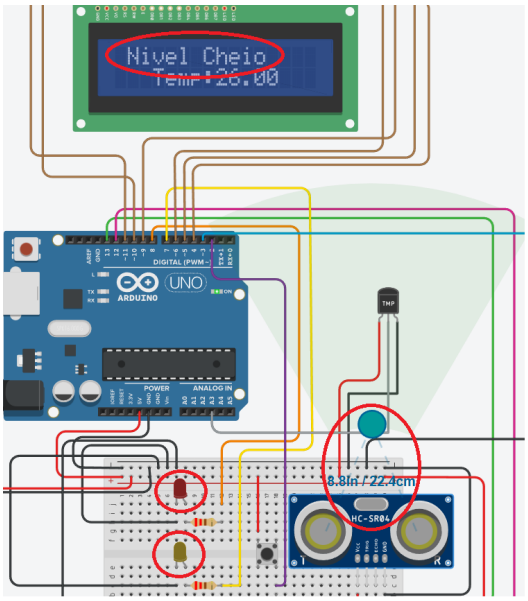

Figura 83 – Funcionamento do circuito em uma distância maior que 5 cm.
Fonte: Autor

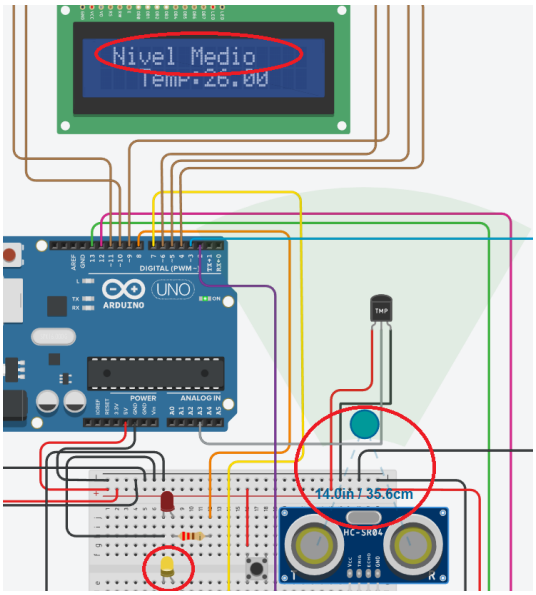

Figura 84 – Funcionamento do circuito em uma distância maior que 30 cm.
Fonte: Autor

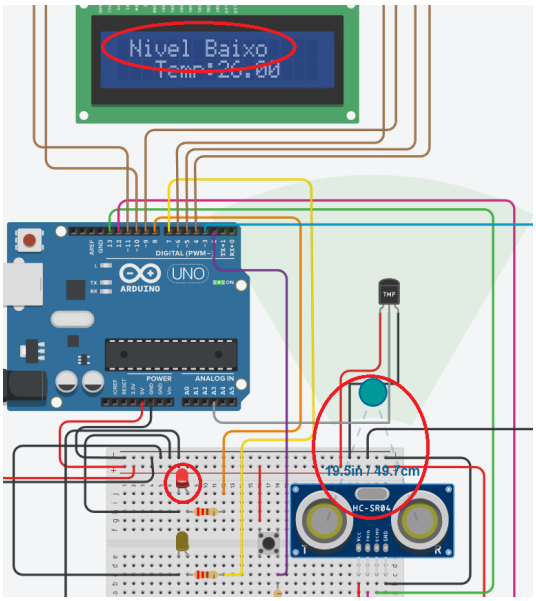

Figura 85 – Funcionamento do circuito em uma distância maior que 45 cm.
Fonte: Autor

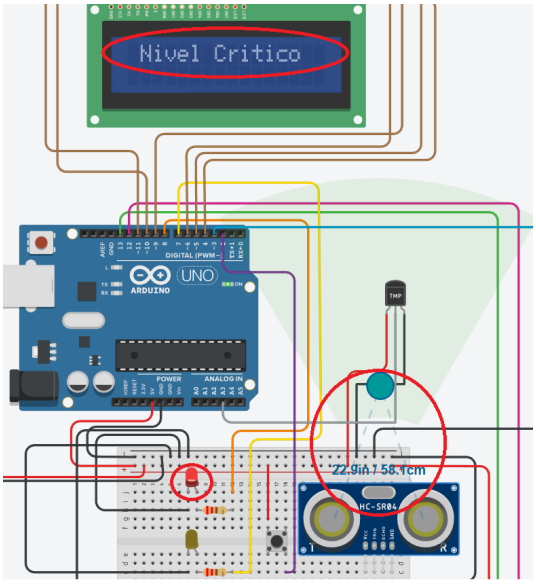

Figura 86 – Funcionamento do circuito em uma distância maior que 55 cm.
Fonte: Autor

A seguir analiza-se o funcionamento do procedimento de alimentação. Como este procedimento é baseado na temperatura captada pelo termistor, é necessário comparar duas diferentes temperaturas e observar se o funcionamento do motor está respeitando esta condição e se o serial está emitindo as mensagens necessárias. Primeiramente roda-se o código com uma temperatura de 25 graus, onde o motor deve ficar ligado por 54 segundos. Posteriormente roda-se o código com uma temperatura de 20 graus, onde o motor deve ficar ligado por 24 segundos. Este procedimento é demonstrada nas figuras 87 - 90.

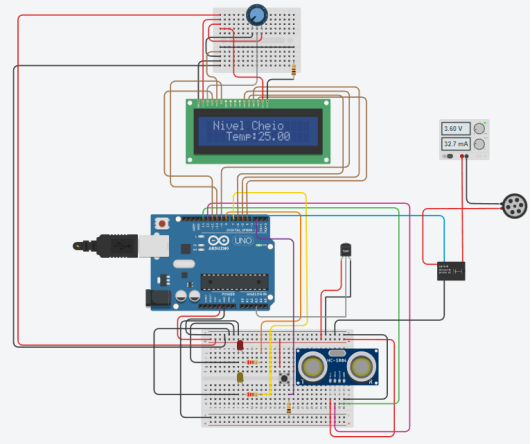

Figura 87 – Programa rodando com uma temperatura de 25 graus.
Fonte: Autor

```
Inicio do dia: agora sao 6 da manha

Reservatorio de comida cheio!

temperatura agora: 25.00
motor alimentador ficou ligado por: 54 segundos

Reservatorio de comida cheio!
Comeco da hibernacao ate a proxima alimentacao
```

Figura 88 – Serial temperatura 25 graus.
Fonte: Autor

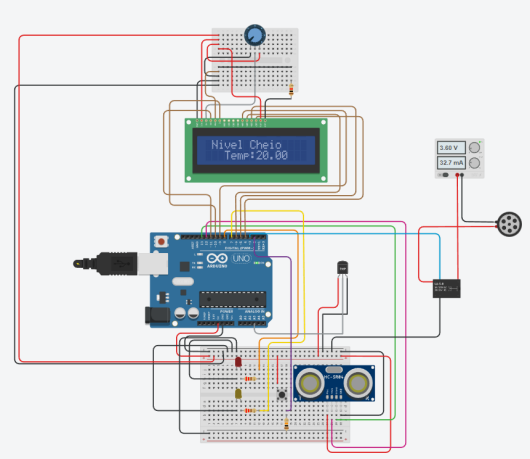

Figura 89 – Programa rodando com uma temperatura de 20 graus.
Fonte: Autor

```
temperatura agora: 20.00
motor alimentador ficou ligado por: 24 segundos

Reservatorio de comida cheio!
comeco da hibernacao ate a proxima alimentacao
```

Figura 90 – Serial temperatura 25 graus.
Fonte: Autor

Para analisar o funcionamento do timer programado no código é necessário a diminuição de escala de horas dia, pois o timer é programado para uma espera de oito horas. Sendo assim, reduz-se o tempo do timer arbitrariamente para um valor pequeno para observar o comportamento do circuito e as mensagens no serial. Primeiramente inicia-se o código na temperatura de 25 graus e observa-se que no horário do simulador 00:00:53 o motor ainda está trabalhando e que no horário 00:00:54 o motor desliga e o serial mostra a mensagem do tempo total do funcionamento do motor de forma correta. Posteriormente altera-se a temperatura para 19 graus e observa-se que no horário 00:01:21 o motor ainda esta trabalhando e que no horário 00:01:27 o motor desliga e o serial mostra a mensagem do tempo total de funcionamento do motor de forma correta. Este procedimento é demonstrada nas figuras 91, 92, 93 e 94. Posteriormente deixa-se o programa rodando por dois dias para conferir as mensagens no serial e poder verificar se o loop de três alimentações por dia está sendo respeitado. Este procedimento é demonstrado na figura 95.

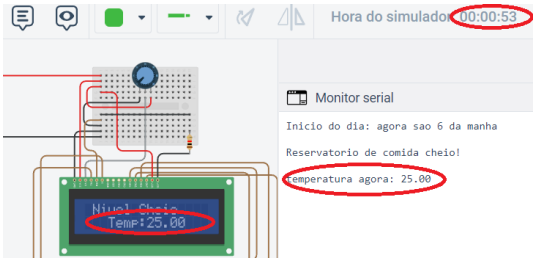

Figura 91 – Serial temperatura 25 graus.
Fonte: Autor

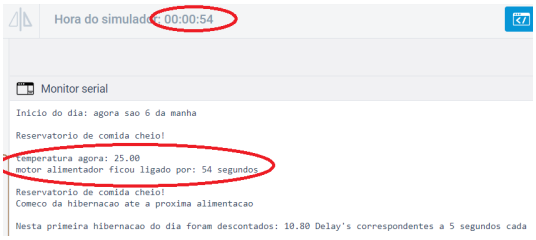

Figura 92 – Serial temperatura 25 graus.
Fonte: Autor

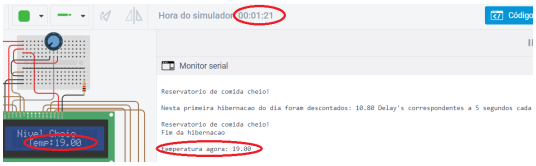

Figura 93 – Serial temperatura 19 graus.

Figura 94 – Serial temperatura 19 graus.
Fonte: Autor

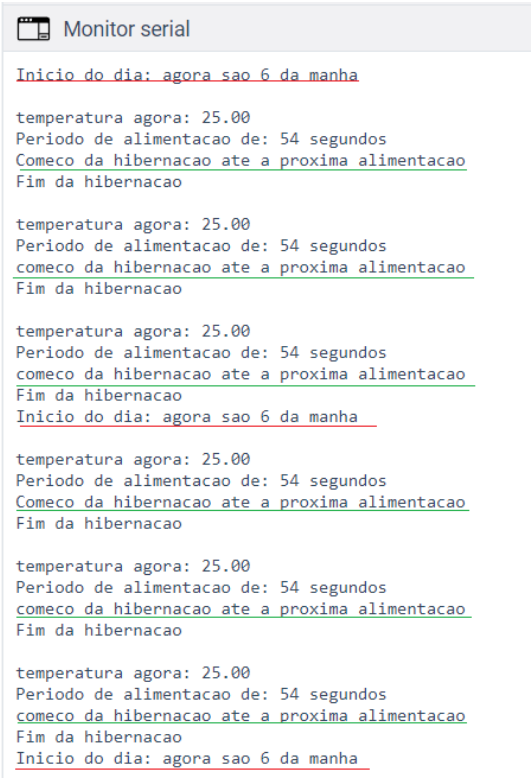

Figura 95 – Serial.
Fonte: Autor

Para realizar o teste do funcionamento do botão contador é necessário esperar algumas alimentações e contabilizar a quantidade alimentada para comparação. Primeiramente roda-se o código na temperatura de 25 graus e pressiona-se o botão ao final da primeira, segunda e terceira alimentação. Após a primeira alimentação, o LCD exibiu um consumo total de 180,36 gramas. Ao final da segunda alimentação, o LCD exibiu um consumo total de 370,72 gramas. E ao final da terceira alimentação o LCD exibiu uma consumo total de 541.08 gramas Estes procedimentos são demonstrados nas figuras 96 - 99.

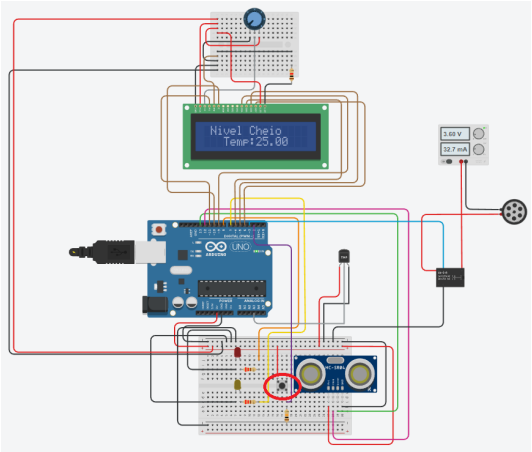

Figura 96 – Botão utilizado temperatura 25 graus.
Fonte: Autor

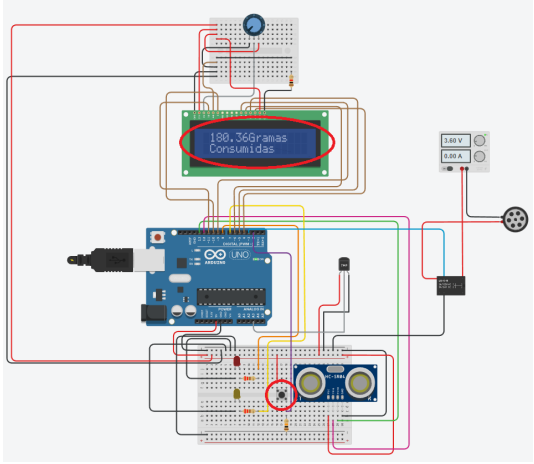

Figura 97 – Botão utilizado temperatura 25 graus.
Fonte: Autor

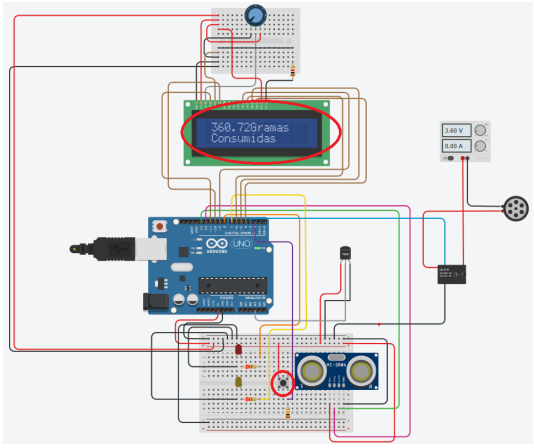

Figura 98 – Botão utilizado temperatura 25 graus.
Fonte: Autor

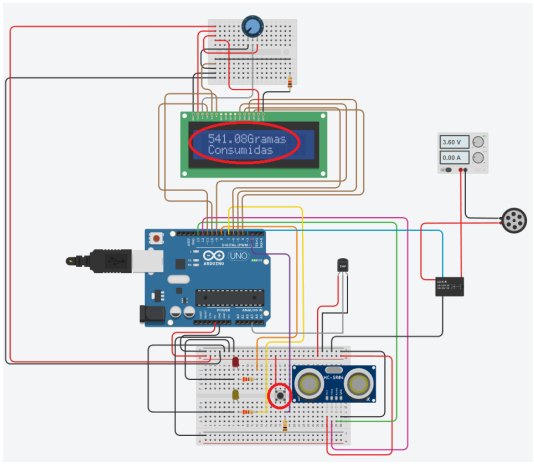

Figura 99 – Botão utilizado temperatura 25 graus.
Fonte: Autor

Realiza-se mais um teste com o botão contador, desta vez alterando o valor da temperatura após a primeira alimentação, realizada em 25 graus. Primeiramente a tela LCD exibe corretamente a quantidade alimentada na primeira alimentação de 25 graus.

A seguir altera-se a temperatura para 21 graus, o que implica em uma alimentação de 80 gramas, totalizando 260 gramas totais. Este procedimento é demonstrada nas figuras 100, 101 e 102.

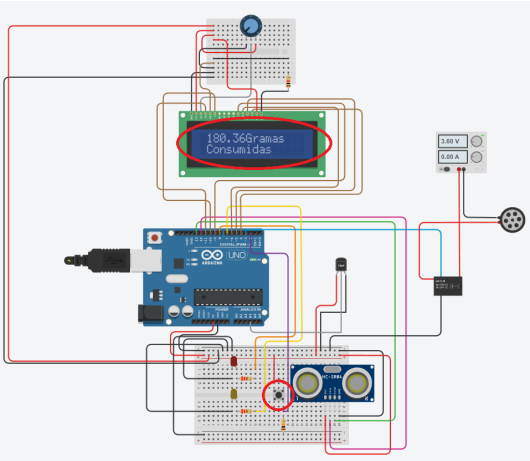

Figura 100 – Botão utilizado temperatura 25 graus segundo teste.
Fonte: Autor

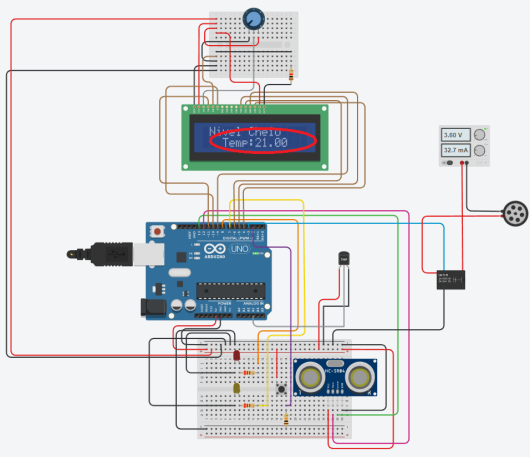

Figura 101 – Botão utilizado temperatura 21 graus.
Fonte: Autor

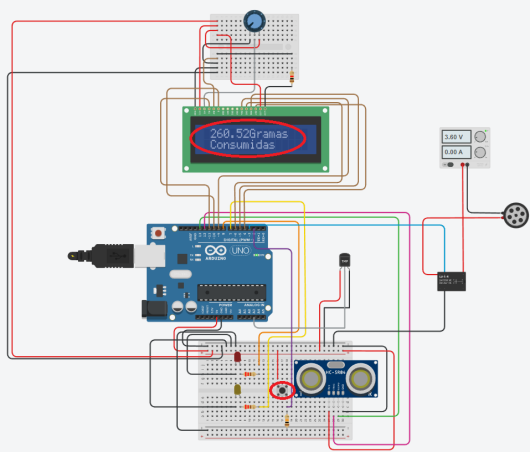

Figura 102 – Botão utilizado temperatura 21 graus.
Fonte: Autor

4.2 ANÁLISE DO VALOR DO PROJETO

Para ser possível realizar uma análise do projeto proposto perante as outras opções de mercado é necessário primeiramente definir o valor final do mecanismo proposto. A Tabela 3 apresenta os valores de todos os materiais utilizado na construção do alimentador, desde o micro-controlador e os periféricos necessários até os materiais mais simples como canos PVC, bombona alimentícia, arame galvanizado, dentre outros. Além de apresentar a quantidade necessária de cada item e a data de compra.

Tabela 3. Custos do Projeto.

Item	**Qtd**	**Unid**	**Descrição**	**Data**	**Valor Total**
01	1	pç	Arduino Uno	3/06/2022	R$ 85,00
02	1	pç	Bombona 55L	6/06/2022	R$ 199,00
03	1	pç	Caixa protetora	3/06/2022	R$ 75,32
04	1,00	m	Cano Pvc 75mm	6/06/2022	R$ 15,66
05	1	pç	Case Arduino	3/06/2022	R$ 14,90
06	1	pç	Display LCD	3/06/2022	R$ 27,00
07	1	pç	LED difuso 5mm Amarelo	3/06/2022	R$ 0,28
08	1	pç	LED difuso 5mm Vermelho	3/06/2022	R$ 0,28
09	1	pç	Módulo Relé Arduino	3/06/2022	R$ 8,90
10	1	pç	Mola Helicoidal	6/06/2022	R$ 68,00
11	1	pç	Potenciômetro Linear	3/06/2022	R$ 2,37
12	1	pç	Protoboard 400 pontos	3/06/2022	R$ 14,90
13	1	pç	Redução Pvc 100mm X 75mm	6/06/2022	R$ 21,74
14	1	m	Sensor de distância HC-Sr04	3/06/2022	R$ 15,00
15	1	pç	Tampão PVC 75mm	6/06/2022	R$ 7,00
16	1	pç	Tê Pvc 75mm	6/06/2022	R$ 31,90
17	2,00	m	Termistor NTC 100K	3/06/2022	R$23,80
-	-	-	-	-	-
-	-	-	-	**Valor Final**	R$ 611,05

4.2.1 Comparativo com Itens do Mercado

Há diversos tipos de alimentadores automatizados no mercado, alguns exemplos são demonstrados nas figuras 103, 104 e 105:

A maioria dos alimentadores automatizados do mercado seguem o mesmo princípio, utilizam um temporizador, onde o usuário especifica um intervalo de tempo no qual a ração será despejada ao ecossistema. O funcionamento desses alimentadores supre uma grande parte da demanda alimentícia dos animais, porém, não possuem controle de nível ou qualquer controle referente a temperatura, o que é fundamental para uma otimização no processo de criação de algumas espécies. Os valores dos alimentadores variam de R$ 530,00 a R$ 1395,00 sendo que as opções mais baratas não possuem um armazenador para a ração, sendo necessário comprar a parte, o que elevaria o valor em pelo menos R$ 200,00. O projeto elaborado neste trabalho custou aproximadamente R$ 610 valor inferior

Figura 103 – Ilustração de um alimentador automatizado, valor R$ 526,00.
Fonte: (MERCADO LIVRE, 2021b)

Figura 104 – Ilustração de um alimentador automatizado, valor R$ 739,00.
Fonte: (MERCADO LIVRE, 2021c)

aos disponíveis no mercado atualmente, logicamente, este preço é referente apenas ao custo dos componentes utilizados, sem contabilizar valor de mão de obra. Como o objetivo deste documento é a elaboração open source do alimentador, o valor de R$ 610 fica muito atrativo, visto que está próximo do valor mínimo de mercado e possui um controle muito maior.

Figura 105 – Ilustração de um alimentador automatizado, valor R$ 1395.
Fonte: (MERCADO LIVRE, 2021a)

5 CONCLUSÕES

Este trabalho de conclusão de curso buscou desenvolver um alimentador automatizado de baixo custo para piscicultores. A escolha do microcontrolador utilizado se mostrou muito efetiva, pois obteve uma rápida e fácil interação com o código, sendo possível a modificação das variáveis de cada ecossistema proposto, conseguindo assim uma melhor eficiência alimentícia para diferentes espécies de peixes.

Através dessa pesquisa foi possível desenvolver um mecanismo de alimentação de peixes que se adéqua aos mais diversos ecossistemas de criações. Com um valor de custo final aproximado de R$ 610,00 o valor está abaixo da média de preços do mercado atual. Outra questão que pode ser observada é que a partir da sua implantação, o piscicultor obtém um maior controle na alimentação e eficiência na criação dos peixes, resultando em um melhor custo benefício, proporcionando um maior ganho de peso na criação e um menor desperdício de alimentos. Sendo assim, para aqueles que desejarem implantar esse sistema de alimentação de peixes em suas propriedades, farão um investimento menor com um resultado melhor em ganhos de peso, menos desperdício de alimentos, pouca manutenção e pouca mão de obra.

Com a utilização dos microcontroladores, é possível a instalação de mais periféricos caso o produtor necessite de um maior controle. Elevaria o valor do projeto, porém, ainda seria um valor inferior quando comparado com as opções atuais do mercado.

O Brasil possui uma enorme capacidade para se tornar líder na criação de peixes em cativeiro, a constante evolução dos números na última década é um forte indicativo desta capacidade. Com pequenos ajustes na produção, pode-se atingir grandes resultados, aumentando em muito a lucratividade de pequenos e médio produtores, resultando na possibilidade de expansão da produção, visando não só o mercado interno como o mercado externo. Aumentando a exportação do país há a injeção de capital na economia interna, além de gerar inúmeros empregos diretos e indiretos, contribuindo para uma economia mais sólida em todo território nacional.

REFERÊNCIAS

ADILSON THOMSEN. **Como conectar o Sensor Ultrassônico HC-SR04 ao Arduino**. [*S.l.*: *s.n.*], 2011. Disponível em: https://www.reichelt.com/de/en/led-5-mm-low-current-2-ma-green-led-5mm-2ma-gn-p21625.html. Acesso em 06/05/2022.

ALFABOT. **Caixa Plástica Multiuso C281814C Média com Fecho**. [*S.l.*: *s.n.*], 2022. Disponível em: https://www.alfabot.com.br/caixa-plastica-multiuso-c281814c-media-com-fecho?utm_source=google&utm_medium=Shopping&utm_campaign=caixa-plastica-multiuso-c281814c-media-com-fecho&inStock. Acesso em 06/06/2022.

AMÉRICO, Jonas de Jesus. Automação residencial de baixo custo.

ANALÓGICA INSTRUMENTAÇÃO E CONTROLE. **Termistores**. [*S.l.*: *s.n.*], 2013. Disponível em: http://www.analogica.com.br/arquivos/nt-011-termistores.pdf. Acesso em: 06/05/2021.

BARBACENA, Ilton L; FLEURY, Claudio Afonso. Display lcd. **Outubro**, 1996.

BRAGA, Newton C. **Relés: Circuitos e aplicações**. [*S.l.*]: Editora Newton C. Braga, 2017.

CARVALHO FILHO, Luiz Henrique de *et al.* Estudo e fabricação de termistores de ZnO dopados com CU por reação de combustão com otimização de eletrodos. Universidade Federal da Paraıba, 2021.

COSTA GOMES, Emanuelle da; SILVA, Victor Leandro da;
CARVALHO, Gessymara Suele Almeida Sousa; EVANGELISTA, Luiz Fernando Vieira;
COSTA GRANJEIRO, Denise da; SOUSA, Bruce Allan Lima de *et al.*
PROTOTIPAGEM DE UMA TRENA ELETRÔNICA OPEN-SOURCE.

CRISTIANO BERTULUCCI SILVEIRA. **Sensor de Temperatura: Encontre o Tipo Ideal para sua Aplicação**. [*S.l.*: *s.n.*], 2018. Disponível em: https://www.citisystems.com.br/sensor-de-temperatura/. Acesso em 06/05/2022.

CRUZ, Roniel Ferreira; CORREA, Camila Santos; SILVA, Wilton Lacerda.
DESENVOLVIMENTO DE UM SENSOR DE TEMPERATURA DE BAIXO CUSTO APLICADO AO CONTROLE DA QUALIDADE DE VACINAS.

EBC, EMPRESA BRASIL DE COMUNICAÇÃO. **ONU diz que população mundial chegará a 8, 6 bilhões de pessoas em 2030**. [*S.l.*: *s.n.*], 2021. Disponível em: https://agenciabrasil.ebc.com.br/internacional/noticia/2017-06/onu-diz-que-populacao-mundial-chegara-86-bilhoes-de-pessoas-em-2030. Acesso em: 02/02/2022.

EBC, EMPRESA BRASIL DE COMUNICAÇÃO. **PRODUÇÃO**. [*S.l.*: *s.n.*], 2020. Disponível em: https://www.peixebr.com.br/producao/#:~:text=A%20produ%C3%A7%C3%A3o%20de%20peixes%20cultivados,%E2%80%9D%2C%20informa%20a%20PEIXE%20BR.. Acesso em: 19/05/2022.

ELEC FREAKS. **Ultrasonic ranging module HC-SR04 datasheet**. [*S.l.*: *s.n.*], 2018. Disponível em: https://cdn.sparkfun.com/datasheets/Sensors/Proximity/HCSR04.pdf. Acesso em 05/05/2022.

ELETRONIK REICHELT. **LED 5MM 2MA GN LED**. [*S.l.*: *s.n.*], 2019. Disponível em: :https://www.reichelt.com/de/en/led-5-mm-low-current-2-ma-green-led-5mm-2ma-gn-p21625.html. Acesso em 05/05/2022.

EVANS, Martin; NOBLE, Joshua; HOCHENBAUM, Jordan. **Arduino em ação**. [*S.l.*]: Novatec Editora, 2013.

FABÍOLA DE LUCA COIMBRA BOMTEPO, MARCELO DE SOUSA NUNES, VALÉRIA GEDANKEN, WENDERSON ARAÚJO, TONY OLIVEIRA, PLÍNIO QUARTIM. **Piscicultura: alimentação**. [*S.l.*: *s.n.*], 2019. Disponível em: https://www.cnabrasil.org.br/assets/arquivos/263-Piscicultura-Alimenta%C3%A7%C3%A3o_191025_203233.pdf. Acesso: em 20/05/2022.

FOGAÇA, Fabíola. O protagonismo do Brasil na produção mundial de pescado. Empresa Brasileira de Pesquisa Agropecuaria (Embrapa), 2020.

FREITAS, Leonardo Marques de *et al.* Prototipagem e construção de um acoplador para óculos com comunicação bluetooth para fins de automação industrial e transmissão de dados. Universidade Federal Rural do Semi-Árido, 2019.

FREITAS, Luciano Barbosa. Apresentação da utilização e aplicação da Metodologia SMED na Indústria. **Engenharia de produção: Inovação na indústria 4.0**, p. 36.

JAVED, Adeel. **Criando projetos com Arduino para a Internet das Coisas**. [*S.l.*]: Novatec Editora, 2017.

JESUS, Walisson Xavier de. Controle de eficiência energética do tunel de retraçãoda embaladora da fábrica heineken (Alagoinhas-BA). Instituto Federal de Sergipe-IFS, 2018.

LIMA FERREIRA, Lucas Eduardo de; ALVES, Alisson Caetani; AMARAL, Ana Maria Santana; GÓES, Bruno César. Análise da concentração do mercado de exportação de pescados. **Revista em Agronegócio e Meio Ambiente**, v. 14, Supl. 1, p. 1–13, 2021.

LOMBAS, Jaqueline da Silva; CARDOSO, Liliane Maria da Silva; CARVALHO, Mayke Gustavo de; SALATINI, Wagner Matheus da Silva. Alimentador automático para Pet. 136, 2020.

MADEIRO, C. Com 12% da água doce mundial, o Brasil cuida bem dela. **Entenda por**, 2015.

MCROBERTS, Michael. **Arduino básico**. [*S.l.*]: Novatec Editora, 2018.

MEKANUS ROBÓTICA EDUCACIONAL. **Protoboard Breadboard 400 Pontos Furos Pinos Arduino Pic Arm**. [*S.l.*: *s.n.*], 2016. Disponível em: https://mekanus.mercadoshops.com.br/MLB-859860069-protoboard-breadboard-400-pontos-furos-pinos-arduino-pic-arm-_JM. Acesso em: 06/05/2021.

MERCADO LIVRE. **Alimentador Automático Para Lagos De Carpas Tratador Peixes**. [*S.l.*: *s.n.*], 2021. Disponível em: https://produto.mercadolivre.com.br/MLB-1586662417-alimentador-automatico-para-lagos-de-carpas-tratador-peixes-_JM#position=5&search_layout=stack&type=item&tracking_id=1a7f714e-d535-47e6-9a0b-d217fdb794a5. Acesso em: 06/06/2021.

MERCADO LIVRE. **Alimentador Tratador Automático Animais Timerfonte Divisor Y**. [*S.l.*: *s.n.*], 2021. Disponível em: https://produto.mercadolivre.com.br/MLB-2100121815-alimentador-tratador-automatico-animais-timerfonte-divisor-y-_JM?searchVariation=173960349493#searchVariation=173960349493&position=17&search_layout=stack&type=item&tracking_id=1a7f714e-d535-47e6-9a0b-d217fdb794a5. Acesso em: 06/06/2021.

MERCADO LIVRE. **Alimentador Tratador Automático Animais Timerfonte Divisor Y**. [*S.l.*: *s.n.*], 2021.

MOREIRA, Michele Paulino Carneiro; ROMEU, Mairton Cavalcante; ALVES, Francisco Regis Vieira; SILVA, Francisco Roberto Oliveira da. Contribuições do Arduino no ensino de Fısica: uma revisão sistemática de publicações na área do ensino. **Caderno Brasileiro de Ensino de Fısica**, Universidade Federal de Santa Catarina (UFSC), v. 35, n. 3, p. 721–745, 2018.

NEWARK. **STARTER KIT, ARDUINO, UNO BOARD, ITALIAN**. [*S.l.*: *s.n.*], 2021. Disponível em: https://www.newark.com/arduino/k010007/starter-kit-arduino-uno-board/dp/29X1574?ICID=I-HP-STM7REC-RP-1. Acesso em 02/02/2022.

OLIVEIRA, Raphael Henrique; SANTOS, TR. Lâmpadas de LED: vantagens e desvantagens em instalações elétricas. **Instituto Federal de Goiás. Disponıvel em: http://w2.ifg.edu.br/goiania/mecanica/images/Arquivos/TCC_MECANICA/ata2014-01me_autor_raphael_e_tulio_lampadas_led_vantagens_e_desvantagens.pdf. Acesso em 30/04/2021**, v. 30, 2020.

PATSKO, Luis Fernando. Tutorial–Aplicações, Funcionamento e Utilização de Sensores. **Maxwell Bohr: Instrumentação eletrônica**, p. 84, 2006.

PETRUZELLA, Frank. **Motores Elétricos e Acionamentos: Série Tekne**. [*S.l.*]: Bookman Editora, 2013.

PINTO, Sinézio Henrique. Desenvolvimento de software para controle de dispositivos elétricos via bluetooth e comando de voz. Universidade Estadual Paulista (UNESP), 2016.

RANGEL, Marcelle Gusmão; SILVA, Paula Barsaglini; GUEDE, José Ricardo Abalde. Led–Iluminação de Estado Sólido. **São José do Campos**, 2009.

ROBOCORE. **Case para BlackBoard UNO**. [*S.l.*: *s.n.*], 2022. Disponível em: https://www.robocore.net/acessorios-arduino/case-para-blackboard-uno-r3?gclid=CjwKCAjwp7eUBhBeEiwAZbHwkb8_7o_QwLIZNcwv3uAhYr08Q-ujXIhudobM90Oaf4ISlGBLUO7_9RoC7XkQAvD_BwE. Acesso em 07/06/2022.

SILVA, Otavio Henrique da; LOCASTRO, João Karlos; UMADA, Murilo Keith; POLASTRI, Paula; NETO, Generoso De Angelis. Proposta de gerenciamento de resıduos sólidos para um empreendimento industrial. **Revista Técnico-Cientıfica**, n. 9, 2017.

TANNUS, Alexandre Moraes. Arduino: Display LCD, 2018.

USINAINFO. **Sensor de Temperatura NTC 100K 3950 para Impressora 3D**. [*S.l.*: *s.n.*], 2019. Disponível em: https://www.usinainfo.com.br/impressora-3d-e-cnc-arduino/sensor-de-temperatura-ntc-100k-3950-para-impressora-3d-5717.html. Acesso em 06/05/2022.

VENTURI, Jiancarlo. Análise do sistema mecânico da roda d'água do Museu Weg com o intuito de aumentar a eficiência no processo de geração de energia, 2019.

WANZELER, Tiago Machado. AutomaÇÃo residencial de baixo custo utilizando a plataforma de prototipagem eletrÔnica arduino. **Universidade Federal Do Pará–UFPA Campus De Tucuruı**, p. 30, 2015.

WERNEC, Andressa; LAMEU, Valmir Rodrigues. APLICAÇÃO DE RELÉS MICROPROCESSADOS NA PROTEÇÃO DE MOTORES DE MÉDIA TENSÃO, 2018.

ZIMMERMANN, Sergio; FITZSIMMONS, Kevin. Tilapicultura intensiva. **Tópicos especiais em piscicultura de água doce tropical. São Paulo, SP: TecArt**, v. 1, p. 239–266, 2004.

APÊNDICE A – Código do Sistema

```
include<LiquidCrystal.h>
LiquidCrystal lcd(11,10,9,6,5,4);
const int sensorPin = A3;
int trigger = 12;
int echo = 13;
int led = 8;
int led2 = 7;
int motor = 3;
int comida= 10;
int botao= 2;
float temp = 0;
float duracao = 0;
float cm = 0;
bool tem_comida;
long int tempo_racao=0;
int estadobotao = 0;
int contador_dia=0;
float tempo_para_desconto=0;
long int tempo_racao_reset=0;
void setup() {
Serial.begin(9600);
pinMode(trigger, OUTPUT);
pinMode(botao,INPUT);
pinMode(echo, INPUT);
pinMode(led, OUTPUT);
pinMode(led2, OUTPUT);
pinMode(motor, OUTPUT);
pinMode(sensorPin, INPUT);
lcd.begin(16, 2);
}
void mostrar temp() {
if(comida==0) lcd.clear();
lcd.setCursor(1,0);
lcd.print("Nivel Cheio");
lcd.setCursor(3,1);
lcd.print("Temp:");
lcd.print(temp);
```

```
}
if(comida==1) {
lcd.clear();
lcd.setCursor(1,0);
lcd.print("Nivel Medio");
lcd.setCursor(3,1);
lcd.print("Temp:");
lcd.print(temp);
}
if(comida==2) {
lcd.clear();
lcd.setCursor(1,0);
lcd.print("Nivel Baixo");
lcd.setCursor(3,1);
lcd.print("Temp:");
lcd.print(temp);
}
void sensor() {
digitalWrite(trigger, LOW);
digitalWrite(trigger, HIGH);
digitalWrite(trigger, LOW);
duracao = pulseIn(echo, HIGH);
cm = duracao*0.034/2;
if(cm <=29.9cm>1) {
tem_comida=true;
Serial.print("de comida cheio!");
comida=0;
}
if (cm <=45cm>30) {
digitalWrite(led2, HIGH);
tem_comida=true;
comida=1;
}
else {
digitalWrite(led2, LOW);
}
if (cm<55cm>45) digitalWrite(led, HIGH);
tem_comida=true; comida=2;
else digitalWrite(led, LOW);
```

```
if (cm>55) {
digitalWrite(led, HIGH);
tem_comida=false; comida=2;
lcd.clear(); lcd.setCursor(1,0);
lcd.print("Nivel Critico");
}
}
void medir temperatura() {
temp = map(((analogRead(sensorPin) - 20) * 3.04), 0, 1023, -40, 125);
mostrar temp();
}
void alimentar() {
if(tem_comida==true) {
if (temp>29) {
Serial.print("agora: ");
Serial.print(temp);
mostrar temp();
digitalWrite(motor, HIGH);
delay(1200);
digitalWrite(motor, LOW);
Serial.print("de alimentacao de: ");
Serial.print(1200/1000);
Serial.print("segundos");
tempo racao=tempo racao+1200;
tempo_racao_reset=tempo_racao_reset+1200;
}
if (temp>=27temp<=29) {
Serial.print("agora: ");
Serial.print(temp);
mostrar temp();
digitalWrite(motor, HIGH);
delay(60000);
digitalWrite(motor, LOW);
Serial.print("de alimentacao de: ");
Serial.print(60000/1000);
Serial.print("segundos");
tempo racao=tempo racao+60000;
tempo_racao_reset=tempo_racao_reset+60000;
}
```

```
if( temp>=25temp<=26.9) {
Serial.print("agora: ");
Serial.print(temp);
mostrar temp();
digitalWrite(motor, HIGH);
delay(54000);
digitalWrite(motor, LOW);
Serial.print("de alimentacao de: ");
Serial.print(54000/1000);
Serial.print("segundos");
tempo racao=tempo racao+54000;
tempo_racao_reset=tempo_racao_reset+54000;
}
if (temp>=22temp<=24.9) {
Serial.print("agora: ");
Serial.print(temp);
mostrar temp();
digitalWrite(motor, HIGH);
delay(42000);
digitalWrite(motor, LOW);
Serial.print("de alimentacao de: ");
Serial.print(42000/1000);
Serial.print("segundos");
tempo racao=tempo racao+42000;
tempo_racao_reset=tempo racaø reset+42000;
}
if (temp>=20temp<=21.9) {
Serial.print("agora: ");
Serial.print(temp);
mostrar temp();
digitalWrite(motor, HIGH);
delay(24000);
digitalWrite(motor, LOW);
Serial.print("de alimentacao de: ");
Serial.print(24000/1000);
Serial.print("segundos");
tempo racao=tempo racao+24000;
tempo_racao_reset=tempo racaø reset+24000;
}
```

```
if (temp>=18temp<19.9) {
Serial.print("agora: ");
Serial.print(temp);
mostrar temp();
digitalWrite(motor, HIGH);
delay(6000);
digitalWrite(motor, LOW);
Serial.print("de alimentacao de: ");
Serial.print(6000/1000);
Serial.print("Segundos");
tempo racao=tempo racao+6000;
tempo_racao_reset=tempo_racao_reset+6000;
}
if (temp<18) {
Serial.print("agora: ");
Serial.print(temp);
mostrar temp();
digitalWrite(motor, HIGH); R
delay(1200);
digitalWrite(motor, LOW);
Serial.print("de alimentacao de: ");
Serial.print(1200/1000);
Serial.print("Segundos");
tempo racao=tempo racao+1200;
tempo_racao_reset=tempo_racao_reset+1200;
}
}
}
void timer de 8 horas() {
sensor();
Serial.print("Comeco da hibernacao ate a proxima alimentacao");
for (int i=0;i<5760;i++) {
delay(5000);
sensor();
estadobotao = digitalRead(botao); {
if(estadobotao == HIGH)
lcd.clear();
lcd.setCursor(1,0);
lcd.print(((tempo racao/1000)*(3.34)));
```

```
lcd.setCursor(7,0);
lcd.print("Gramas");
lcd.setCursor(1,1);
lcd.print("Consumidas");
}
}
medir temperatura();
Serial.print("Fim da hibernacao");
}
void timer de 8 horas descontado() {
int i;
sensor();
tempo_para_desconto=tempo_racao_reset;
tempo_para_desconto=tempo_para_desconto/5000;
Serial.print("Comeco da hibernacao ate a proxima alimentacao");
for (i=0;i<5760-(int)tempo_para_desconto;i++) {
Serial.print("primeira hibernacao do dia foram descontados: ");
Serial.print(tempo_para_desconto);
Serial.print("Delay's correspondentes a 5 segundos cada");
delay(5000);
sensor();
estadobotao = digitalRead(botao);
if(estadobotao == HIGH) {
lcd.clear();
lcd.setCursor(1,0);
lcd.print(((tempo racao/1000)*(3.34)));
lcd.setCursor(7,0);
lcd.print("Gramas");
lcd.setCursor(1,1);
lcd.print("Consumidas");
}
}
medir temperatura();
Serial.print("Fim da hibernacao");
tempo_racao_reset=0;
tempo_para_desconto=0;
}
void loop() {
Serial.print("Inicio do dia: agora sao 6 da manha");
```

```
medir temperatura();
sensor();
alimentar();
timer de 8 horas descontado ();
alimentar();
timer de 8 horas();
alimentar();
timer de 8 horas();
if(contador_dia==7) {
tempo racao=0;
contador_dia=0;
Serial.print("DA SEMANA");
}
contador_dia++;
}
```