UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ

EMÍLIO GABRIEL DALLASTELLA
ERIC KARL SCHMIDT

FEEDMYPET: ALIMENTADOR AUTOMÁTICO DE ANIMAIS DE ESTIMAÇÃO

CURITIBA
2021

EMÍLIO GABRIEL DALLASTELLA
ERIC KARL SCHMIDT

FEEDMYPET: ALIMENTADOR AUTOMÁTICO DE ANIMAIS DE ESTIMAÇÃO

Feedmypet: Automatic pet feeder

Trabalho de conclusão de curso de graduação apresentado como requisito parcial à obtenção do título de Bacharel em Engenharia Eletrônica do Departamento de Eletrônica da Universidade Tecnológica Federal do Paraná (UTFPR).

Orientador: Prof. Dr. Guilherme de Santi Peron.

CURITIBA
2021

EMÍLIO GABRIEL DALLASTELLA

ERIC KARL SCHMIDT

FEEDMYPET: ALIMENTADOR AUTOMÁTICO DE ANIMAIS DE ESTIMAÇÃO

Trabalho de Conclusão de Curso de Graduação apresentado como requisito para obtenção do título de Bacharel em Engenharia Eletrônica da Universidade Tecnológica Federal do Paraná (UTFPR).

Data de aprovação: 09/dezembro/2021

_____

Guilherme de Santi Peron
Doutorado
Universidade Tecnológica Federal do Paraná

_____

Luciano Scandelari
Doutorado
Universidade Tecnológica Federal do Paraná

_____

Luiz Fernando Copetti
Mestrado
Universidade Tecnológica Federal do Paraná

CURITIBA

2021

Este trabalho foi dedicado a todos os amigos, colegas e professores da Universidade Tecnológica Federal do Paraná, especialmente do curso de Engenharia Eletrônica, que nos deram as bases teórica e prática para a realização deste trabalho.

AGRADECIMENTOS

Agradecemos primeiramente ao professor Guilherme Peron por ter sanado tantas dúvidas e nos atendido sempre que necessitamos. Agradecemos aos familiares e amigos pelas palavras de incentivo, apoio e ajuda nos momentos de incerteza e grandes dificuldades. Agradecemos também a todas as pessoas que nos ajudaram direta ou indiretamente na execução, escrita e apresentação deste trabalho de conclusão de curso para que fosse possível a realização de nos tornarmos engenheiros.

Quando algo é importante o suficiente, você o faz mesmo que as probabilidades não estejam a seu favor. (MUSK, Elon, 2020).

RESUMO

Durante a pandemia de 2020 a população brasileira precisou ficar em casa por mais de um ano, o que causou um crescimento na incorporação de animais de estimação, ou *pets* nos lares brasileiros. Como os países estão cada vez mais abrindo suas fronteiras para viagens, surgiu uma necessidade de cuidar dos animais de estimação à distância. Para resolver este problema, a proposta do projeto foi desenvolver um alimentador automático para *pets*, com reservatórios de água e comida, controlado remotamente por um aplicativo móvel nativo. Através deste aplicativo móvel, o usuário cadastra uma conta e configura os horários e porções das refeições de seu *pet*. O desenvolvimento deste projeto foi dividido em diferentes partes, a mecânica, o hardware, o *firmware*, e o software. Inicialmente na parte mecânica foi projetado um modelo para impressão 3D, mas o aumento de 100% do preço do material inviabilizou sua produção. Como solução, o mesmo modelo foi criado em MDF (Placa de Fibra de Média Densidade, do inglês, *Medium Density Fiberboard*), um tipo de madeira. Já no *hardware*, uma PCB foi projetada e impressa para conectar diversos componentes e módulos. Nesta placa foram soldadas as seguintes partes: um módulo ADC responsável por ler duas células de carga de 5kg, um *driver* DRV8825 responsável por controlar um motor de passo, componentes responsáveis por acionar uma bomba d'água, e um Raspberry Pi Zero W, além de conectores para alimentar o circuito. Para a criação do aplicativo móvel foi escolhida a *framework* React Native, que torna possível o desenvolvimento de aplicativos nativos tanto para Android quanto para iOS. Durante o desenvolvimento, foi usada a ferramenta Expo para simular e posteriormente exportar o aplicativo para uso. Já no servidor, foi usada a *framework* Node.js Express para desenvolvimento, e o serviço Heroku para sua hospedagem. Um banco de dados não relacional, chamado MongoDB, foi criado e hospedado no serviço MongoDB Atlas. Para controlar os atuadores e ler os sensores no Raspberry Pi Zero W, foi desenvolvido um programa em Python, que foi agendado para execução a cada minuto em seu sistema operacional, Linux. Nos testes finais o produto funcionou conforme o esperado durante dois dias seguidos, portanto o resultado foi um sucesso em termos de funcionalidade. Levando em conta os custos de produção do protótipo e seus benefícios únicos em comparação aos outros produtos no mercado, seu preço seria competitivo. O projeto também foi um sucesso do ponto de vista de aprendizado por incorporar diferentes áreas e tecnologias atuais em um único projeto.

Palavras-chave: Alimentador. Animal de estimação. Aplicativo. IoT. Sistema Embarcado.

ABSTRACT

During the 2020 pandemic, the Brazilian population had to stay at home for more than a year, which caused a growth in the incorporation of pets in Brazilian homes. As countries are increasingly opening their borders for travel, there was a need to take care of pets remotely. To solve this problem, the project proposal was to develop an automatic pet feeder, with water and food reservoirs, remotely controlled by a native mobile application. Through this mobile application, the user registers an account and configures the times and portions of their pets meals. The development of this project was divided into different parts, mechanics, hardware, firmware, and software. Initially, in the mechanical part, a model for 3D printing was designed, but the 100% increase in the price of the material made its production unfeasible. As a solution, the same model was created in MDF (Medium Density Fiberboard), a type of wood. In the hardware, a PCB was designed and printed to connect different components and modules. The following parts were soldered on this board: an ADC module responsible for reading two 5kg load cells, a DRV8825 driver responsible for controlling a stepper motor, components responsible for activating a water pump, and a Raspberry Pi Zero W, as well as connectors for powering the circuit. For the creation of the mobile application, the React Native framework was chosen, which makes it possible to develop native applications for both Android and iOS. During development, the Expo tool was used to simulate and later export the application for use. On the server side, the Node.js Express framework was used for development, and the Heroku service was used for hosting it. A non-relational database, called MongoDB, was created and hosted on the MongoDB Atlas service. To control the actuators and read the sensors on the Raspberry Pi Zero W, a Python script was developed, which was scheduled to run every minute on its operating system, Linux. In the final tests the product worked as expected during two days in a row, therefore it was a success in terms of functionality. Taking into account the costs in the prototype production and its unique benefits in comparison to the other products in the market, its price would be competitive. The project was also a success from the learning point of view for incorporating different areas and current technologies in a single project.

Keywords: Feeder. Pet. App. IoT. Embedded System.

LISTA DE FIGURAS

LISTA DE ABREVIATURAS E SIGLAS

ABNT	Associação Brasileira de Normas Técnicas
DECOM	Departamento de Computação
ABS	Acrilonitrila butadieno estireno
CAD	Computer-aided Design
API	Application Programming Interface
CLI	Command Line Interface
URL	Uniform Resource Locator
URI	Uniform Resource Identifier
APK	Android application package
AWS	Amazon Web Services
MDF	Medium Density Fiberboard
PCB	Printed Circuit Board
STL	Standard Triangle Language
DC	Direct current
DC-DC	Direct current - Direct current
LC	Indutor Capacitor
SBC	Single Board Computer
GPIO	General Purpose Input/Output
LED	Light Emitting Diode
RAM	Random Access Memory
SO	Sistema Operacional
Mac OS	Macintosh Operating System
Ghz	Gigahertz
Mb	Megabyte

HDMI	High-Definition Multimedia Interface
mm	Milimetro
Wi-Fi	Wireless Fidelity
cm	Centímetro
A	Ampère
USB	Universal Serial Bus
GND	Ground
ADC	Analog to Digital Converter
kgf.cm	Quilograma-força por Centímetro
NPN	Negativo-Positivo-Negativo
Vcc	Tensão Corrente Contínua
SSID	Service Set Identifier
SSH	Secure SHell
kg	quilograma
g	grama
MicroSD	MicroSecureDigital

SUMÁRIO

1 INTRODUÇÃO

Desde 2013, o número de animais de estimação ou *pets* superou o número de crianças de até 12 anos nos lares brasileiros. Esse cenário por si só abre muitas portas para o mercado empreendedor no Brasil. No entanto, durante a pandemia de 2020, a população foi obrigada a ficar em casa por mais de um ano. Isso levou ao crescimento ainda maior do número de animais de estimação. Como recentemente outros países reabriram suas fronteiras para brasileiros, muitas pessoas querem voltar a viajar, o que gerou a necessidade de cuidar dos *pets* à distância. A proposta do projeto alia-se a esta situação atual. A ideia foi desenvolver um produto que permite alimentar os animais de estimação com comida e água de qualquer lugar do mundo, desde que possua conexão à internet no *smartphone*. O produto desenvolvido também oferece informações de retorno da quantidade de comida e água que resta nos recipientes.

1.1 AUMENTO DO MERCADO PET

Em plena pandemia, com vários setores entrando em crise e sofrendo com retração do mercado, o comércio de produtos e serviços *pet* expandiu, fechando 2020 com faturamento superior a R$40 bilhões (PETZ, 2021).

A explicativa desse fenômeno é que, devido ao distanciamento social e ao aumento do tempo de permanência em casa, as pessoas buscaram novas companhias para atenuar a solidão. Além disso, o setor *pet* foi considerado como essencial, o que manteve *pet shops* abertos e funcionando normalmente. Devido ao papel cada vez mais importante dos animais de estimação, seus donos estão dispostos a gastar uma quantidade maior de dinheiro com os mesmos, já que estão mais preocupados com sua saúde e bem estar (JULIANA AMéRICO, 2021).

De acordo com uma pesquisa feita pela *Euromonitor Internacional* que identificou as três principais áreas fundamentais para o desenvolvimento do mercado *pet*, nomeadamente diversificação de produtos e serviços, vendas *online*, e busca crescente por produtos de melhor qualidade. Este projeto se enquadra na terceira, que seria uma busca maior por produtos de melhor qualidade, do inglês *premium*, e que ofereçam uma qualidade de vida melhor tanto ao animal, quanto ao seu dono, conforme o estudo (EUROMONITOR, 2021).

1.2 EXPECTATIVA DE VIAGEM

Uma recente pesquisa da *Opinion Box* em conjunto com a Maxmilhas apontaram que 86% dos brasileiros já pensam em viajar e que 35% já pensam em viajar assim que estiverem vacinados contra a Covid-19. A pesquisa também mostrou que 43% dos entrevistados planejam fazer entre 3 e 6 viagens no próximo ano, mostrando a demanda reprimida gerada pela pandemia. O tipo de viagem mais citado pelos entrevistados, representando 59%, foi com a família (MAXMILHAS, 2021).

Segundo o presidente da Associação Brasileira das Operadoras de Turismo, as vendas de pacotes, passagens e diárias para destinos dentro do Brasil subiram 20% desde o mês de maio. Em julho, o Brasil era o segundo país com a maior restrição de ingresso em outros países, representado pelo número de viagens no primeiro trimestre 90,3% menor que no mesmo período de 2019. Por mais que as pessoas tivessem uma condição financeira favorável e a vontade de viajar, não podiam por restrições sanitárias, o que aumenta ainda mais a demanda reprimida (JOãO JOSé OLIVEIRA, 2021).

1.3 O QUE ACONTECE COM OS *PETS*?

Visto que o mercado *pet* aumentou muito, assim como a vontade de viajar, o que acontece com os *pets* enquanto seus donos viajam?

Gatos, por exemplo, se estressam muito em viagens e podem ter problemas de saúde decorrente desse estresse, de acordo com a Dra. Juliana Brondino, médica veterinária da Petz. Uma das opções mais adotadas por quem tem gato e gosta de fazer viagens curtas é de deixá-lo sozinho em casa. Essa opção funciona, já que gatos fazem suas necessidades fisiológicas na caixinha de areia e são seres que costumam seguir rotinas e costumam comer sempre no mesmo horário. Mesmo assim, alguns gatos não conseguem controlar seus impulsos alimentares e, se deixada, por exemplo, comida suficiente para os dois dias à disposição do bichinho, ele vai acabar comendo tudo de uma vez e passando mal (PETZ, 2021).

Para eliminar essa possibilidade, o produto foi projetado com o foco em disponibilizar a quantidade certa no momento certo, mantendo a rotina do animal em dia. O *FeedMyPet* é totalmente controlável por aplicativo celular, onde o usuário vai poder programar os horários de depósito de comida assim quanto a quantidade depositada por vez. O usuário também tem acesso também a quantidade de comida que está no pote, assim como o volume de água.

1.4 OBJETIVOS

Nesta seção é discutido o objetivo geral, assim como os objetivos específicos do projeto.

1.4.1 OBJETIVO GERAL

O objetivo geral do trabalho foi desenvolver um sistema alimentador de animais de estimação no qual o usuário utilize apenas seu *smartphone* e tenha o controle da quantidade de ração e do horário das refeições do seu *pet*. Dessa maneira, o usuário não precisa estar em casa para garantir que seu animal de estimação siga sua rotina alimentar. Além disso, o proprietário do alimentador tem o *feedback* sobre o volume de água e quantidade de comida disponíveis.

1.4.2 OBJETIVOS ESPECÍFICOS

- Desenvolver um aplicativo *mobile* nativo para Android e iOS que permite o usuário cadastrar uma conta, a quantidade de refeições no dia, quantidade de ração por refeição, e o horário de cada refeição;
- Desenvolver um *server* para conectar o aplicativo a um banco de dados para armazenar as informações cadastradas pelo usuário;
- Obter a quantidade de ração no pote através de uma célula de carga;
- Obter a quantidade de água que há no pote através de uma célula de carga;
- Enviar e receber dados entre o módulo e o banco de dados através de uma rede sem fio;
- Desenvolver um sistema para depósito de ração;
- Desenvolver um sistema para enchimento do recipiente com água;
- Desenvolver um recipiente em 3D que caibam todos os sistemas desenvolvidos;
- Desenvolver uma placa de circuito impresso com todos os módulos integrados;
- Testar cada sistema separadamente e analisar sua funcionalidade;
- Integrar todos os sistemas e analisar sua funcionalidade;
- Analisar a viabilidade econômica do projeto.

1.5 JUSTIFICATIVA

Apesar de o Brasil ser um país multicultural, com extremas diferenças em crenças, sotaques e gírias, algo que é comum de norte a sul do país é o apego e amor aos animais de estimação. Segundo o IBGE (ABINPET, 2020) a população de *pets* tem crescido a cada ano, o que acumulou em um crescimento de 10,7% de 2018 a 2019. A Figura 1 ilustra a distribuição da população de *pets* no Brasil em 2020.

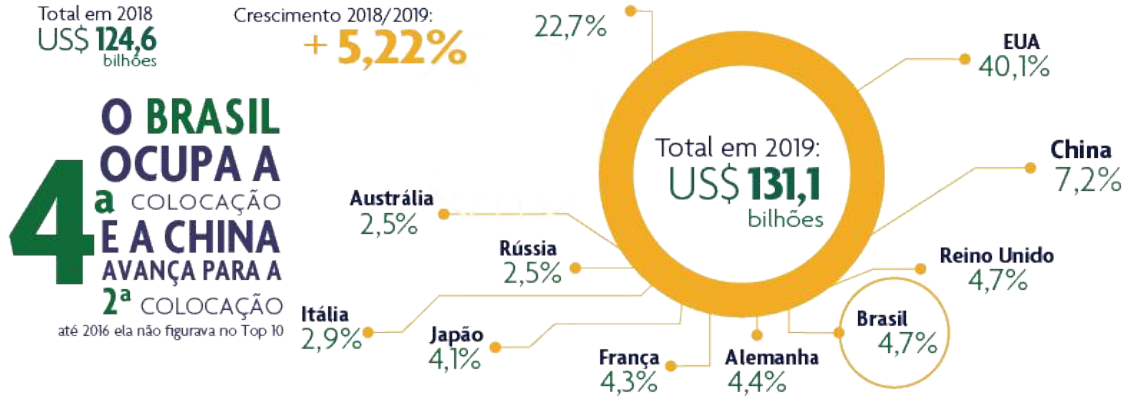
Figura 1 – Distribuição de espécies

Fonte: (ABINPET, 2020)

Não só a população tem aumentado, mas os gastos com os animais aumentaram 13,5% de novembro de 2019 para novembro de 2020, em plena pandemia. Em 2018 o Brasil tornou-se o

quarto maior mercado para *pets* no mundo, ficando atrás apenas dos EUA, China e Reino Unido, segundo o IBGE (ABINPET, 2020). Por isso, há um potencial enorme de exploração de mercado. A Figura 2 ilustra o faturamento de diferentes países no mercado mundial de *pets* em 2019.

Figura 2 – Mercado *Pet*



Fonte: (ABINPET, 2020)

Durante o ano de 2020 e ainda 2021 as fronteiras mundiais ficaram fechadas para turistas brasileiros. Em meados de junho de 2020 a União Europeia impôs critérios para entrada de estrangeiros em seu território, onde eram aceitas pessoas de países com até 16 contaminados pela Covid-19 por 100 mil habitantes. Essa restrição vetou a entrada de brasileiros, já que o país estava com taxa de 197 contaminados a cada 100 mil habitantes, muito acima dos critérios estabelecidos (JORNAL DA USP NO AR 1ª EDIçãO, 2021).

A partir da segunda quinzena de julho de 2021 alguns países europeus reabriram as fronteiras para alguns brasileiros, variando de acordo com a progressão da vacinação do indivíduo. Conforme o passar dos meses a quantidade de pessoas vacinadas aumentou, chegando no início de novembro de 2021 em 123.671.574 brasileiros vacinados com ambas as doses ou dose única. A Figura 3 ilustra a quantidade de doses aplicadas da vacina no Brasil até o dia 08 de novembro de 2021.

De acordo com Fernando Rocha, chefe de departamento de estatística do Banco Central, o gasto no Brasil com viagens internacionais no mês de setembro de 2021 foi 50% maior que do mesmo mês do ano anterior. Ainda em sua fala, afirma que a diminuição no número de casos e de mortes em todo mundo faz com que as economias reabram, recebendo turistas e que as restrições impostas anteriormente foram flexibilizadas devido às vacinações, o que gera interesse no turismo novamente. A abertura das fronteiras gera um impacto direto nos *pets*. Como seus donos vão viajar, os animais de estimação inevitavelmente vão ficar sozinhos em casa. Isso gera uma enorme dúvida e potencial: como alimentar seus *pets* remotamente e ter certeza que estejam se alimentando bem? (FERNANDA STRICKLAND, 2021)

Uma das respostas para essa pergunta é alimentadores controlados remotamente por

Figura 3 – Quantidade de vacinas

Fonte: (MINISTéRIO DA SAúDE, 2021)

aplicativo móvel.

1.6 ORGANIZAÇÃO DO TRABALHO

Na sequência é apresentado o *Referencial Teórico*, que consiste em fornecer a base teórica para o funcionamento dos materiais, periféricos e tecnologias envolvidos no trabalho. Então no capítulo chamado *Desenvolvimento*, é detalhado o desenvolvimento do projeto em diferentes frentes, a estrutura física, *hardware* e *software*. Por fim, o capítulo *Discussão dos Resultados* apresenta os resultados do desenvolvimento, e a *Conclusão* destacou os principais resultados e contribuições do trabalho para a área.

2 REFERENCIAL TEÓRICO

Este capítulo apresenta a base teórica dos principais materiais utilizados no decorrer do projeto, além de abordar todos os módulos de *hardware* empregados, características e funcionamento. Ele também apresenta as principais tecnologias envolvidas no projeto.

2.1 IMPRESSÃO 3D

A impressão 3D é uma tecnologia inovadora surgida em meados de 1984 que criou um mercado antes inexplorado. As décadas de 80 e 90 foram fundamentais para o desenvolvimento da tecnologia, até que em 2001 foi desenvolvida a primeira impressora 3D de mesa, base para as impressoras domésticas atuais. A impressão inicia-se a partir de um modelo digital geralmente no formato *.STL* ou *.OBJ* que o usuário cria com *softwares* específicos, como *SolidWorks*. O arquivo criado pelo programa é dividido em diversas camadas posteriormente impressas uma a uma. A matéria prima é usualmente depositada utilizando um injetor de matéria quente, onde o ABS (Acrilonitrila Butadieno Estireno), uma resina termoplástica composta por acrilonitrila, butadieno e estireno é a mais comercializada (TECNOBLOG, 2018).

2.2 *SOLIDWORKS*

O SolidWorks é um *software* CAD desenvolvido pela empresa *SolidWorks Corporation* cuja primeira versão foi lançada em novembro de 1995 e mais tarde comprado pela *Dassault Systems*. Funciona criando modelos tridimensionais a partir de formas geométricas básicas como retângulos, círculos e linhas (SOLIDWORKS, 2019).

2.3 *CÉLULA DE CARGA*

A célula de carga é um transdutor que converte força peso em corrente elétrica através de um extensômetro. Utiliza uma ponte de *Wheatstone* para fazer a medição mais precisa da carga (HBM, 2021).

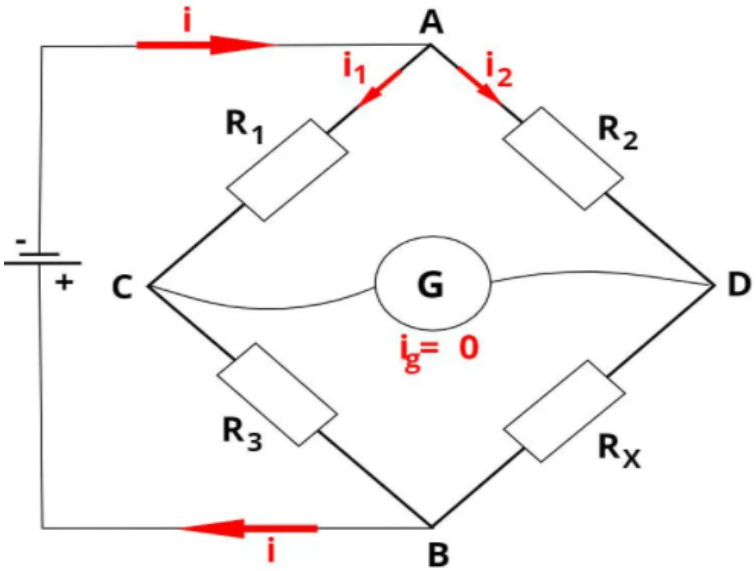
2.3.1 EXTENSÔMETRO

O extensômetro é um sensor formado por um fio resistivo que altera sua resistência de acordo com o alongamento do fio. É alongado quando colocado um objeto sobre o mesmo, já que vai deformar a superfície do fio. Dessa maneira, a corrente que circulará pelo fio será menor conforme maior o peso do objeto (ENSUS, 2016).

2.3.2 PONTE DE *WHEATSTONE*

A ponte de *wheatstone* desenvolvida por Samuel Hunter Christie, é utilizada para medir uma resistência desconhecida. É formada por um galvanômetro e quatro resistores. O primeiro, de resistência desconhecida que deseja-se medir. Dois com resistência conhecida e o quarto, de resistência variável. O circuito foi montado conforme a Figura 4.

Figura 4 – Ponte de Wheatstone



i_g – corrente no galvanômetro

R_X – resistência desconhecida

R_1, R_2, R_3 – resistências conhecidas

Fonte: (MUNDO EDUCAçãO, 2021)

O funcionamento da ponte de *wheatstone* pode ser descrito da seguinte maneira: quando a corrente i_g é igual a zero, a ponte está em equilíbrio através da Segunda Lei de Kirchoff, comumente conhecida como Lei das Malhas, ou seja, a resistência variável tem mesmo valor que a resistência desconhecida. Assim, variando a resistência R_x descobre-se o valor de R3 (MUNDO EDUCAçãO, 2021).

2.4 *MOTOR DE PASSO*

Inventado em 1936 por Marius Lavet, o motor de passo é um tipo de motor elétrico em que sua rotação é controlada por campos eletromagnéticos, a partir da corrente fornecida ao mesmo. Amplamente utilizado em impressoras 3D por conseguirem fornecer uma precisão muito alta. Foi utilizado um motor em que uma volta equivale a 200 passos, ou seja, 1,8 grau por movimentação. Dessa maneira, tem-se um controle preciso da quantidade de ração fornecida ao

animal de estimação.

O motor de passo possui diversos eletroímãs dentados dispostos em formato de uma engrenagem. Quando um eletroímã recebe corrente, gera um campo eletromagnético que atrai esses dentes e ambos ficam alinhados. No entanto, o dente fica levemente desalinhado com o segundo eletroímã, que assim que acionado, atrai este dente em um sentido de rotação, sucessivamente para girar o rotor (EL PROCUS, 2021).

2.5 *MOTOR DC*

O motor DC é nomeado desta forma pelo fato de utilizar corrente contínua como alimentação. Assim que a corrente percorre o condutor posicionado em um campo magnético, a força do campo atua sobre o rotor e gera torque. Esse torque gira o eixo do motor e assim, gera força mecânica. Um exemplo de motor DC é a bomba d'água. Este tipo de motor é acoplado a um rotor com diversas lâminas curvadas que giram rapidamente. O movimento circular gera inércia que atua do centro para fora do encapsulamento. Esta inércia gera duas zonas: uma de alta pressão, na periferia, e uma de baixa pressão no centro. A zona de baixa pressão faz a sucção do fluído através do bico de entrada. Este fluído, pela atuação das lâminas curvas que geram inércia, é projetado para as extremidades, onde há alta pressão. Por fim, a alta pressão expele o fluído através do bico de saída da bomba (HOME STEADY, 2021).

2.6 *CONVERSOR ANALÓGICO DIGITAL*

O conversor analógico é utilizado para converter valores de tensão físicos em código binário para que microcontroladores possam, por exemplo, interpretar sinais de sensores. Um sensor de pressão, por exemplo, que recebe cargas variadas tem diversos valores de tensão gerados pelo extensômetro. O conversor recebe esses valores e transforma-os num código binário, e, dependendo de sua resolução, terá uma precisão maior ou menor. Um conversor de 10 bits que tenha entrada de 0V a 5V, por exemplo, pode assumir valores de 0000000000 a 1111111111, onde o primeiro representa 0V e o último 5V. O valor 2,5V resultará em 512 em binário (SCIENCEDIRECT, 2021).

2.7 *MULTIPLEXADOR*

O multiplexador é um dispositivo utilizado para combinar a entrada de dois sinais diferentes, como duas células de carga em um único canal de informações, com taxa de transmissão mais alta. Evita, por exemplo, a utilização de um segundo conversor analógico-digital (SCIENCEDIRECT, 2015).

2.8 *CONVERSOR DC-DC*

Conversores DC-DC do tipo Step-Down, são utilizados como circuitos abaixadores de tensão. Eles recebem como entrada uma tensão mais alta, como por exemplo 12V e convertem para uma mais baixa, como 5V. Isto é alcançado ligando e desligando o transistor de saída rapidamente. Em seguida, utiliza um filtro LC para que a tensão de saída volte a ser DC (PROTOSUPPLIES, 2021).

2.9 *PLACA DE CIRCUITO IMPRESSO*

As placas de circuito impresso são as placas mais utilizadas em eletrônicos, geralmente feitas de fibra de vidro ou compostos de epóxi. Podem ter diversas camadas, cada uma com um circuito próprio, interligadas por canais. Cada camada é feita com o substrato envolto de um material condutor, onde cobre é o mais utilizado pelo custo benefício. Acima desse material vem a máscara de solda, que protege o material condutor de oxidação. São feitas para que os componentes sejam soldados diretamente nelas, no *layout* escolhido pelo engenheiro, e estejam interligados entre si para que não haja a necessidade de serem utilizados fios ou *jumpers*, tornando-as mais resilientes (PRINTED CIRCUITS, 2021).

2.10 *Single Board Computer*

O *Single Board Computer*, ou computador de placa única, possui todas as atribuições de um computador em apenas uma placa, como processador, memória, entradas e saídas e seu microprocessador. Por ter esse design compacto, diminui bastante os custos de desenvolvimento e produção. Por outro lado, são extremamente limitados por não existir a possibilidade de trocar o processador por outro mais recente, uma vez que ele está integrado com a placa e seus periféricos e estão atrelados ao projeto e *layout* do SBC (Single Board Computer) (TECHOPEDIA, 2017).

2.10.1 GPIO

General Purpose Input/Output, são entradas e saídas de uso geral. Responsável por fazer a transferência de dados digitais, em binário, como valores de sensores e botões, acionar LEDs e chaves (FAZEDORES, 2021).

2.11 *SISTEMA OPERACIONAL*

O sistema operacional, ou SO, é um conjunto de *softwares* que faz a interface entre o usuário e os comandos dados por ele com o *hardware* ou outros *softwares*. Administra os recursos do computador, por exemplo o uso da memória RAM e do processador. O SO é responsável por traduzir linguagem de máquina para algo abstrato para o usuário, que consiga entender e

controlar facilmente. Traduz também o inverso, todos os comandos do usuário para linguagem de máquina (TECNOBLOG, 2019).

2.11.1 LINUX

Criado por Linus Torvalds que teve sua primeira versão lançada em 1991, o Linux é um sistema operacional livre, distribuído sem custos pelos responsáveis. Possui milhares de desenvolvedores ao redor do mundo, uma vez que possui seu código aberto para que mais pessoas possam desenvolver aplicativos e soluções no mesmo (4LINUX, 2021).

2.11.2 ANDROID

Estimado em rodar em aproximadamente 73,3% dos dispositivos móveis ao redor do mundo, o Android é um sistema operacional de código aberto, ou seja, acessível gratuitamente para qualquer pessoa, até mesmo para comercializá-lo (ANDROID AUTHORITY, 2021).

2.11.3 iOS

Assim como Android, o iOS é um sistema operacional para dispositivos móveis, entretanto, de código fechado e comercializado apenas em dispositivos da marca Apple, presente em aproximadamente 26,4% dos aparelhos móveis no mundo. Baseado no sistema Mac OS, sistema operacional para os computadores da marca, foi projetado para ser de uso simples e com conectividade fácil entre produtos da marca (INVESTOPEDIA, 2021).

2.12 LINGUAGENS

No projeto desenvolvido, foram usadas algumas linguagens de programação em diferentes contextos, ambientes e sistemas. Nesta seção são discutidas as bases teóricas de cada linguagem.

2.12.1 Python

Python é uma linguagem de programação orientada a objetos, que também suporta outros paradigmas como programação procedural e funcional. Uma das características interessantes do Python é o uso de indentação no código na definição de funções e laços. (PYTHON SOFTWARE FOUNDATION, 2021b)

Por causa da sua versatilidade e grande quantidade de bibliotecas, ela é usada em diversas aplicações, de científicas e numéricas a desenvolvimento web. Atualmente, a linguagem está na versão 3.10 (PYTHON SOFTWARE FOUNDATION, 2021a).

2.12.2 JavaScript

JavaScript é uma linguagem de programação que permite a criação de conteúdos atualizados dinamicamente, animações, e outros elementos visuais em páginas web. Ela é muito usada na parte de interfaces gráficas em aplicações web, mas também é usada na parte de servidores. A linguagem é uma das mais usadas do mundo, em grande parte devido à vasta disponibilidade de APIs e bibliotecas, tornando o desenvolvimento de aplicações complexas mais simples (MOZILLA, 2021h).

2.12.3 TypeScript

TypeScript é um *superset* de JavaScript. Ou seja, TypeScript tem a mesma sintaxe e todas as funcionalidades de JavaScript, porém oferece uma camada adicional acima delas, o sistema de tipos. Este sistema permite o TypeScript verificar o programa desenvolvido por erros antes da sua execução, através dos tipos de valores no programa. Portanto, TypeScript auxilia na prevenção de erros de execução, e economiza tempo durante o desenvolvimento (MICROSOFT, 2021b).

2.12.4 HTML

HTML, ou *Hyper Text Markup Language*, é uma linguagem que consiste de vários elementos que são usados para envolver diferentes conteúdos em uma página para alterar as suas aparências ou adquirir propriedades diferentes (MOZILLA, 2021c).

2.12.5 CSS

CSS, ou *Cascading Style Sheets*, é uma linguagem que permite o controle da aparência de elementos HTML, como por exemplo cor, tamanho, fonte do texto e posicionamento na tela (MOZILLA, 2021g).

2.12.6 JSX

O JSX, ou JavaScript XML, é uma extensão de sintaxe do JavaScript que produz elementos React. O JSX permite a escrita direta de HTML no código JavaScript (FACEBOOK INC., 2021c).

2.13 *FRONTEND*

Em um aplicativo *mobile*, a parte da interface gráfica, com a qual o usuário interage, é chamada de *frontend*. No desenvolvimento *frontend*, uma combinação de linguagens, como HTML, CSS e JavaScript, é usada para a criação de páginas, elementos interativos como botões, animações e menus (CONCEPTA TECHNOLOGIES LLC, 2019).

2.13.1 Aplicativo *mobile*

Aplicativos, ou *apps mobile*, são aplicações de *software* que são executadas em dispositivos móveis, como *smartphones* e *tablets*. Geralmente aplicativos *mobile* possuem uma interface gráfica para a interação do usuário, e utilizam a internet para receber e transmitir dados para recursos computacionais remotos, como servidores e bancos de dados (AMAZON WEB SERVICES, 2021).

2.13.2 React Native

Criada como um projeto *open-source* pela empresa Facebook, React é uma biblioteca JavaScript usada na construção de interfaces gráficas. Uma das suas grandes vantagens é a possibilidade da criação de componentes, que são elementos reutilizáveis entre páginas, como por exemplos botões (FACEBOOK INC., 2021b).

React Native por sua vez é uma *framework* baseada em React, que permite o desenvolvimento de aplicações multiplataforma com apenas um código base JavaScript. Ou seja, o React Native compila o mesmo código JavaScript em elementos nativos para Android e iOS (TANIA RASCIA, 2018).

2.14 Cloud

Cloud computing, ou computação na nuvem, é o acesso via internet de recursos computacionais, como aplicativos, *servers* e armazenamento de dados (IBM, 2020).

2.15 *BACKEND*

No geral, o termo *backend* é usado para a parte do *software* que os usuários não veem. Como por exemplo as regras de acesso de transferência de dados em um *server*, ou o esquema de armazenamento de dados em um banco de dados.

2.15.1 *Server*

De um modo geral, o servidor, do inglês *server*, é um *software* que controla como usuários de uma certa aplicação recebem e enviam informações. Por exemplo, o *server* pode conter as regras de acesso e envio de informações de páginas *web* através de rotas (MOZILLA, 2021f).

2.15.2 Node.js

Node.js é um ambiente de tempo de execução, do inglês *runtime environment*, assíncrono e baseado em eventos que permite o uso de JavaScript em um *server*. Uma das grandes vantagens do Node.js é a sua escalabilidade devido a sua natureza não bloqueante (OPENJS FOUNDATION, 2021a).

2.15.3 Express

Express é uma *framework* Node.js, que permite a implementação de diferentes mecanismos, como utilizar diferentes métodos de requisições HTTP como *GET* e *POST* para diferentes rotas, e outras operações (MOZILLA, 2021b).

2.15.4 *Middleware*

Dentro do contexto do Node.js,*middleware* é um conjunto de funções que têm acesso ao objeto de requisição e resposta. Estas funções podem executar código, modificar os objetos de requisição e resposta, e terminar o ciclo de requisições e respostas, assim como chamar outra função *middleware* (OPENJS FOUNDATION, 2017).

2.15.5 *Hash*

Uma função *hash* recebe como entrada uma sequência de dados e gera um saída diferente que representa esta mesma sequência. Ela geralmente é usada para armazenar senhas em bancos de dados, sem que a senha original seja armazenada. Assim, caso haja o banco de dados seja comprometido, a senha do usuário ainda estaria segura (BINANCE ACADEMY, 2019).

2.15.6 Heroku

Heroku é uma plataforma em nuvem que permite a implementação e gestão de aplicativos. Uma de suas aplicações mais usadas é a hospedagem de *servers* (HEROKU, 2021b).

2.16 BANCO DE DADOS

Bancos de dados são sistemas que armazenam dados de forma organizada, tanto localmente, quanto na nuvem (MOZILLA, 2021a).

Há diferentes paradigmas de bancos de dados, porém os mais usados são o relacional e o orientado a documentos. A vantagem do uso de bancos de dados não relacionais, como orientados a documentos, é a flexibilidade na estrutura de organização dos dados, implementação mais simples, o que resulta em um tempo de desenvolvimento reduzido, e maior facilidade de escala horizontal, que é escala através de uma maior quantidade de *servers*, ao invés de *servers* maiores. Alguns exemplos de bancos de dados orientados a documentos são MongoDB, Amazon DynamoDB, e Firestore (MONGODB, INC, 2021c).

2.16.1 JSON

JSON, ou JavaScript *Object Notation*, é um formato de dados leve, de simples interpretação, e independente de linguagens (JSON, 2021).

2.16.2 MongoDB

É um banco de dados gratuito orientado a documentos, que armazena dados em documentos JSON, cuja estrutura é flexível. A sua implementação é simples, e pode ser escalado horizontalmente. O MongoDB armazena os documentos em coleções, e um banco de dados armazena uma ou mais coleções de documentos (MONGODB, INC, 2021e).

2.17 PROTOCOLO DE COMUNICAÇÃO

Protocolos de comunicação são um conjunto de regras que definem como sistemas enviam e recebem dados entre si (MOZILLA, 2021e).

2.17.1 HTTP

HTTP, ou *Hypertext Transfer Protocol*, é um protocolo de comunicação usado para buscar recursos como documentos HTML. O HTTP é um protocolo cliente-*server*, no qual o cliente envia uma requisição, e o *server* envia uma resposta de volta após recebê-la. É importante observar que neste protocolo o cliente sempre inicia a comunicação com o *server*, e não ao contrário. Cada recurso, que é buscado por uma requisição HTTP, é identificado por um URI, *Uniform Resource Identifier*. Um exemplo de URI são os endereços *web* usados para acessar sítios no navegador (MOZILLA, 2021d).

2.17.2 TCP/IP

TCP/IP, ou *Transmission Control Protocol/Internet Protocol*, é um conjunto de regras que permite computadores se comunicarem em uma rede. IP é um protocolo que obtém o endereço para onde os dados são transmitidos. TCP é um protocolo responsável pela transmissão de dados. No uso de TCP/IP, cada mensagem é dividida em pacotes que são reconstruídos após a transmissão (AVAST, 2021).

2.17.3 WebSocket

Ao contrário do protocolo HTTP, o WebSocket é uma conexão persistente entre um cliente e um *server*. Ele permite uma comunicação *full-duplex*, que inicia com um *handshake* HTTP, e cria uma conexão TCP/IP para a comunicação até o encerramento do WebSocket (ABLY, 2021).

3 DESENVOLVIMENTO

Neste capítulo é detalhada a definição do *hardware* e tecnologias envolvidas no projeto, e como cada um desses elementos foram utilizados no desenvolvimento.

3.1 PROJETO MECÂNICO

Para se realizar a alimentação do *pet* com comida e água, o circuito precisaria ser capaz de prover na medida certa, nos horários certos, com confiabilidade e repetibilidade. Além disso, foi necessário um recipiente para armazenar os perecíveis sem que estraguem e que o animal não tenha acesso a comida que está guardada.

3.1.1 Raspberry Pi Zero W

O Raspberry Pi Zero W é um microcomputador de placa única, com processador *single-core* de 1Ghz, 512Mb de memória RAM, porta mini-HDMI e porta GPIO de 40 pinos, de baixo custo, com integração à internet sem fio de fábrica, de fácil desenvolvimento e tudo isso num tamanho reduzido (65mm de comprimento por 30mm de largura). Ele foi escolhido para o projeto pela facilidade de aplicação, ampla disponibilidade de documentação, recursos e conectividade Wi-Fi sem uso de módulos externos. Como no projeto o Wi-Fi é utilizado constantemente, o consumo de corrente é em média 0,17A.

O Raspberry Pi Zero W pode ser alimentado de duas maneiras: diretamente por sua porta Micro-USB (descrita como *Micro Power Slot*) ou pelas portas 2 ou 4 com +5V e o *Ground* em qualquer uma das portas assinaladas como GND, conforme seu *datasheet* apresentado na Figura 5 (RASPBERRY PI, 2021).

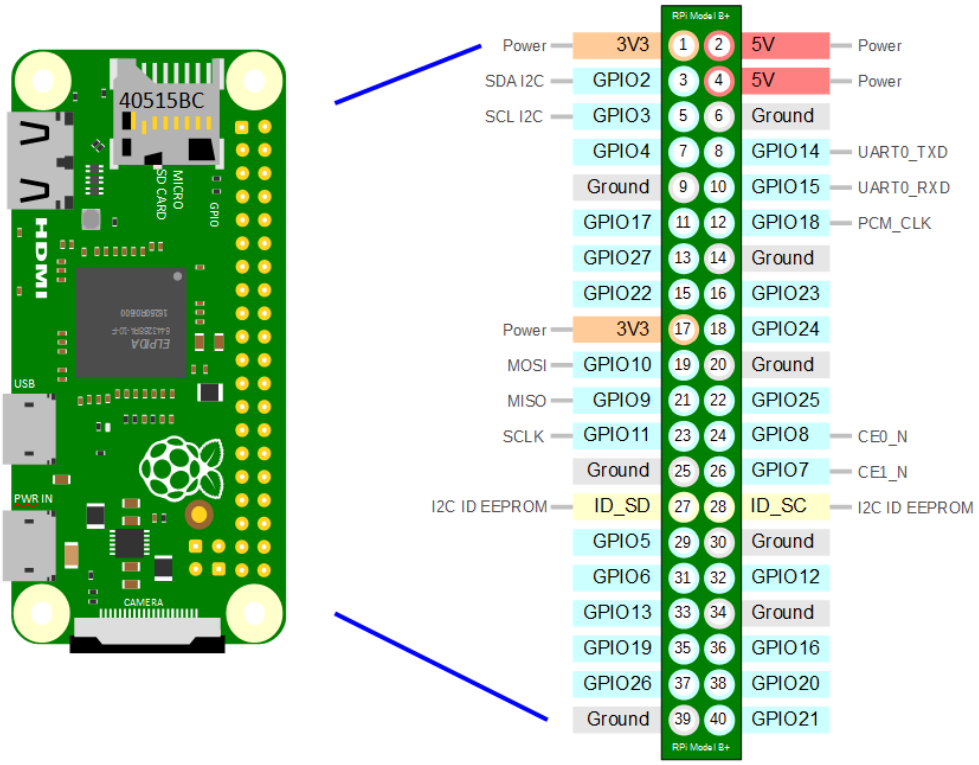
As seguintes saídas GPIO foram utilizadas:

- Pino 32 - GPIO 12: Chaveamento da bomba d'água;
- Pino 36 - GPIO 16: Controle de passos do motor;
- Pino 38 - GPIO 20: Controle de direção da rotação do motor;
- Pino 40 - GPIO 21: Controle do *clock* do HX711;
- Pino 37 - GPIO 26: Transferência de dados entre sensor de carga e Raspberry Pi.

Além delas, é preciso conectar o GND do Raspberry Pi com o GND da fonte de 12V, do conversor DC-DC, com o pino emissor do transistor TIP120 e do ADC HX71. Caso não sejam conectados, nenhum dos módulos funcionaria corretamente por não perceber a diferença de tensão entre os pinos GPIO e GND do Raspberry Pi. Dito isso, as conexões são:

- Pino 6 - GND: Conectado ao GND do módulo ADC HX711;

Figura 5 – *Datasheet* Raspberry Pi



Fonte: (RASPBERRY PI, 2021)

- Pino 9 - GND: Conectado ao emissor do TIP120 para realizar o chaveamento da bomba d'água;
- Pino 25 - GND: Conectado aos pinos 9 e 15, ambos GND, do *driver* DRV8825;
- Pino 39 - GND: Conectado a saída -Vcc do conversor DC-DC.
- Pino 2 - POWER: Conectado a saída +Vcc do conversor DC-DC.

Com as conexões feitas, todos módulos podem ser controlados e lidos diretamente pelo Raspberry Pi. Todo o circuito é alimentado pela fonte de 12V, sendo uma parte alimentada após a conversão para 5V pelo conversor DC-DC.

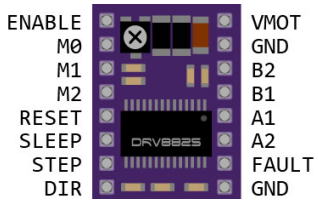
3.1.2 Motor de passo Nema 17 - WS17-0035-04-4

O motor de passo Nema 17 é um motor com torque 3,5kgf.cm, o suficiente para girar o mecanismo de abertura do reservatório para que a comida seja depositada no pote de alimento do animal. A corrente de fase é 0,4A. Como possui duas fases e é bipolar, a corrente é dobrada, para 0,8A. Sua tensão por fase é de 8V/fase. Diferentemente da corrente, não é necessário alimentá-lo com a soma das fases, ou seja, 8V em sua entrada são suficientes. O motor possui 4 fios de entrada, sendo dois para cada bobina. O *datasheet* do motor indica qual a cor dos pares, para que seja possível o controle do mesmo, uma vez que se forem conectados os fios errados o motor não funciona (WOTIOM, 2021).

3.1.3 Driver DRV8825

Para controlar o motor, foi utilizado o *driver* DRV8825, fácil de ser encontrado no mercado, de baixo custo, com especificações para controlar correntes de até 3A e que suporta altas temperaturas, ou seja, bastante resiliente. Seu diagrama foi apresentado na Figura 6 (ALL DATASHEET, 2021a).

Figura 6 – Pinagem DRV8825



Fonte: (ALL DATASHEET, 2021a)

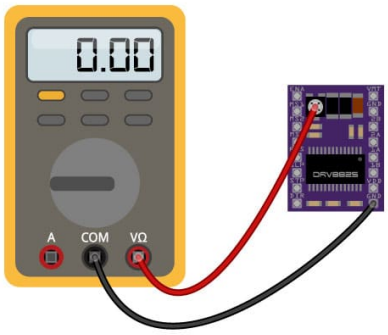
Os pinos do *driver* são:

- Pino 1 - ENABLE: Conectado diretamente a alimentação da placa, tem como função ligar o controle do motor;
- Pino 2 - VMOT: Entrada da alimentação do motor, nesse caso, conectado ao +12V da fonte;
- Pinos 3, 5 e 7 - Tem como função selecionar o tipo de passo, se é completo ou fracionado. Todos ficam sem alimentação, dessa maneira, cada *step* é completo;
- Pino 4 - GND: Entrada -Vcc do motor, conectado no GND da fonte;
- Pino 6 e 8 - Controle da primeira bobina: Conectados nos fios vermelho e verde do motor, referentes à bobina 1. A ordem desses dois não impacta no funcionamento do circuito;
- Pino 9 - RESET - Acionado com 0V, desabilita a ponte H de saída. Conectado a tensão de 5V do conversor DC-DC para não desabilitar o controle do motor;
- Pinos 10 e 12 - Controle da segunda bobina: Conectados nos fios amarelo e azul do motor, referentes a segunda bobina. A ordem desses dois não impacta no funcionamento do circuito;
- Pino 11 - SLEEP: Responsável por deixar o *driver* num estado dormente de baixo consumo de energia. Acionado com 0V, conectado ao 5V do conversor DC-DC.
- Pino 13 - STEP - Faz com que o motor de um passo, acionado por borda de subida.
- Pino 14 - FAULT: Acionado quando há sobrecorrente ou temperatura elevada. Não há conexão para o mesmo;
- Pino 15 - DIR: Responsável pela seleção da direção do motor, se é horário (0V) ou anti-horário com 5V;
- Pino 16 - GND: *Ground* que será conectado ao Raspberry Pi, para que haja diferença de potencial entre os pinos controlados por ele e o *driver*.

Para que a quantidade de corrente fornecida seja apenas o suficiente para o motor, foi ajustada a limitação de corrente no *driver* DRV8825 pelo próprio potenciômetro embutido. Para

realizar essa calibração, conectou-se apenas os pinos 2, 4, 9, 11 e 16, conforme citado acima. Com o auxilio de um multímetro, foi verificada a tensão entre o potenciômetro e o pino 16 conforme a Figura 7.

Figura 7 – Medição DRV8825



Fonte: (MAKERS GUIDE, 2021)

A equação para definir a corrente é dada por

$$V_{\text{ref}} = \frac{C_L}{2}, \tag{1}$$

onde C_L é a corrente máxima e V_{ref} é a tensão lida pelo multímetro. Como a corrente máxima no motor é 0,8A então, pela equação, a tensão entre o potenciômetro e o GND deve ser de 0,4V. Para reduzir o valor mostrado, o potenciômetro foi rotacionado em aproximadamente um quarto de volta no sentido horário (MAKERS GUIDE, 2021).

Após o *driver* ser configurado, foi necessário implementar uma solução no *firmware* do Raspberry Pi para o controle do motor. Os detalhes desta implementação são discutidos na subseção 3.6.2.3 deste documento.

3.1.4 Bomba d'água RS385

A mini Bomba de Água RS385 foi criada especialmente para projetos prototipados em plataformas microcontroladas, como o Raspberry Pi. Alimentada por uma tensão de 12V, possui vazão de no mínimo 90L por hora e máximo de 120L por hora. Tem tamanho de 90mm de comprimento por 40mm de largura e 35mm de espessura, relativamente pequena. Opera com 0,5A de corrente, e a corrente de carga vazio é de 0,18A. Conforme o fabricante, a vida útil de até 2500 horas. Possui dois bicos, um de entrada e saída com diâmetro de 1,6mm cada (MERCADO LIVRE, 2021).

Para o controle da fonte, foi implementada uma chave com transistor controlada pelo Raspberry Pi. A implementação no Raspberry Pi foi bastante simples, sendo necessário apenas acionar a saída GPIO e configurar seu parâmetro como ALTO durante um certo tempo utilizando *delay* de tempo. O período de tempo necessário para abastecer com água o pote do animal de estimação foi determinado através de testes. Nos testes o volume de água no pote foi colocado

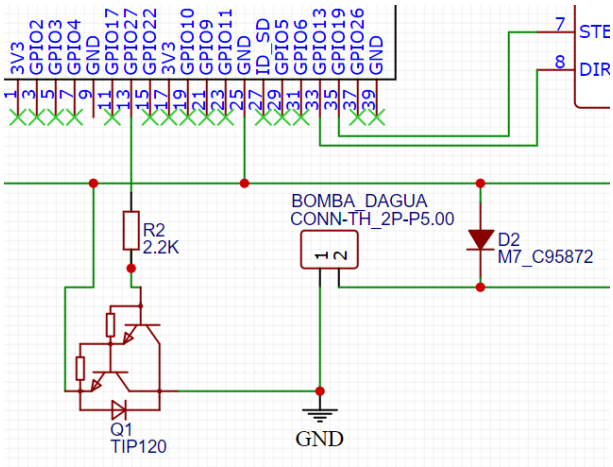
no volume mínimo de água, 55 mL. Então, foi acionada a bomba d'água até que o nível máximo da tigela fosse atingido, 185 mL. Este tempo foi cronometrado e fixado no *firmware* para garantir que a bomba d'água encha a tigela sem ultrapassar o limite do recipiente.

3.1.5 Chaveamento por Transistor

Para realizar o chaveamento, foi utilizado o transistor NPN TIP120. O transistor NPN, quando não há corrente na base, ou seja, na que foi conectada ao Raspberry Pi funciona como uma chave aberta, onde a tensão no coletor é a mesma que entre o coletor e o emissor, funcionando como sistema aberto. Para o chaveamento, quando há circulação de corrente na base, a tensão entre coletor e emissor foi dada por aproximadamente 0,2V, enquanto entre a base e o emissor foi 0,7V. Com essas diferenças de tensão, há circulação de corrente entre emissor e base, fazendo com que a bomba d'água seja ativada (ALL DATASHEET, 2021c).

O circuito utilizado para o chaveamento é ilustrado na Figura 8.

Figura 8 – Circuito Chaveador



Fonte: Autoria Própria

- +12Vcc da fonte conectado diretamente em uma das entradas da bomba d'água;
- Segunda conexão da bomba d'água conectada no coletor do transistor;
- Base do transistor conectado a um resistor de 2,2k ohms e este conectado ao pino 32 do Raspberry Pi;
- Emissor do transistor conectado ao GND da fonte e do Raspberry Pi;
- Diodo 1N4007 conectado do -Vcc para +Vcc da fonte.

3.1.6 Célula de carga

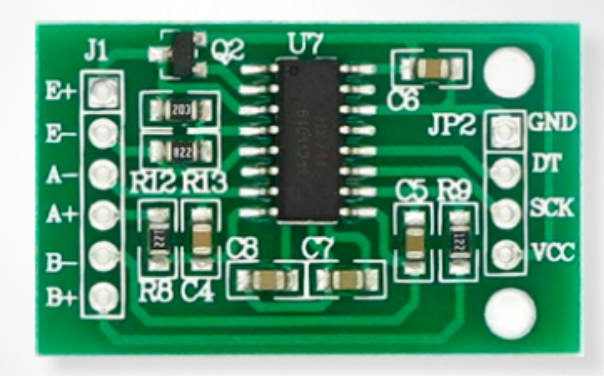
A célula de carga é composta por um *strain gauge* e uma ponte de *Wheatstone*. A célula de carga disponibiliza valores de tensão quando uma massa é depositada sobre a mesma, que se deforma. Alimentada por 5Vcc, tem consumo de aproximadamente 170 μA de corrente cada. Pelo fato da deformação, foi necessária uma fixação da célula na estrutura do alimentador,

através de dois cortes quadrados nas laterais centrais do mesmo. Os cortes permitiram que ela ficasse fixa e servisse de sustentação para os potes de água e comida, assim, consegue medir com precisão de $\pm 0{,}05\%$ a quantidade de comida e água depositada. Seus quatro fios foram conectados ao módulo HX711 sendo que dois foram utilizados para alimentação e dois para transferência de dados. O detalhamento da conexão é feito na seção 3.1.7.

3.1.7 HX711

O HX711 é um conversor ADC de 24 bits de resolução com alimentação de 5Vcc e consumo de corrente de 160 μA. Possui dois canais, ou seja, pode ser ligado em dois sensores simultaneamente. Possui 10 pinos, sendo eles apresentados na Figura 9.

Figura 9 – Módulo HX711



Fonte: (MóDULO HX711, 2021)

- Pino 1 - E+: Pino de alimentação positivo das células de carga;
- Pino 2 - E-: Pino de alimentação negativo das células de carga;
- Pino 3 - A-: Sinal negativo do primeiro canal;
- Pino 4 - A+: Sinal positivo do primeiro canal;
- Pino 5 - B-: Sinal negativo do segundo canal;
- Pino 6 - B+: Sinal positivo do segundo canal;
- Pino 7 - GND: Alimentação negativa da placa;
- Pino 8 - DT ou DOUT: Saída de dados para Raspberry Pi;
- Pino 9 - SCK: Entrada de *clock* serial;
- Pino 10 - Vcc: Alimentação positiva da placa.

Quando o pino DT está com nível alto, significa que o módulo não está preparado para enviar dados. Quando vai para nível baixo, indica que os dados estão prontos para serem transferidos. Aplicando de pulsos positivos de *clock* no pino 9, os dados são transferidos do pino 8. No último pulso, a saída DT se altera novamente para nível alto e se reinicia o ciclo (ALL DATASHEET, 2021b).

A configuração das células de carga com o HX711 foi dada por:

- Ambos fios vermelhos curto-circuitados e conectados no pino 1;

- Ambos fios pretos curto-circuitados e conectados no pino 2;
- Fio branco da primeira célula no pino 3;
- Fio verde da primeira célula no pino 4;
- Fio branco da segunda célula no pino 5;
- Fio verde da segunda célula no pino 6.
- Pino 7 conectado no GND do Raspberry Pi
- Pino 8 no pino 18 do Raspberry Pi;
- Pino 9 no pino 16 do Raspberry Pi;
- Pino 10 na saída positiva do conversor DC-DC.

3.1.8 Conversor DC-DC

Para obter 5Vcc no circuito, foi utilizado um regulador de tensão DC-DC do tipo *Step-Down*, para reduzir a tensão da fonte de 12Vcc para 5Vcc. Foi utilizado o Módulo Regulador de Tensão LM2596, adquirido na loja Baú da Eletrônica. Este módulo tem saída de tensão ajustável, de 1,5V a 35V e tendo entrada de 3,2V a 40V. Pelas suas especificações, a corrente nominal é de 2A e a corrente máxima de 3A, utilizando um dissipador de calor. Pela saída ser regulável, foi necessário realizar o ajuste para que a mesma estivesse com 5V de diferença de potencial. O procedimento foi simples, com os pinos de entrada voltados para a esquerda, foi necessário ligá-los a fonte e colocar um multímetro em paralelo com a saída, conforme Figura 10. Então, o potenciômetro foi rotacionado no sentido anti-horário até que o visor do multímetro marcasse 5V (BAú DA ELETRôNICA, 2021).

3.1.9 Esquemático

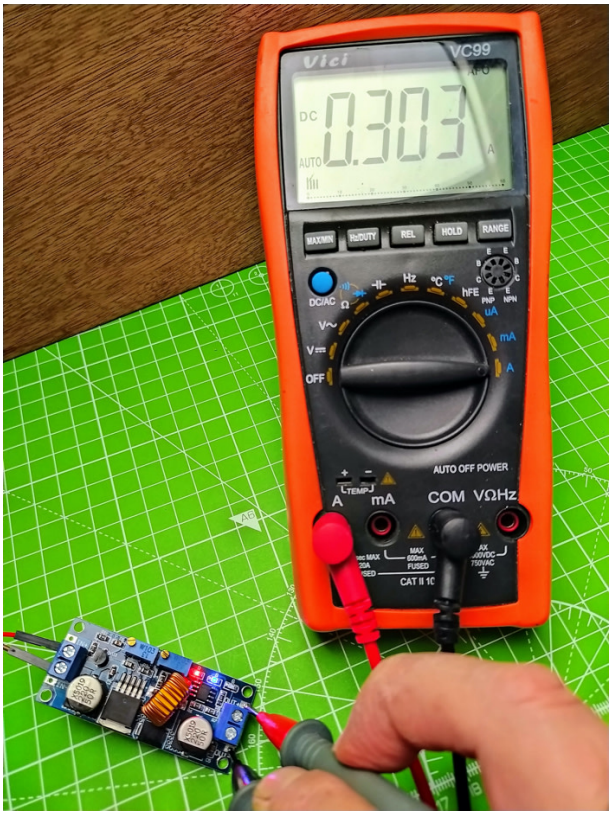
O esquemático da placa foi realizado no software *EasyEDA* conforme explicado em cada módulo acima. Além do que foi citado, foi inserido um diodo de roda-livre entre os polos da fonte. Porque após desligar o motor, o campo magnético que estava sendo criado gerava uma alta tensão inversa suficiente para queimar transistores ou outros componentes do circuito. O diodo invertido, então, curto-circuita a corrente gerada no desligamento, não permitindo sua circulação (EASYEDA, 2021).

No esquemático constam todos os módulos utilizados no projeto:

- Raspberry Pi Zero W;
- HX711;
- Conector para fonte;
- Conector para Bomba d'água;
- Driver do motor DRV8825.

Os *footprints* de cada objeto foram inseridos para que coubessem na placa. Para a soldagem, utilizou-se pinos macho para PCB. A Figura 11 apresenta o esquemático resultante.

Figura 10 – Ajustando Conversor



Fonte: (BAú DA ELETRôNICA, 2021)

3.1.10 PCB

Após realizar o esquemático, foi necessário transformá-lo numa placa que pudesse ser impressa. Para isso, foi utilizado o mesmo software, *EasyEDA*. O esquemático foi convertido internamente para PCB, que criou o tamanho da placa com os componentes e conexões fora da mesma, como pode ser visto na Figura 12.

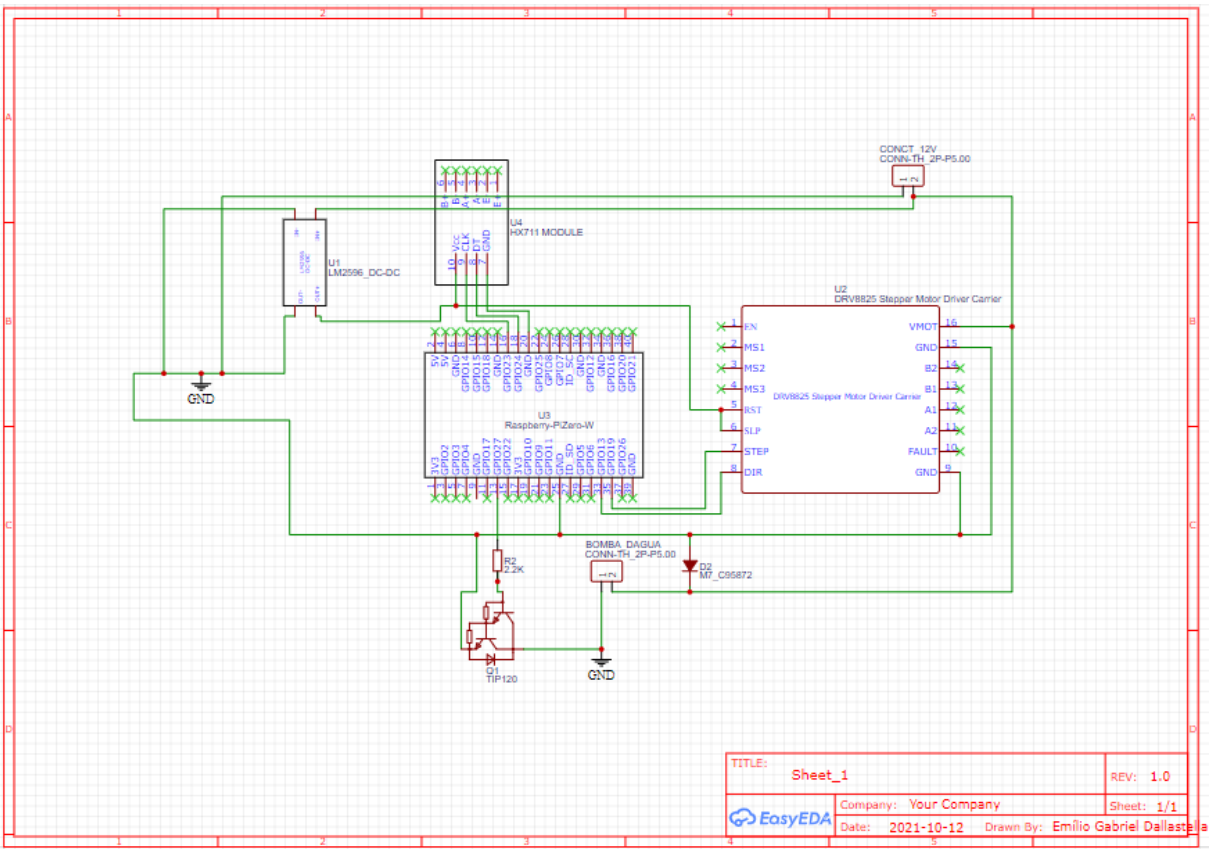
Então, foram colocados todos os componentes dentro dos limites da placa, retângulo delimitado pela cor roxa e alinhados da melhor maneira possível, que resultou na Figura 13. A modelagem em 3D da placa resultou na Figura 14.

A Figura 15 apresenta a PCB com todos os componentes soldados.

3.2 MODELO MECÂNICO 3D

O projeto do modelo 3D foi necessário para que fosse realizada a alocação do circuito com *hardware* integrado, como o motor e a bomba d'água, além de haver espaço para armazenamento da comida e da água para o animal de estimação. Deveria ser leve e de fácil movimentação, além de impossibilitar o acesso ao animal à ração e água do armazenamento. O projeto foi feito pensando na impressão 3D do molde, utilizando um polímero plástico resistente e maleável. A produção seria feita por uma loja especializada em impressões 3D.

Figura 11 – Esquemático do circuito



Fonte: Autoria própria

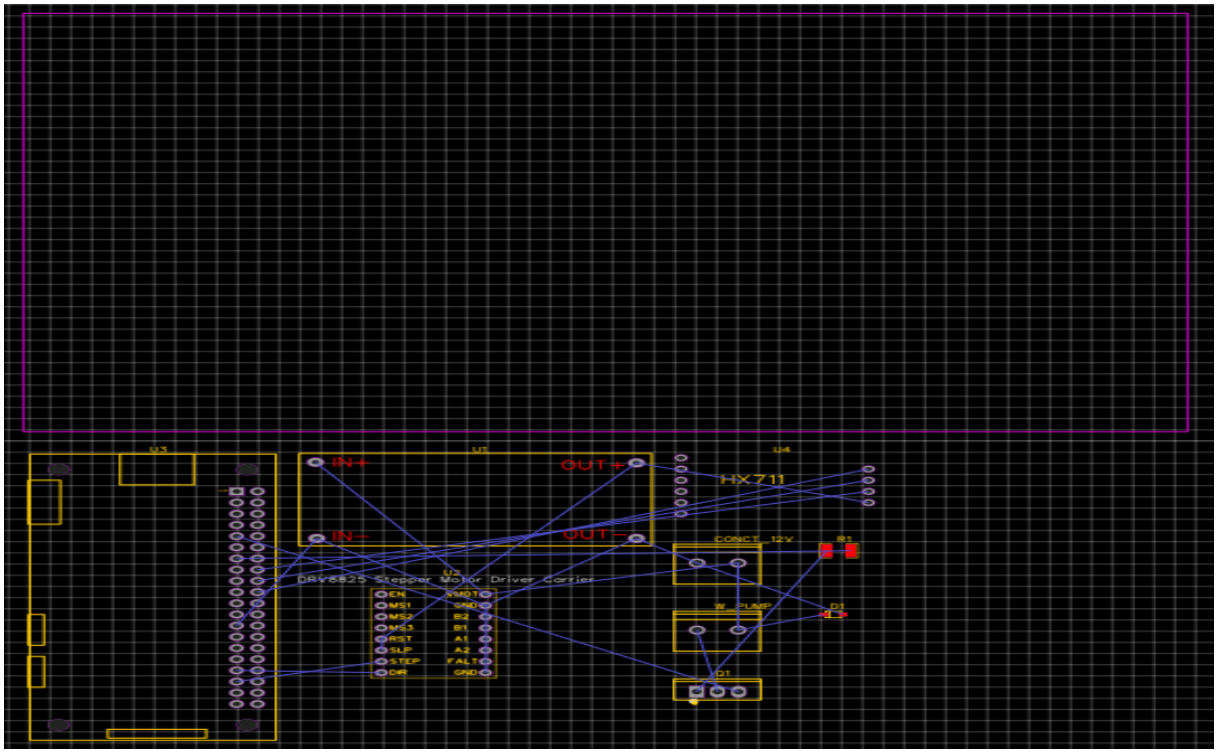
3.2.1 Definição do material

O material foi definido como ABS. Escolhido por ser bastante abundante no mercado, ter preço relativamente baixo, ser flexível e resistente, além de dar bom acabamento ao projeto. Entretanto, a alta demanda de resinas plásticas gerada pela pandemia, um inverno rigoroso no maior produtor de resinas plásticas do mundo, aumento do dólar e enorme demanda no mercado asiático pela matéria prima contribuíram para um aumento de 100% no preço do ABS. Aumentou também a demanda por impressoras 3D, para impressão de *face-shields*, por exemplo. Esse aumento gerou inviabilidade financeira do produto, que obrigou a alteração de material para MDF.

3.2.2 Desenvolvimento no SOLIDWORKS

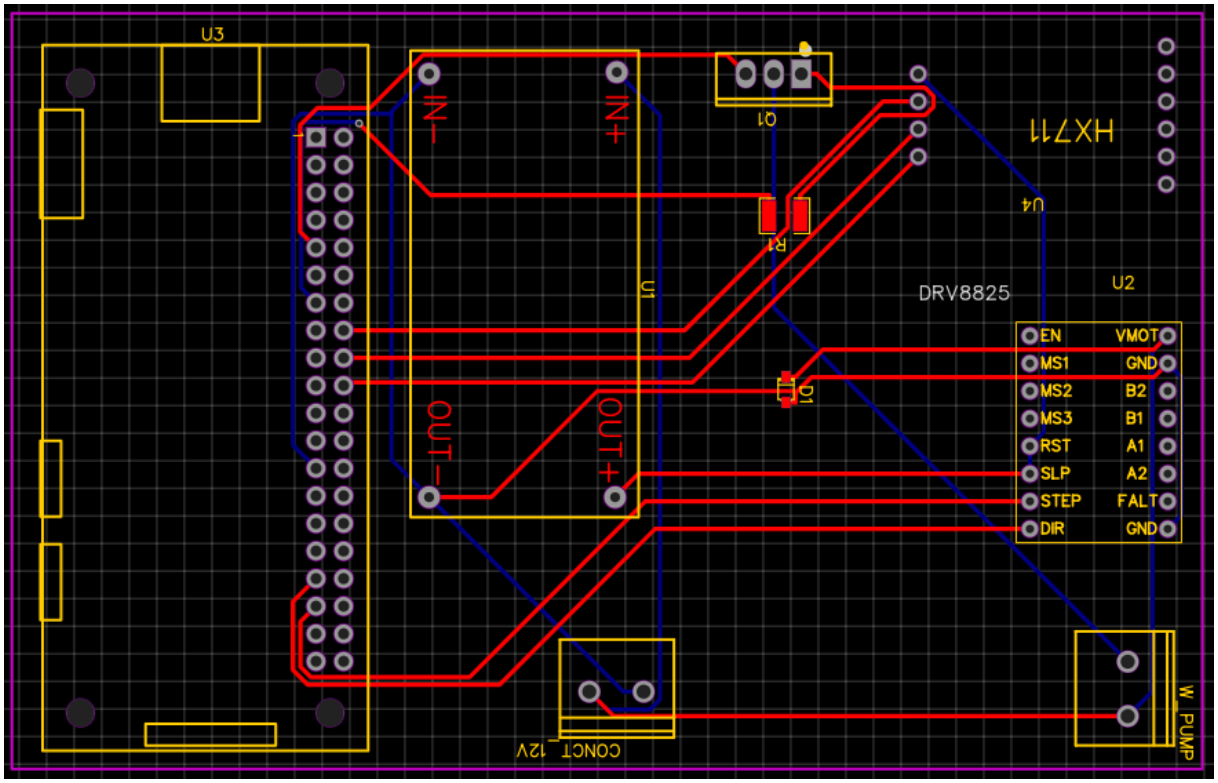
O desenvolvimento do projeto mecânico se deu no aplicativo *SolidWorks* 2021. Utilizou-se as funções de retângulo centrado, retângulo de canto, círculo, *Smart Dimension*, extrusão preenchida e oca. O modelo resultou nas Figuras 16, 17, 18, 19.

Figura 12 – Esquemático para PCB



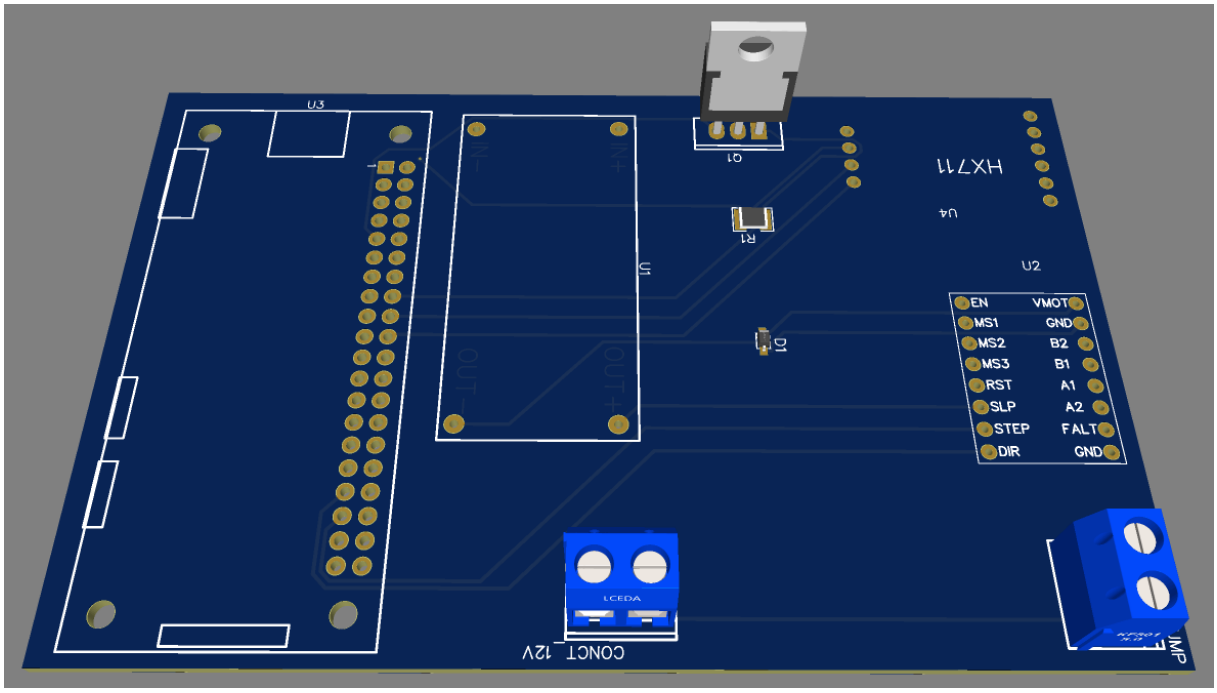
Fonte: Autoria própria

Figura 13 – PCB com rotas



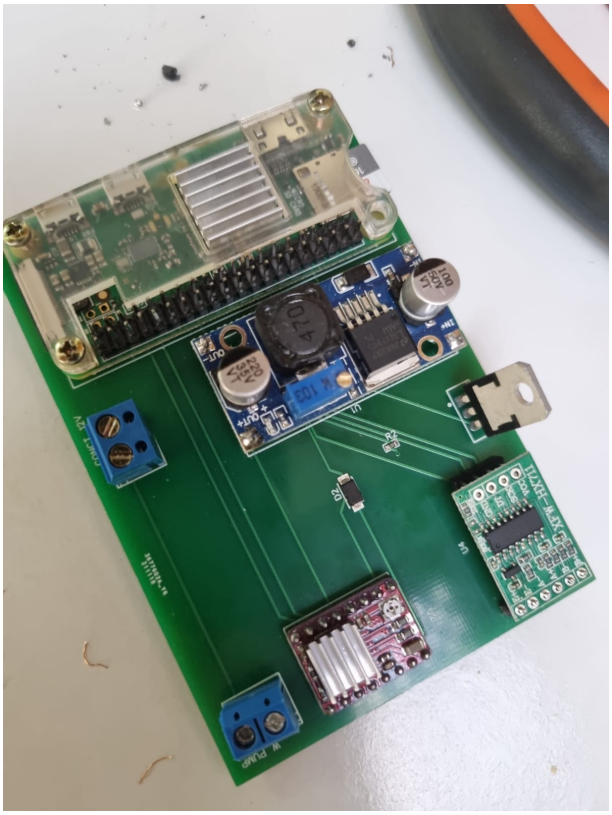
Fonte: Autoria própria

Figura 14 – PCB em 3D



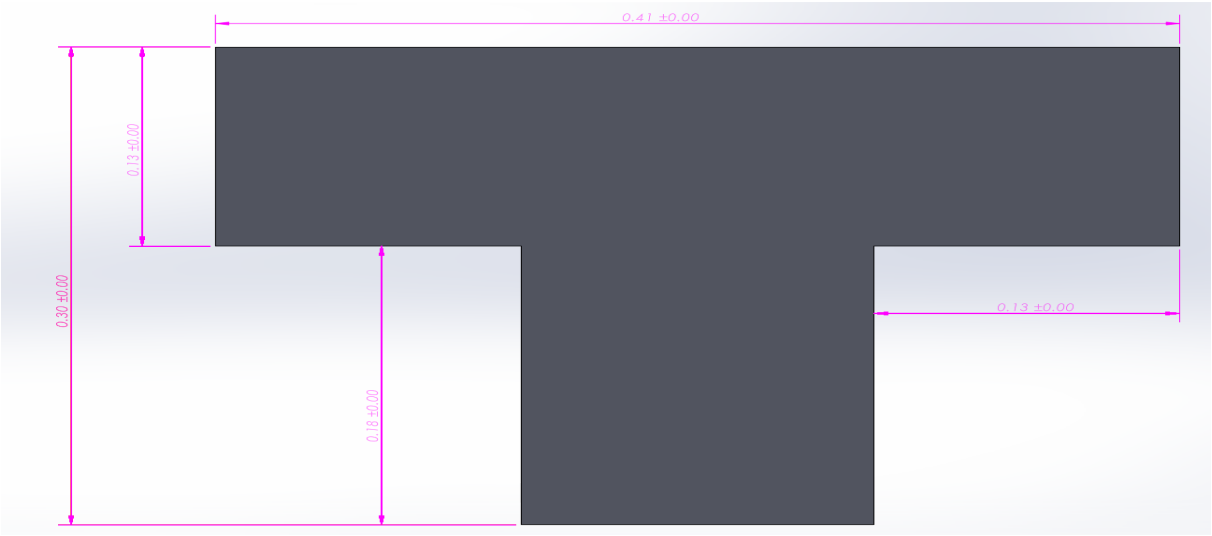
Fonte: Autoria própria

Figura 15 – PCB final



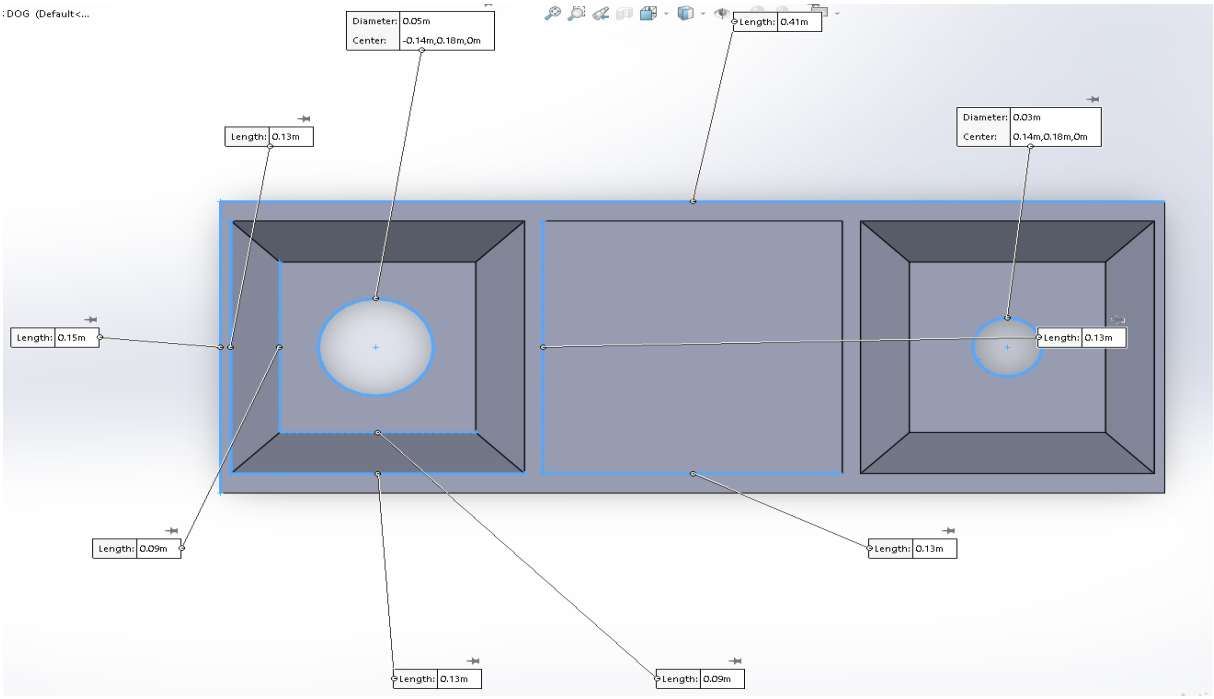
Fonte: Autoria própria

Figura 16 – Modelo 3D - Frontal



Fonte: Autoria própria

Figura 17 – Modelo 3D - Vista Aérea

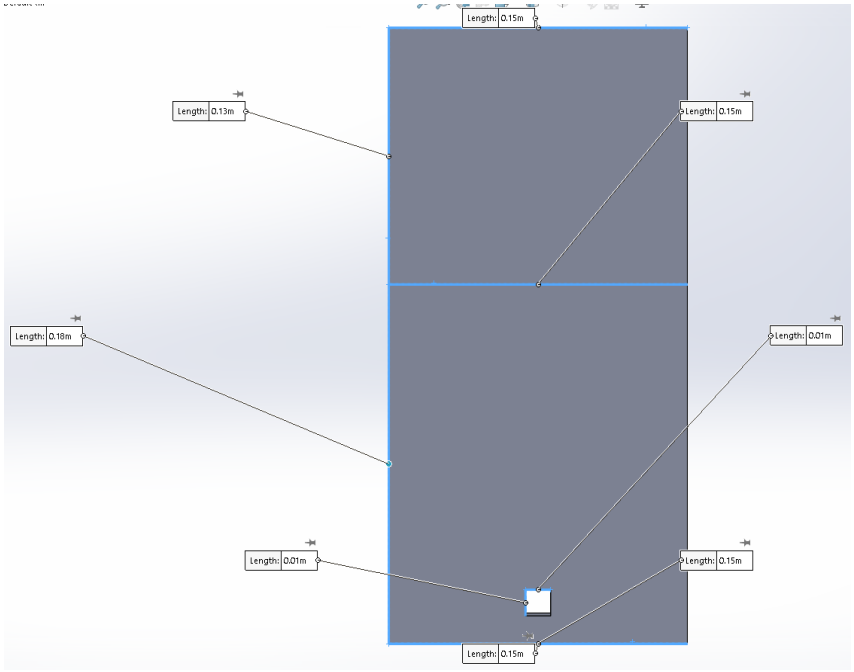


Fonte: Autoria própria

3.2.3 Montagem do Protótipo

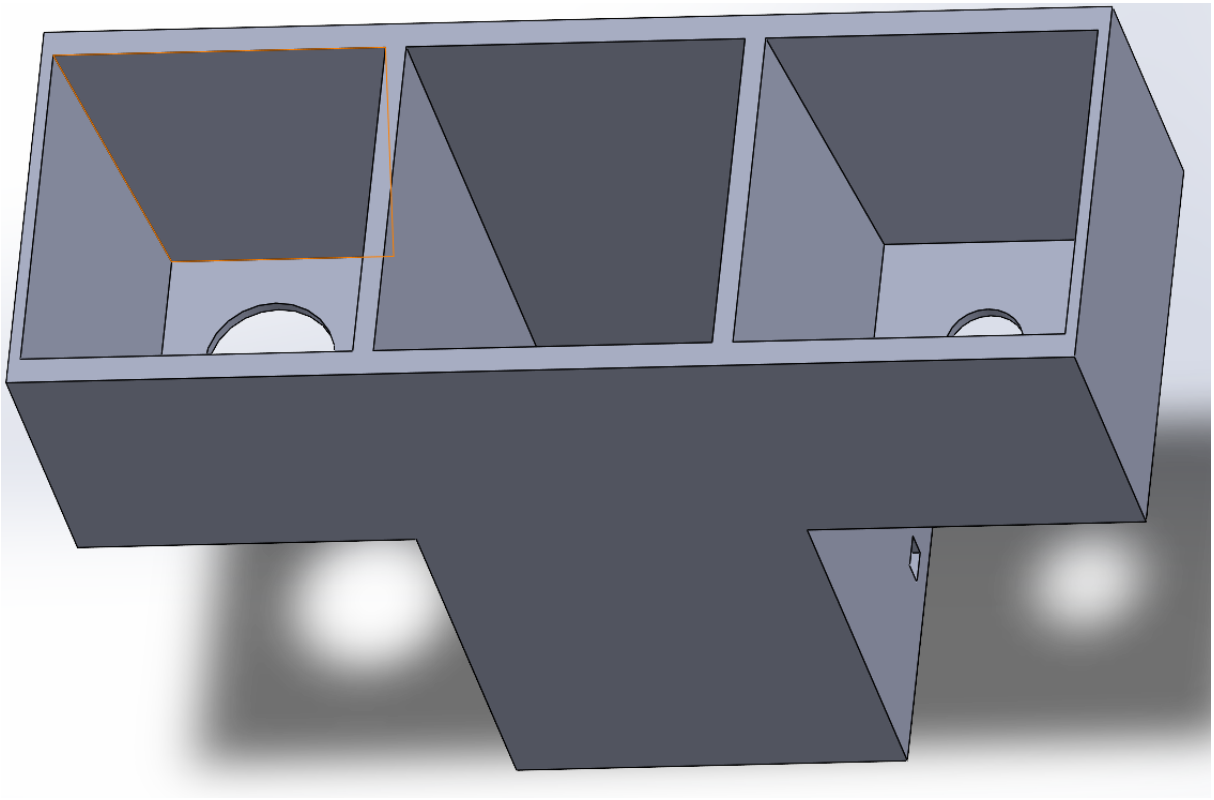
A montagem do protótipo, agora em MDF, se deu utilizando parafusos. As chapas de madeira foram compradas pré-cortadas nos tamanhos do projeto e então, foram unidas para dar forma ao recipiente do alimentador. Com o molde montado e os módulos soldados na PCB, esta foi fixada no alimentador junto com o motor de passo, bomba d'água, a fonte e os recipientes plásticos para água e comida.

Figura 18 – Modelo 3D - Lateral



Fonte: Autoria própria

Figura 19 – Modelo 3D - Inclinado



Fonte: Autoria própria

Os recipientes de plástico foram implementados pois caso a comida e a água ficassem em contato direto com o MDF, a madeira mofaria e apodreceria, diminuindo a vida útil do reservatório. Os fios foram então higienizados de maneira que não fossem danificados ou expostos.

Foram feitos furos nas laterais da coluna central do alimentador para passar os fios das células de carga para dentro do módulo. A célula de carga foi fixada usando duas mãos francesas de metal e parafusos.

Para controlar a passagem de ração, foi acoplado no eixo de motor de passo uma chapa plástica de dois milímetros menor que a passagem de ração para permitir a sua rotação.

A Figura 20 apresenta o protótipo montado.

Figura 20 – Protótipo montado

Fonte: Autoria própria

3.3 Aplicativo *mobile*

Como atualmente os dispositivos *smartphones* têm sido cada vez mais utilizados pelos usuários em comparação aos computadores de mesa ou *laptops*, foi escolhido para este projeto o desenvolvimento de um aplicativo móvel, ao invés de um aplicativo **web** que seria acessado pelo navegador (BROADBAND SEARCH, 2021).

Os requisitos deste aplicativo eram:

- O agendamento de refeições;

- A escolha da quantidade de porções para as refeições;
- A disponibilidade da informação sobre a quantidade de água e ração nos potes;
- Cadastro de usuário para o acesso das funcionalidades acima.

A linguagem de programação principal usada para o desenvolvimento foi TypeScript. O editor de texto usado para o desenvolvimento do aplicativo foi VSCode, ou *Visual Studio Code* (MICROSOFT, 2021a).

Como gerenciador de pacotes, foi usado o NPM, *Node Package Manager*, que permite a instalação de recursos, como bibliotecas, no projeto através da linha de comando do Windows (NPM, INC, 2021).

O nome escolhido para o aplicativo foi *FeedMyPet*.

3.3.1 React Native

No desenvolvimento convencional de aplicativos, é necessário desenvolver em duas diferentes linguagens, como por exemplo Swift para iOS, e Kotlin para Android. Para evitar este problema, foi escolhida a *framework* React Native. Além da possibilidade de desenvolver apenas um aplicativo cujas funcionalidades serviriam tanto para Android quanto para iOS, outro fator que influenciou na adesão da *framework* ao projeto foi a vasta quantidade de bibliotecas, recursos e tutoriais disponíveis para ela.

3.3.2 Expo

Para a visualização do aplicativo durante o desenvolvimento em cada ambiente, Android e iOS, seria necessário configurar e instalar diferentes emuladores para cada um. No entanto, com o Expo, é possível visualizar e utilizar o aplicativo durante o desenvolvimento diretamente do *smartphone* (EXPO, 2021d).

O Expo é uma *framework* e plataforma para aplicativos React. No caso deste projeto, foram usados a Expo CLI, *Command Line Interface*, e o Expo Go, o aplicativo, ou Expo Client, que permite a execução do projeto no *smartphone* (EXPO, 2021b).

Após inicializar o projeto através da Expo CLI, o Expo gera um arquivo JavaScript para ser executado pelo Expo Client através de algo chamado Metro Bundler. Então, o Expo inicializa o *Server* de Desenvolvimento Expo, e fornece um QR Code com o endereço local para o *Expo Client* no *smartphone* ler, conectar-se e receber os dados necessários para executar o aplicativo. Enquanto o *server* estiver funcionando, o Expo Client recarrega as mudanças feitas no código pelo desenvolvedor, o que facilita o teste de novas funcionalidades durante o desenvolvimento (FACEBOOK INC., 2021a).

3.3.3 Telas

No total foram feitas oito telas para o aplicativo. A seguir são detalhadas as telas do aplicativo e suas funcionalidades, assim como os aspectos mais importantes de seus desenvolvi-

mentos.

3.3.3.1 Rotas

Para realizar a navegação entre as telas, foi criado um componente que contém as rotas para todas as páginas usando a biblioteca *React Navigation*. Esta biblioteca permite a navegação simples entre as telas do aplicativo. O arquivo principal do aplicativo tem a função de retornar estas rotas, para que as páginas sejam acessáveis (EXPO, 2021c).

3.3.3.2 *Welcome*

A tela *Welcome* é a primeira tela que o usuário visualiza quando abre o aplicativo. Nela há a logo do *FeedMyPet*, um título, um texto breve explicando o benefício de usar o produto, e um botão para avançar abaixo. A Figura 21 apresenta a tela.

A logo foi obtida através do *plugin* Iconify no programa Figma. Ela foi criada por Dave Gandy e distribuída sob a licença *Attribution 4.0 International (CC BY 4.0)*, a qual torna possível o compartilhamento e adaptação do ícone por quaisquer fins, mesmo comerciais. Uma mudança feita na logo foi a sua cor, que antes era preta, e foi alterada para vermelha (ICONIFY, 2021).

O ícone no botão é chamado de "chevron-right"e foi obtido através da biblioteca *vector-icons* do pacote Expo.

3.3.3.3 *UserIdentification*

Na tela *UserIdentification* o usuário preenche o e-mail com o qual realizou o cadastro, e a sua senha. A tela começa com um texto indicando que é a página de “Login”, dois campos de texto com *placeholders* indicando o que deve ser preenchido. Logo abaixo há o botão “Confirmar” que chama uma função que valida o preenchimento dos campos, e utiliza a biblioteca *Axios* para realizar uma requisição HTTP e enviar o e-mail e a senha do usuário via o URL do *server* no Heroku através da rota “/user/signin” do *server* (AXIOS, 2021).

Mais explicações sobre as rotas e hospedagem do *server* no Heroku são detalhadas na seção 3.5, deste documento. Caso o usuário não preencha os dados corretos, ou deixe de preencher algum dos campos, receberá um aviso em sua tela. A Figura 22 apresenta a tela.

Logo abaixo do botão “Confirmar”, há um texto indicando ao usuário que ainda não tem conta como realizar o cadastro. No fim da tela, há o botão “Quero me cadastrar!” que chama uma função que lida com a navegação para a próxima tela. Abaixo do botão, é indicado o e-mail de suporte “support@feedmypet.com”, para caso o usuário encontre problemas ou tenha alguma dúvida sobre a sua conta.

3.3.3.4 *RegisterForm*

A tela *RegisterForm* é a primeira tela de cadastro. Nela há um título indicando o preenchimento dos dados, campos de texto para o nome do usuário, nome do seu animal de

Figura 21 – Tela *Welcome*

Fonte: Autoria Própria

estimação, e-mail, senha, e confirmação de senha. No canto superior esquerdo há um botão para retornar para a tela anterior, *UserIdentification*, e outro botão abaixo dos campos com o texto "Confirmar". A Figura 23 apresenta a tela.

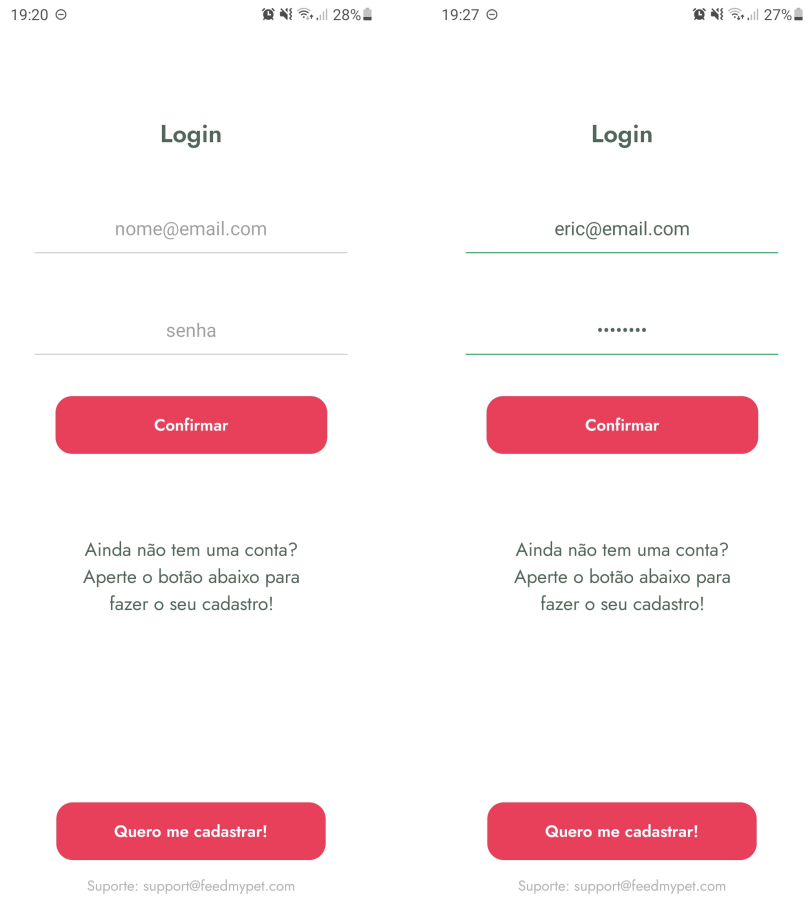
O botão de retorno foi um elemento criado para ser importado nas páginas necessárias, a fim de aderir à reutilização de código no projeto através de componentes TypeScript.

Ao ser pressionado, o botão "Confirmar" chama uma função que verifica se todos os campos foram preenchidos e valida o e-mail e a senha usando a biblioteca "yup" que permite a validação de valores seguindo um formato pré-definido (YUP, 2021).

Caso todos os campos tenham sido preenchidos, as senhas preenchidas sejam iguais, a senha tenha no mínimo 8 caracteres e o formato do e-mail seja válido, uma requisição HTTP é enviada para o *server* através do *Axios* pela rota "/user/signup" do *server*. O funcionamento das rotas é explicado na seção 3.5.

Caso haja algum erro, como por exemplo já existir um cadastro com o mesmo e-mail, este erro retorna em forma de uma janela de alerta para o usuário. Caso o cadastro tenha sido completado com sucesso, o e-mail e nome do *pet* são salvos no dispositivo do usuário através da

Figura 22 – Tela *UserIdentification*, à esquerda não preenchida e à direita preenchida.



Fonte: Autoria Própria

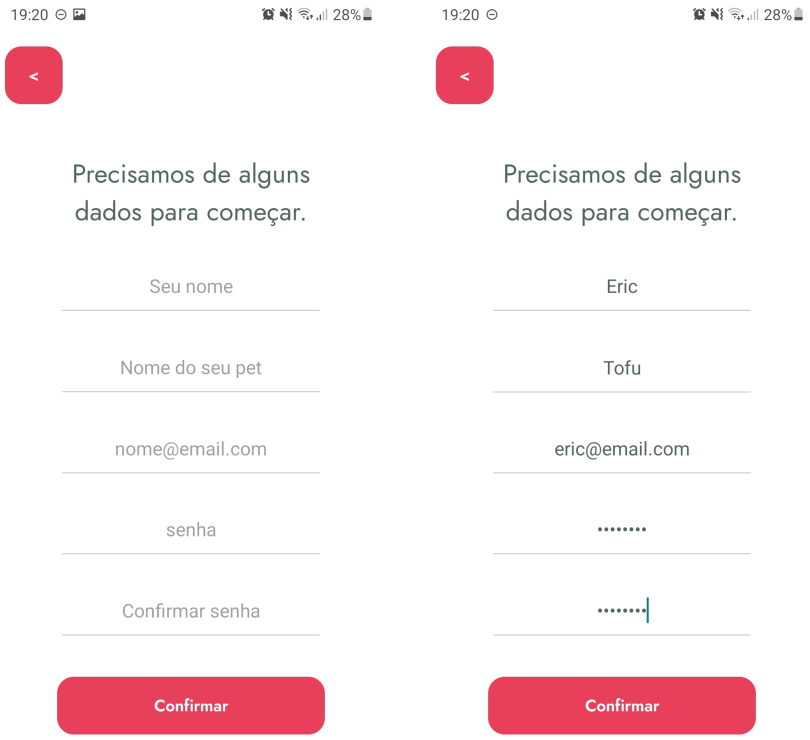
biblioteca chamada *AsyncStorage*, e ocorre a navegação para a próxima tela *RegisterFood*. A biblioteca *AsyncStorage* permite a criação de um sistema de armazenamento persistente para o React Native (ASYNCSTORAGE, 2021).

3.3.3.5 *RegisterFood*

Na tela *RegisterFood* há um botão no canto superior esquerdo para a navegação para a tela anterior, um título indicando o preenchimento dos campos de texto, texto indicando a quantidade de ração em cada porção, um campo de texto para a quantidade de refeições por dia, e outro campo de texto para a quantidade de porções por refeição. Como os textos *placeholder* indicam, o usuário pode escolher de uma a cinco refeições por dia, e de uma a cinco porções por refeição. Cada porção contém 10g de ração seca, um valor razoável que deve atender às necessidades da maior parte dos cachorros e gatos de pequeno a médio porte (PETZ, 2021).

Ao ser pressionado, o botão “Confirmar” chama uma função que valida as informações dos campos, e caso sejam válidos salva localmente no dispositivo do usuário através da *AsyncStorage*. Após isto, ocorre a navegação para a próxima tela. A Figura 24 apresenta a tela.

Figura 23 – Tela *RegisterForm*, à esquerda não preenchida e à direta preenchida.



Fonte: Autoria Própria

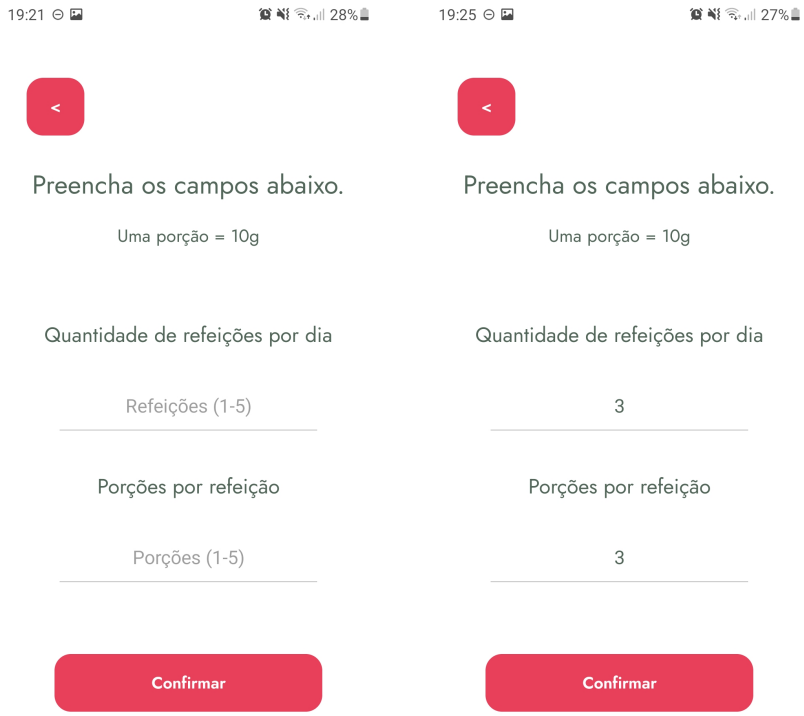
3.3.3.6 *RegisterFoodTime*

Na tela *RegisterFoodTime* há o botão para navegar para a tela anterior no canto superior esquerdo, um texto indicando a escolha dos horários das refeições do animal de estimação, os campos de horário, e o botão "Cadastrar horário". Para realizar a escolha do horário, foi utilizada a biblioteca *DateTimePicker*, que permite a utilização de um componente de escolha de data e horário nativo para Android e iOS (DATETIMEPICKER, 2021). A Figura 25 apresenta a tela.

Para variar a quantidade de horários disponíveis baseado na entrada do usuário na tela anterior, "RegisterFood", foi usado a *AsyncStorage* para resgatar a quantidade de refeições por dia do armazenamento, e impor condições para que os campos de horários fossem carregados de acordo com a quantidade de refeições escolhida.

Ao ser pressionado, o botão "Cadastrar horário" chama uma função que regasta da *AsyncStorage* os dados necessários como e-mail do usuário, quantidade de refeições, porções por refeição, e o peso da comida e da água que foram inicializados como zero. Estes dados em conjunto com os horários definidos são inseridos em um objeto e através do *Axios* uma requisição HTTP é enviada para o *server* pela rota "/user/register" do *server*. Mais explicações sobre as

Figura 24 – Tela *RegisterFood*, à esquerda não preenchida e à direta preenchida.



Fonte: Autoria Própria

rotas encontram-se na seção 3.5. Caso haja algum erro, a mensagem de erro aparece como um alerta na tela do usuário. Caso o cadastro seja bem sucedido, ocorre a navegação para a próxima tela, *RegisterSuccess*.

3.3.3.7 *RegisterSuccess*

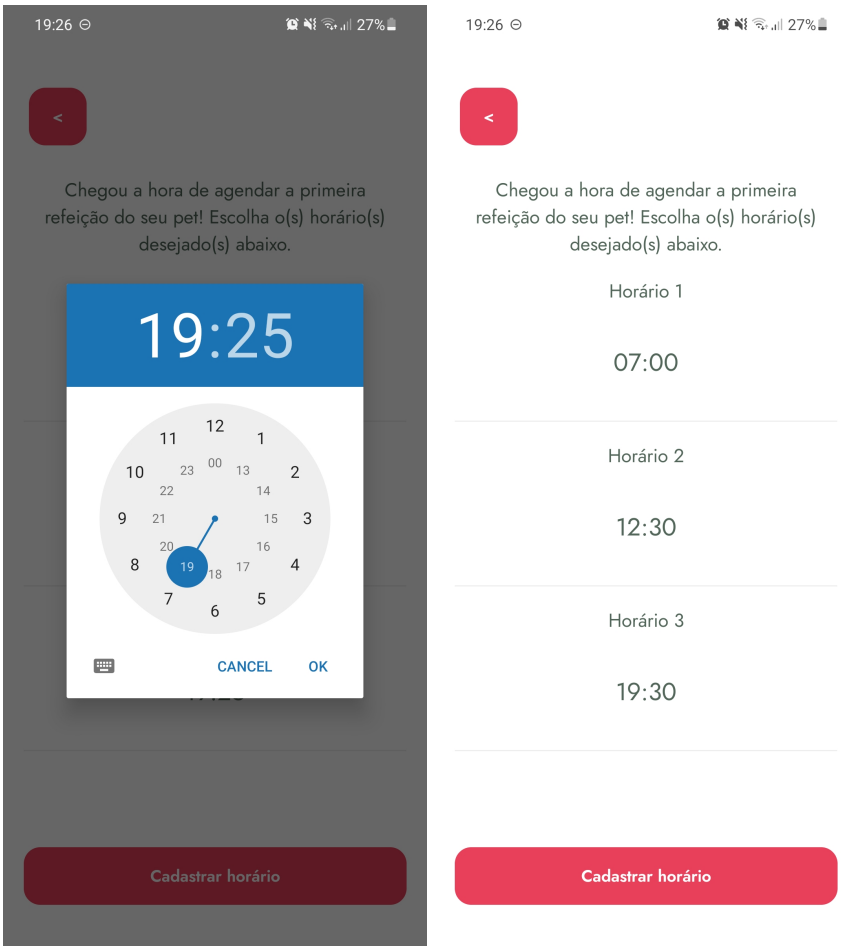
A função da tela *RegisterSuccess* é mostrar para o usuário que o cadastro ocorreu com sucesso através do texto no centro da tela “Cadastro completo!”. Ao ser pressionado, o botão com o texto "Começar"chama uma função que navega para a próxima tela “UserIdentification”. A Figura 26 apresenta a tela.

3.3.3.8 *ModuleSelect*

Na tela *ModuleSelect* há um *header* com uma saudação “Olá” no topo da tela, um texto indicando para o usuário escolher o módulo cadastrado e um botão com o nome do animal de estimação cadastrado no módulo. A Figura 27 apresenta a tela.

Os dados do módulo cadastrado são buscados através do *Axios* em uma requisição

Figura 25 – Tela *RegisterFoodTime*, à esquerda seleção do horário e à direta horários selecionados.



Fonte: Autoria Própria

HTTP, pela rota "/user/modules" do *server*, e são salvos no armazenamento local através da *AsyncStorage*. Caso haja algum erro, um alerta aparece na tela do usuário.

Ao ser pressionado, o botão com o nome do animal de estimação cadastrado no módulo chama uma função que navega para a próxima tela, *ModuleEdit*.

3.3.3.9 *ModuleEdit*

Na tela *ModuleEdit* há o botão de retorno para a tela anterior no canto superior esquerdo, um texto explicando que o usuário pode atualizar os dados, campos de texto para o nome do animal de estimação, quantidade de refeições, porções por refeição, campos de horário para os horários das refeições, e o valor da quantidade de água e ração que há nos potes. A Figura 28 apresenta a tela.

A função desta tela é apresentar os dados que foram salvos no banco de dados pelo usuário, e também as quantidades de água e comida nos potes do animal de estimação. Através dos dados salvos na tela anterior ModuleSelect no *AsyncStorage*, os campos são atualizados com

Figura 26 – Tela *RegisterSuccess*

Fonte: Autoria Própria

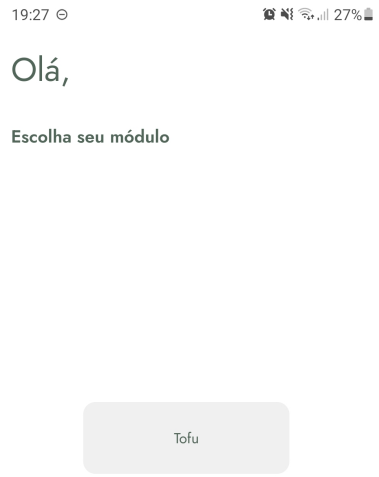
os dados atuais do módulo do usuário. O usuário pode alterar os campos, fora a quantidade de ração e água.

Ao pressionar o botão com o texto “Cadastrar horário”, através do *Axios* os novos dados são enviados por uma requisição HTTP pela rota “/user/register” do *server*, e o usuário é levado à página *UserIdentification*.

3.3.4 Exportação

O processo para a exportação do *APK*, ou Pacote de Aplicativo Android, é explicado nas documentações do *Expo*, assim como a exportação do pacote para iOS. Para propósitos demonstrativos, foi gerado apenas o *APK*. Primeiro é explicitado no arquivo *app.json* do projeto os nomes dos pacotes para Android e iOS. Logo em seguida, através da Expo CLI, executa-se um comando para construir o *APK* do aplicativo. Assim que o *APK* é construído, o Expo faz o *upload* do pacote para sua plataforma através da conta do dono do projeto, onde é possível fazer o download do pacote. Basta transferir o *APK* para o dispositivo Android desejado e realizar a instalação (EXPO, 2021a).

Figura 27 – Tela *ModuleSelect*



Fonte: Autoria Própria

3.4 BANCO DE DADOS

Como o projeto não requer operações relacionais entre os dados de cada usuário, o MongoDB foi escolhido para o armazenamento dos dados. Como ele é não relacional e orientado a documentos, a interação com os dados é feita de forma mais simples e flexível, ao contrário de banco de dados relacionais.

Outra razão pela escolha, foi a vasta quantidade de recursos e documentação do MongoDB, o que auxiliou no processo de desenvolvimento.

3.4.1 Hospedagem

É possível criar um banco de dados MongoDB local, mas para os propósitos do projeto, foi escolhido realizar sua hospedagem no MongoDB Atlas. O MongoDB Atlas é um serviço de hospedagem de MongoDB na nuvem, que é gratuito até certo espaço de armazenamento. Este serviço utiliza outros serviços como AWS, Google Cloud e Azure para a hospedagem de seus dados (MONGODB, INC, 2021a).

Figura 28 – Tela *ModuleEdit*, à esquerda o começo da tela e à direta o fim.

Fonte: Autoria Própria

3.4.2 Conexão

Após o cadastro de uma conta gratuita no MongoDB Atlas, um banco de dados foi criado no serviço para armazenar os dados dos usuários. O MongoDB Atlas fornece um URI que serve como ponte de conexão com o *cluster*, ou aglomerado, que contém o banco de dados dentro do serviço. Este URI foi usado para realizar a conexão tanto do aplicativo e *server* com o banco de dados, quanto do Raspberry Pi com o banco de dados.

O MongoDB tem um protocolo chamado *MongoDB Wire Protocol*, um protocolo baseado em *sockets*. Na documentação, é explícito que a comunicação com o *server* do banco de dados acontece através de um *socket* comum TCP/IP (MONGODB, INC, 2021b).

3.4.3 Dados do usuário

Os dados dos usuários são armazenados no banco de dados em documentos JSON. Apesar de o e-mail ser o identificador único no modelo de dados do projeto, o MongoDB também cria um identificador único para cada documento. Os horários das refeições são salvos no fuso horário UTC+0. A Figura 21 apresenta um documento com os dados de um usuário cadastrado.

Figura 29 – Documento JSON contendo os dados cadastrados de um usuário no MongoDB Atlas

Fonte: Autoria Própria

3.5 SERVER

Nesta seção é explicado o desenvolvimento e funcionalidade do *server*. Para o desenvolvimento do *server* foi utilizada a *framework Express* do *runtime* Node.js. Também foi utilizado o pacote *cors*, que é um *middleware Express* responsável por incluir o mecanismo *CORS*, ou *Cross-origin resource sharing*, permitindo o compartilhamento de recursos entre domínios diferentes, para que os domínios especificados no *server* não sejam bloqueados por questões de segurança. Além disso, as requisições são formatadas em formato JSON (OPENJS FOUNDATION, 2021b).

3.5.1 Conexão com o banco de dados

Para realizar a conexão do *server* com o banco de dados, é preciso obter o *URI*, Identificador de Recurso Único do banco de dados em questão. No *server*, foi usado a biblioteca *dotenv* que permite o carregamento variáveis de ambiente a partir de um arquivo *.env* na pasta do projeto. Este recurso foi usado para definir o *URI* como uma variável de ambiente (DOTENV, 2021).

Para realizar a conexão com o MongoDB, foi usada a biblioteca *Mongoose*, que permite a modelagem de dados necessária para a validação do modelo dos dados, ou *schema*, e faz a tradução entre objetos no código e os dados no banco de dados (MONGOOSE, 2021).

3.5.2 Rotas da *API*

Para realizar a conexão com o MongoDB, foi utilizado um *schema*, ou modelo de dados, para modelar os dados das requisições e respostas HTTP nas rotas da API.

Através da rota "/signup" é realizado o cadastro do usuário no banco de dados. Após validação do preenchimento dos dados, verifica-se caso o e-mail preenchido já existe no banco de dados. Caso não exista, é utilizada a biblioteca *bcrypt* para realizar o *hashing* da senha do usuário antes de enviá-la para o banco de dados. A senha é transformada em um *hash* de 60 caracteres. Por fim, os dados do usuário são salvos no banco de dados (BCRYPT, 2021).

Através da rota "/signin" procura-se o e-mail inserido pelo usuário no banco de dados, e caso seja encontrado, a senha inserida e o *hash* correspondente no banco de dados são passados pela função *compare* da biblioteca *bcrypt*. Caso a senha seja correta, retorna-se que o *signin* do usuário foi um sucesso.

Através da rota "/register" realiza-se o *update* dos dados inseridos nas telas *RegisterFood* e *RegisterFoodTime* nos dados do e-mail correspondente no banco de dados.

Através da rota "/modules", retorna-se os dados correspondentes ao e-mail fornecido pelo usuário.

3.5.3 Hospedagem

Para que a comunicação do aplicativo com o banco de dados pudesse acontecer fora de uma rede local, foi necessário realizar a hospedagem do *server* na nuvem. O Heroku foi escolhido pela sua confiabilidade, simplicidade de implementação, e por ser gratuito até certa quantidade de projetos e horas mensais de uso (HEROKU, 2021a).

Foi criada uma conta na plataforma do Heroku, e instalada a Heroku CLI no ambiente de desenvolvimento do projeto. Utilizando a Heroku CLI, foi realizada a hospedagem do *server* que antes era local, no Heroku.

Após a hospegadem ser completada, foi obtida uma URL que se tornou o ponto de acesso da API do *server* no aplicativo.

3.6 *FIRMWARE*

Foi utilizado o Raspberry Pi Zero W para a leitura do HX711, e controle do DRV8825 e da chave do transistor que aciona a bomba d'água. O Raspberry Pi Zero W conecta com o Wi-Fi sem necessitar de um módulo externo e é através do Wi-Fi que acontece a comunicação com o *server*. Para o desenvolvimento do *firmware* foi utilizada a linguagem Python.

3.6.1 Configuração do ambiente

Foi instalada no cartão microSD de 16 Gb do Raspberry Pi Zero W uma imagem do sistema operacional *Raspian*, um sistema baseado em uma distribuição do Linux e otimizado para o Raspberry Pi. Então através dos arquivos deste cartão micro SD foi configurado o Wi-Fi local e habilitada a conexão através de SSH (*Secure SHell*), um protocolo que permite a comunicação entre computadores na mesma rede. Neste projeto, o SSH foi utilizado a fim de possibilitar o desenvolvimento remoto do *firmware*, através da linha de comando de outro computador. Esta

configuração do Raspberry Pi é comumente chamada de configuração *headless* (RASPBERRY PI FOUNDATION, 2021b).

3.6.2 Algoritmo

Primeiramente no *script* Python principal do *firmware*, é realizada a conexão com o banco de dados, e procura-se os dados do usuário baseado em seu e-mail. Logo em seguida é comparado o horário atual, que o sistema obtém pela internet, com o horário das refeições cadastradas no banco de dados. Caso seja o mesmo horário, o motor é ativado baseado na quantidade de porções cadastrada pelo usuário. Então, os canais A e B do módulo HX711 são lidos, os pesos são armazenados em variáveis no *script*, e seus valores no banco de dados são atualizados. Por fim, caso o peso da água esteja abaixo de certo limiar, a bomba é ativada. A Figura 30 ilustra o fluxograma do algoritmo do *script* Python.

3.6.2.1 Comunicação com o banco de dados

Para realizar a conexão com o banco de dados, foi escolhido o *driver* oficial do MongoDB, o PyMongo. Este *driver* permite a importação da classe MongoClient, que é uma representação do banco de dados. Foi criada uma instância desta classe para realizar a conexão com o banco de dados no MongoDB Atlas, ter acesso aos métodos de busca e atualização de dados (MONGODB, INC, 2021d).

Para buscar e atualizar os dados do usuário correto, é utilizado o e-mail cadastrado como identificador único. Para os propósitos deste projeto, o e-mail foi fixado no *script* Python.

3.6.2.2 Leitura dos sensores

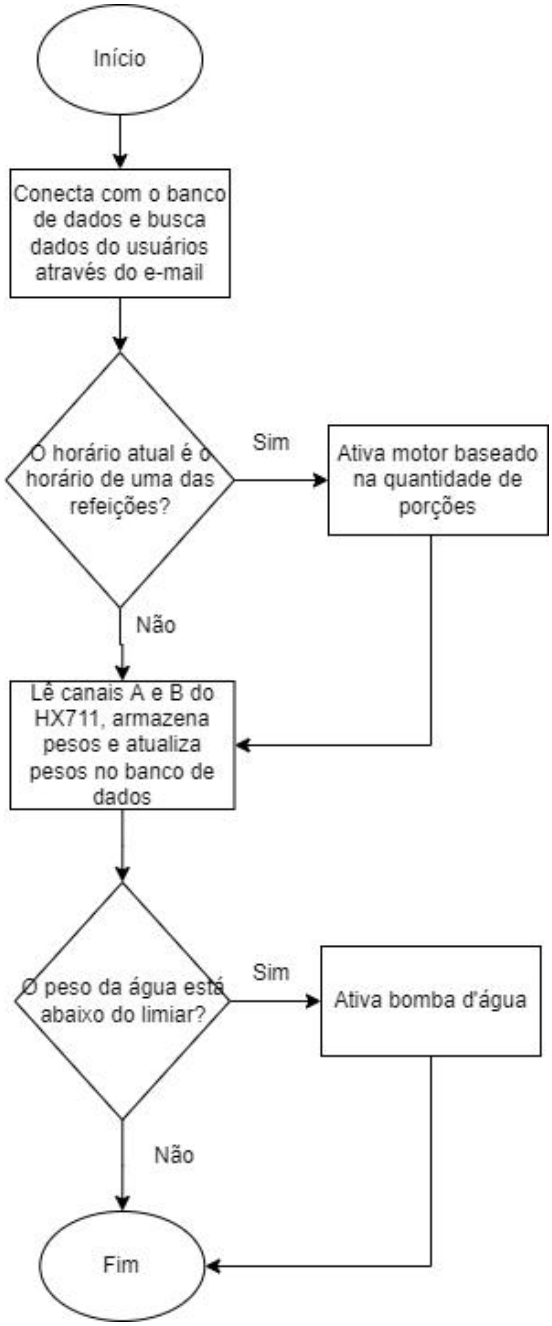
Na leitura dos sensores, foi utilizada a biblioteca HX711, que foi distribuída sob a licença *BSD 3-Clause* (MARCEL ZAK, 2021).

Primeiramente é criada uma instância da classe HX711, que representa o HX711 no projeto, e no método construtor foram definidos os GPIOs conectados aos pinos DOUT e SCK do HX711. O GPIO conectado ao pino DOUT é definido como entrada, e o GPIO conectado ao pino SCK é definido como saída.

Para iniciar a leitura, o estado de SCK é colocado em 0V e DOUT é verificado. Caso seu estado seja nível baixo, os dados estão prontos para a leitura. Um pulso de menos de 60 us é enviado através de SCK afim de pedir o próximo bit a ser lido no DOUT. Então a leitura de DOUT é feita e o bit é armazenado em uma variável. Um deslocamento para a esquerda é feito na variável e cada um dos 24 bits são armazenados no formato complemento de 2. Então, os bits são convertidos para um número decimal. Este processo é repetido 30 vezes, e é aplicada uma média para obter o valor final.

Primeiramente foi inicializado o valor de *offset* do canal, para que ele seja descontado das leituras. Assim, as leituras são correspondentes apenas à quantidade de ração, ou água, no

Figura 30 – Fluxograma do algoritmo do *script* Python



Fonte: Autoria Própria

pote, e não leva em conta o pote do animal. O valor de *offset* foi obtido empiricamente através de várias leituras com o produto já montado.

Para determinar a razão da célula de carga, foi feita uma nova leitura e o valor obtido foi dividido pelo peso aplicado na célula de carga no momento da leitura.

A leitura da ração e água nos potes é o resultado da média de 20 leituras do canal desejado menos o *offset* deste canal, dividido pela razão calculada. Este cálculo foi feito tanto para o canal A quanto o B. Assim foi obtido o valor da quantidade de ração e água nos potes em gramas.

3.6.2.3 Ativação do motor de passo

Foi utilizada a biblioteca *RpiMotorLib*, que possui funções específicas para controle de motores de passo no Raspberry Pi. Iniciou-se declarando quais GPIOs seriam utilizados, e qual a funcionalidade de cada um. Em seguida, foi instanciado um objeto da classe *A4988Nema*, importada da biblioteca *RpiMotorLib*, e foram repassadas em seu construtor as informações de direção, tipo de passo, pinagem e qual *driver* foi utilizado. Através do método *motor_go* da classe *A4988Nema* foram definidos os parâmetros de direção, tipo de passo, quantidade de passos, *delay* entre passos e o *delay* para iniciar o giro do motor (RPIMOTORLIB, 2021).

Como discutido na subseção 3.6.2, caso o horário atual seja o mesmo horário que aquele cadastrado no banco de dados, o motor de passo é ativado, e uma quantidade de ração específica baseada na quantidade de porções escolhida é despejada na tigela de metal do *pet*. O controle da quantidade de comida foi feito através da quantidade de passos do motor.

3.6.2.4 Ativação da bomba

Há uma condição no *script* que compara o valor de peso lido do HX711 com um valor determinado empiricamente. Caso esteja abaixo deste valor, é chamada uma função que coloca em estado alto, +5V, o pino conectado à base do transistor que atua como chave para a alimentação da bomba d'água. Isto resulta na ativação da bomba. Após certo tempo, a função retorna este pino para o estado baixo, 0V, o que desliga a bomba. Para determinar o tempo de ativação da bomba, foram realizados testes até que o valor desejado fosse alcançado.

3.6.3 Agendamento do *script*

No sistema operacional Raspbian, há um *daemon* responsável pela execução de comandos agendados chamado *Cron*. *Daemons* são processos que rodam em *background* do sistema operacional e supervisionam o sistema ou fornecem funcionalidades para outros processos. O comando *crontab* foi utilizado para adicionar a execução do *script* principal do *firmware* a cada 1 minuto (RASPBERRY PI FOUNDATION, 2021a).

3.7 Custos

O custo total do projeto foi de R$1.151,69. Nesta seção é detalhado o custo individual de cada parte do projeto. A impressão do modelo mecânico 3D utilizando o material ABS custaria no mínimo R$1.200,00, o que elevaria o custo total do projeto para R$2.051,69. Utilizando MDF, o custo foi reduzido em R$900,00.

O custo de produção de uma unidade do produto está aproximadamente R$200 acima do preço de produtos concorrentes. No entanto, a maior parte dos alimentadores automáticos de *pets* não oferece reservatórios de água, não possui aplicativo móvel para a programação das refeições e não verifica a quantidade de comida e água nas tigelas do animal. Levando em conta

Tabela 1 – Custos.

Peça	Preço (R$)
Raspberry Pi Zero W	175,07
Raspberry Pi Zero W - Fonte	39,90
Micro SD 16GB	34,70
Raspberry Pi Zero W - Case e Dissipador	21,00
Motor de passo	150,00
Célula de carga 5kg e HX711	35,70
Hélice para despejar comida	10,00
Placa de circuito impresso	190,00
Fonte de alimentação 12v 5A	30,00
MDF	300,00
Tigelas Inox	17,80
Bomba d'água	40,00
Conversor DC-DC	20,00
Driver do motor - DRV8825	23,33
Total	1.151,69

Fonte: Autoria Própria

as características diferenciais deste produto e a redução do custo de produção em escala, o preço deve se tornar ainda mais competitivo.

4 ANÁLISE E DISCUSSÃO DOS RESULTADOS

Neste capítulo foram discutidos os resultados dos testes do projeto. Para facilitar a compreensão, os testes foram divididos em duas seções, de *hardware* e de *software*.

4.1 TESTES DE *HARDWARE*

Os testes foram iniciados primeiramente com os módulos montados numa *protoboard* para verificar a funcionalidade do circuito. Todos os módulos foram conectados conforme o esquemático e o Raspberry Pi programado conforme as saídas utilizadas e funcionalidades requeridas. Foi utilizada uma fonte comercial de tomada classificada como 12V e 2A. A mesma não suportou entregar a corrente necessária. Através da medição de saída de corrente foi averiguado que entregava apenas 0,5A, não sendo suficiente para suportar o circuito. Foi então utilizada uma fonte manufaturada na disciplina de Eletrônica Básica com saída de 3A e 12V fixo. Com esta foi obtido sucesso no teste do circuito. Por ela não ser comercial foi adquirida uma fonte de 12V e 5A do tipo colmeia que, quando utilizada, obteve-se êxito no funcionamento do circuito. Após obtida a garantia de funcionamento do circuito com a bomba d'água e o motor sendo acionados corretamente, foi feita a impressão da PCB. Assim que a placa foi produzida, os mesmos testes foram realizados com sucesso novamente.

A estrutura do alimentador, feita em MDF não exigiu muitos testes. Foram compradas chapas individuais já recortadas nas medidas necessárias que foram, então, parafusadas junto. Na sequência, foi definida a posição do motor e da bomba d'água e com isso, a posição ideal para a PCB. A trajetória dos cabos foi feita de maneira a não deixar cabos soltos ou com risco de rompimento. Os reservatórios de comida foram inseridos de maneira que ficassem bem fixados sem riscos de derrubar comida ou água. As tigelas de metal para depósito de alimento e água enviadas junto com o produto foram postas sobre as células de carga de maneira que ficassem estáveis porém pudessem ser removíveis para que o usuário pudesse fazer a higienização das mesmas.

Com tudo montado, o teste foi refeito ligando a fonte na tomada e verificando se o motor e a bomba d'água funcionariam corretamente. Obtendo sucesso, o teste foi feito enchendo os reservatórios com comida e água, verificando se um passo do motor representava a quantidade desejada de comida e se o tempo que a bomba d'água ficava ligada atendia a quantidade necessária na tigela. A quantidade de água foi despejada na tigela. Já a quantidade de comida variou de 5 a 20 gramas devido ao mecanismo de abertura de ração acoplado ao eixo do motor de passo. Em alguns dos testes, a ração impedia o movimento de rotação do mecanismo, desalinhando momentaneamente o motor. Em média a porção despejada era 10g, portanto foi a execução foi um sucesso.

4.2 TESTES DE *SOFTWARE*

Durante o desenvolvimento foram realizados diversos testes com a interface do aplicativo usando o Expo Go. Várias funcionalidades não funcionavam, como por exemplo, a escolha do horário não respondia a toques, o teclado ocultava parte de algumas telas, e outros pequenos *bugs* na funcionalidade da interface. Todos os problemas foram eventualmente resolvidos.

Após a exportação do *APK* do aplicativo *mobile*, este arquivo foi transferido para um *smartphone* Android, onde foi executado e o aplicativo instalado. Durante o teste foi realizado o cadastro de um novo usuário, que foi um sucesso. No MongoDB Atlas foi possível observar as informações cadastradas, e o *hash* da senha. Então, foram utilizados o e-mail e a senha cadastrados para realizar o *login*. Também foi um sucesso, tanto a tela *ModuleSelect*, 3.3.3.8, quanto a tela *ModuleEdit*, 3.3.3.9, continham as informações do módulo.

Inicialmente foi considerado e testado o uso de células de carga com capacidade de 50kg. No entanto, provavelmente devido à falta de precisão, que é 25 gramas, os valores lidos oscilavam muito, e muitas vezes não eram lidos pelo *software*. Outro ponto negativo, é que estas células de carga possuem apenas meia ponte de *Wheatstone* em seus circuitos, necessitando de complemento externo para funcionar individualmente. Portanto, células de carga de 5kg foram utilizadas, e estas funcionaram na primeira tentativa.

No teste final, após tudo ser montado, o aplicativo funcionou como esperado. As refeições foram despejadas nas porções corretas e no horário correto, e a água foi despejada quando o seu nível diminuía demais. Os pesos também foram atualizados corretamente, tanto no MongoDB, quanto no aplicativo. O sistema também respondeu corretamente à edição dos dados do usuário.

O produto foi testado durante 24 horas seguidas. O alimento foi despejado nas tigelas de acordo com o horário programado no aplicativo corretamente. Já a água, foi despejada quando o nível mínimo foi atingido.

5 CONCLUSÃO

Como a quantidade de *pets* teve um aumento substancial durante 2020 e 2021, o mercado de animais de estimação cresceu muito. Além disso, as pessoas voltaram a viajar, então há uma grande necessidade de cuidar da alimentação dos *pets* à distância. Portanto, o desenvolvimento de produtos voltados aos donos de animais de estimação é uma ótima oportunidade.

O produto desenvolvido neste projeto busca atender à demanda crescente no mercado brasileiro, através do agendamento de refeições por um aplicativo móvel e reabastecimento de água automático para o animal. Estas funcionalidades garantem o fornecimento de ração e água para o animal, possibilitando viagens de curto prazo para os donos.

O produto mais completo encontrado durante a pesquisa de concorrentes foi um alimentador que possui um aplicativo móvel para controle de horários e porções, uma câmera e um microfone (EKAZA, 2021). O preço deste produto era de R$845,00. No entanto, este produto não possui reservatório de água e o pote para a ração era de plástico, o que pode causar acne nos animais de estimação (PETZ, 2020). Apesar do custo de produção do protótipo ser R$1.151,69, há vários pontos essenciais que são diferenciais em relação aos outros produtos do mercado, como potes de metal, reservatório de água e *feedback* para o usuário sobre a quantidade restante de ração e água nas tigelas. Considerando também a redução de custo ao realizar a produção em escala, o preço do produto seria competitivo.

Além dos objetivos propostos e alcançados durante o desenvolvimento, o trabalho buscou agregar conhecimento ao curso de Engenharia Eletrônica através da utilização de tecnologias e linguagens de programação não exploradas na grade curricular, como React Native, NodeJS, MongoDB e JavaScript.

5.1 TRABALHOS FUTUROS

Uma possível ampliação da funcionalidade do projeto seria a adição de mais de um módulo por usuário. Assim, o mesmo usuário poderia escolher entre mais de um módulo, que alimentariam mais de um animal, e obter os mesmos benefícios e funcionalidades.

A impressão em 3D do objeto ao invés da manufatura em MDF seria uma possibilidade de melhoria assim como a elaboração de um recipiente feito em 3D para armazenamento da comida e da água que pudesse ser retirado do molde. Além disso, poderia ser implementado um motor menor, com menos torque e mais silencioso. Isso também se aplicaria para a bomba d'água, que poderia ser com vazão menor, consequentemente menor em tamanho e mais silenciosa.

Devido ao mecanismo de controle de ração utilizado no projeto não ter um nível alto de precisão e repetibilidade, uma melhoria seria utilizar uma rosca transportadora helicoidal acoplada diretamente ao eixo do motor. Assim, a ração não impediria o movimento, e as porções seria mais precisas.

Outra melhoria seria adicionar um suporte entre a célula de carga e a tigela para fixar a tigela.

Seria possível também projetar uma fonte de alimentação junto com um conversor DC-DC e o restante do circuito em apenas uma PCB para economizar em espaço e em custo de compra de materiais.

Como o e-mail, e o SSID e senha do Wi-Fi foram fixadas no Raspberry Pi, outra possibilidade de melhoria seria a implementação de um processo de instalação do módulo no lar do usuário. Seria possível hospedar uma página web em um *server* local no Raspberry Pi. Então, o usuário poderia se conectar localmente no módulo através do Wi-Fi de seu *smartphone* e ler um *QR code* que viria junto com o manual do produto para acessar a página. Nesta página, o usuário colocaria seu email, e o SSID do Wi-Fi e a sua senha. O *server* local receberia estes valores e rodaria um *script* para configurar o Raspberry Pi e criar um conexão com o Wi-Fi do lar do usuário.

A implementação de rolagem em todas as páginas do aplicativo móvel também poderia ajudar a garantir uma melhor compatibilidade com *smartphones* com telas menores, e o desenvolvimento de um aplicativo web permitiria o usuário interagir com o módulo através de um navegador.

Uma melhoria de segurança para garantir a alimentação do animal seria a implementação de um *failsafe* no *firmware* para o caso da alimentação de energia do módulo ser interrompida durante o horário de refeições.

5.2 CONSIDERAÇÕES FINAIS

Este trabalho foi uma ótima maneira de aplicar a base teórica e prática adquirida durante o curso de Engenharia Eletrônica em um projeto prático repleto de tecnologias que são amplamente utilizadas no mercado atualmente.

Referências

4LINUX. *O que é o sistema Operacional Linux?* 2021. Disponível em: <https://4linux.com.br/o-que-e-linux/>. Acesso em: 5 de novembro de 2021. Citado na página 23.

ABINPET. **Mercado Pet Brasil 2020**. 2020. Disponível em: <http://abinpet.org.br/wp-content/uploads/2020/06/abinpet_folder_2020_draft3.pdf>. Acesso em: 21 de julho de 2021. Citado 2 vezes nas páginas 16 e 17.

ABLY. **WebSockets vs. HTTP**. 2021. Disponível em: <https://ably.com/topic/websockets-vs-http>. Acesso em: 5 de novembro de 2021. Citado na página 27.

ALL DATASHEET. **DRV8825 DATASHEET**. 2021. Disponível em: <https://www.alldatasheet.com/datasheet-pdf/pdf/432263/TI/DRV8825.html>. Acesso em: 8 de novembro de 2021. Citado na página 30.

ALL DATASHEET. **HX711 DATASHEET**. 2021. Disponível em: <https://www.alldatasheet.com/datasheet-pdf/pdf/1132222/AVIA/HX711.html>. Acesso em: 8 de novembro de 2021. Citado na página 33.

ALL DATASHEET. **TIP120 Datasheet**. 2021. Disponível em: <https://www.alldatasheet.com/datasheet-pdf/pdf/175429/ONSEMI/TIP120.html>. Acesso em: 8 de novembro de 2021. Citado na página 32.

AMAZON WEB SERVICES. **What is Mobile Application Development?** 2021. Disponível em: <https://aws.amazon.com/mobile/mobile-application-development/>. Acesso em: 5 de novembro de 2021. Citado na página 25.

ANDROID AUTHORITY. **What is Android? Here's everything you need to know**. 2021. Disponível em: <https://www.androidauthority.com/what-is-android-328076/>. Acesso em: 5 de novembro de 2021. Citado na página 23.

ASYNCSTORAGE. **React Native Async Storage**. 2021. Disponível em: <https://github.com/react-native-async-storage/async-storage>. Acesso em: 8 de novembro de 2021. Citado na página 45.

AVAST. **What is TCP/IP and How Does it Work?** 2021. Disponível em: <https://www.avast.com/c-what-is-tcp-ip>. Acesso em: 5 de novembro de 2021. Citado na página 27.

AXIOS. **Getting Started**. 2021. Disponível em: <https://axios-http.com/docs/intro>. Acesso em: 8 de novembro de 2021. Citado na página 43.

BAú DA ELETRôNICA. **LM2596 DATASHEET**. 2021. Disponível em: <https://storage.googleapis.com/baudaeletronicadatasheet/LM2596.pdf>. Acesso em: 8 de novembro de 2021. Citado 2 vezes nas páginas 34 e 35.

BCRYPT. **node.bcrypt.js**. 2021. Disponível em: <https://github.com/kelektiv/node.bcrypt.js>. Acesso em: 8 de novembro de 2021. Citado na página 53.

BINANCE ACADEMY. **What Is Hashing?** 2019. Disponível em: <https://academy.binance.com/en/articles/what-is-hashing>. Acesso em: 5 de novembro de 2021. Citado na página 26.

BROADBAND SEARCH. **Mobile Vs. Desktop Internet Usage (Latest 2021 Data)**. 2021. Disponível em: <https://www.broadbandsearch.net/blog/mobile-desktop-internet-usage-statistics>. Acesso em: 8 de novembro de 2021. Citado na página 41.

CONCEPTA TECHNOLOGIES LLC. **What Is The Difference Between Front-End And Back-End Development?** 2019. Disponível em: <https://www.conceptatech.com/blog/difference-front-end-back-end-development>. Acesso em: 5 de novembro de 2021. Citado na página 24.

DATETIMEPICKER. **React Native DateTimePicker**. 2021. Disponível em: <https://github.com/react-native-datetimepicker/datetimepicker>. Acesso em: 8 de novembro de 2021. Citado na página 46.

DOTENV. **dotenv**. 2021. Disponível em: <https://www.npmjs.com/package/dotenv>. Acesso em: 8 de novembro de 2021. Citado na página 52.

EASYEDA. **EasyEDA**. 2021. Disponível em: <https://easyeda.com/>. Acesso em: 8 de novembro de 2021. Citado na página 34.

EKAZA. **Nutri Alimentador Inteligente Com Câmera Ekaza Pet Fdw020**. 2021. Disponível em: <https://www.petz.com.br/blog/bem-estar/acne-felina/>. Acesso em: 24 de novembro de 2021. Citado na página 60.

EL PROCUS. **What is a Stepper Motor : Types Its Working**. 2021. Disponível em: <https://www.elprocus.com/stepper-motor-types-advantages-applications/>. Acesso em: 5 de novembro de 2021. Citado na página 21.

ENSUS. **Extensômetria (Strain Gauge) – O que é? Quando utilizar?** 2016. Disponível em: <https://ensus.com.br/extensometria-strain-gauge-o-que-e-quando-utilizar>. Acesso em: 5 de novembro de 2021. Citado na página 19.

EUROMONITOR. **Pet Care in Brazil**. 2021. Disponível em: <https://www.euromonitor.com/pet-care-in-brazil/report>. Acesso em: 03 de novembro de 2021. Citado na página 14.

EXPO. **Building Standalone Apps**. 2021. Disponível em: <https://docs.expo.dev/classic/building-standalone-apps/>. Acesso em: 8 de novembro de 2021. Citado na página 49.

EXPO. **Expo Go**. 2021. Disponível em: <https://expo.dev/client>. Acesso em: 8 de novembro de 2021. Citado na página 42.

EXPO. **Getting started**. 2021. Disponível em: <https://reactnavigation.org/>. Acesso em: 8 de novembro de 2021. Citado na página 43.

EXPO. **Introduction to Expo**. 2021. Disponível em: <https://docs.expo.dev/>. Acesso em: 8 de novembro de 2021. Citado na página 42.

FACEBOOK INC. **Concepts**. 2021. Disponível em: <https://facebook.github.io/metro/docs/concepts>. Acesso em: 8 de novembro de 2021. Citado na página 42.

FACEBOOK INC. **Getting Started**. 2021. Disponível em: <https://reactjs.org/docs/getting-started.html>. Acesso em: 5 de novembro de 2021. Citado na página 25.

FACEBOOK INC. **Introducing JSX**. 2021. Disponível em: <https://reactjs.org/docs/introducing-jsx.html>. Acesso em: 5 de novembro de 2021. Citado na página 24.

FAZEDORES. **Raspberry Pi B+: Introdução a Porta GPIO**. 2021. Disponível em: <https://blog.fazedores.com/raspberry-pi-b-introducao-porta-gpio/>. Acesso em: 5 de novembro de 2021. Citado na página 22.

FERNANDA STRICKLAND. **Com avanço da vacinação, cresce a demanda por viagens aéreas**. 2021. Disponível em: <https://www.correiobraziliense.com.br/economia/2021/10/4957380-com-avanco-da-vacinacao-cresce-a-demanda-por-viagens-aereas.html>. Acesso em: 03 de novembro de 2021. Citado na página 17.

HBM. **Como uma célula de carga trabalha**. 2021. Disponível em: <https://www.hbm.com/pt/6768/como-uma-celula-de-carga-trabalha/>. Acesso em: 5 de novembro de 2021. Citado na página 19.

HEROKU. **Build apps for free on Heroku**. 2021. Disponível em: <https://www.heroku.com/free>. Acesso em: 8 de novembro de 2021. Citado na página 53.

HEROKU. **What is Heroku?** 2021. Disponível em: <https://www.heroku.com/about>. Acesso em: 5 de novembro de 2021. Citado na página 26.

HOME STEADY. **How an Electric Water Pump Works.** 2021. Disponível em: <https://homesteady.com/13409072/how-an-electric-water-pump-works>. Acesso em: 5 de novembro de 2021. Citado na página 21.

IBM. **Cloud Computing**. 2020. Disponível em: <https://www.ibm.com/cloud/learn/cloud-computing>. Acesso em: 5 de novembro de 2021. Citado na página 25.

ICONIFY. **Font Awesome 5 Solid**. 2021. Disponível em: <https://icon-sets.iconify.design/fa-solid/cat/>. Acesso em: 8 de novembro de 2021. Citado na página 43.

INVESTOPEDIA. **Apple iOS**. 2021. Disponível em: <https://www.investopedia.com/terms/a/apple-ios.asp>. Acesso em: 5 de novembro de 2021. Citado na página 23.

JORNAL DA USP NO AR 1ª EDIçãO. **Europa veta entrada de brasileiros por falhas no controle da pandemia**. 2021. Disponível em: <https://jornal.usp.br/atualidades/europa-veta-entrada-de-brasileiros-por-falhas-no-controle-da-pandemia/>. Acesso em: 03 de novembro de 2021. Citado na página 17.

JOãO JOSé OLIVEIRA. **Turismo no Brasil deve bombar, mas viagens para fora e de negócios frustram**. 2021. Disponível em: <https://economia.uol.com.br/noticias/redacao/2021/08/08/turismo-cresce-com-domestico-e-verao-mas-sem-eventos-nem-internacional.htm>. Acesso em: 03 de novembro de 2021. Citado na página 15.

JSON. **Introducing JSON**. 2021. Disponível em: <https://www.json.org/json-en.html>. Acesso em: 5 de novembro de 2021. Citado na página 26.

JULIANA AMéRICO. **Mercado pet abre portas para investir além das tradicionais pet shops**. 2021. Disponível em: <https://vocesa.abril.com.br/>. Acesso em: 03 de novembro de 2021. Citado na página 14.

MAKERS GUIDE. **How to control a stepper motor with DRV8825 driver and Arduino**. 2021. Disponível em: <https://www.makerguides.com/drv8825-stepper-motor-driver-arduino-tutorial/>. Acesso em: 8 de novembro de 2021. Citado na página 31.

MARCEL ZAK. **HX711 class for Rasperry Pi Zero, 2 and 3 written in Python 3.** 2021. Disponível em: <https://github.com/gandalf15/HX711>. Acesso em: 8 de novembro de 2021. Citado na página 54.

MAXMILHAS. **Viagem Pós Pandemia: 86% dos brasileiros querem voltar a viajar logo**. 2021. Disponível em: <https://www.maxmilhas.com.br/blog/dicas-de-viagem/viagem-pos-pandemia>. Acesso em: 03 de novembro de 2021. Citado na página 14.

MERCADO LIVRE. **Mini Bomba De Água + Fonte 12v 2a + 2mts Mangueira Arduino**. 2021. Disponível em: <https://produto.mercadolivre.com.br/MLB-1008171738-mini-bomba-de-agua-fonte-12v-2a-2mts-mangueira-arduino-_JM#position=15&search_layout=grid&type=item&tracking_id=322e225c-1afa-4b6a-b5d1-07210c1bcbc3>. Acesso em: 8 de novembro de 2021. Citado na página 31.

MICROSOFT. **Getting Started**. 2021. Disponível em: <https://code.visualstudio.com/docs>. Acesso em: 8 de novembro de 2021. Citado na página 42.

MICROSOFT. **TypeScript for the New Programmer**. 2021. Disponível em: <https://www.typescriptlang.org/docs/handbook/typescript-from-scratch.html>. Acesso em: 5 de novembro de 2021. Citado na página 24.

MINISTéRIO DA SAúDE. **Doses Aplicadas**. 2021. Disponível em: <https://www.gov.br/saude/pt-br/vacinacao>. Acesso em: 08 de novembro de 2021. Citado na página 18.

MONGODB, INC. **Database. Deploy a multi-cloud database**. 2021. Disponível em: <https://www.mongodb.com/atlas/database>. Acesso em: 8 de novembro de 2021. Citado na página 50.

MONGODB, INC. **MongoDB Wire Protocol**. 2021. Disponível em: <https://docs.mongodb.com/manual/reference/mongodb-wire-protocol/>. Acesso em: 8 de novembro de 2021. Citado na página 51.

MONGODB, INC. **NoSQL vs SQL Databases**. 2021. Disponível em: <https://www.mongodb.com/nosql-explained/nosql-vs-sql>. Acesso em: 5 de novembro de 2021. Citado na página 26.

MONGODB, INC. **PyMongo**. 2021. Disponível em: <https://docs.mongodb.com/drivers/pymongo/>. Acesso em: 8 de novembro de 2021. Citado na página 54.

MONGODB, INC. **WhatIsMongoDB**. 2021. Disponível em: <https://www.mongodb.com/what-is-mongodb>. Acesso em: 5 de novembro de 2021. Citado na página 27.

MONGOOSE. **mongoose**. 2021. Disponível em: <https://mongoosejs.com/>. Acesso em: 8 de novembro de 2021. Citado na página 52.

MOZILLA. **Database**. 2021. Disponível em: <https://developer.mozilla.org/en-US/docs/Glossary/Database>. Acesso em: 5 de novembro de 2021. Citado na página 26.

MOZILLA. **Express/Node introduction**. 2021. Disponível em: <https://developer.mozilla.org/en-US/docs/Learn/Server-side/Express_Nodejs/Introduction>. Acesso em: 5 de novembro de 2021. Citado na página 26.

MOZILLA. **HTML basics**. 2021. Disponível em: <https://developer.mozilla.org/en-US/docs/Learn/Getting_started_with_the_web/HTML_basics>. Acesso em: 5 de novembro de 2021. Citado na página 24.

MOZILLA. **An overview of HTTP**. 2021. Disponível em: <https://developer.mozilla.org/en-US/docs/Web/HTTP/Overview>. Acesso em: 5 de novembro de 2021. Citado na página 27.

MOZILLA. **Protocol**. 2021. Disponível em: <https://developer.mozilla.org/en-US/docs/Glossary/Protocol>. Acesso em: 5 de novembro de 2021. Citado na página 27.

MOZILLA. **What is a web server?** 2021. Disponível em: <https://developer.mozilla.org/en-US/docs/Learn/Common_questions/What_is_a_web_server>. Acesso em: 5 de novembro de 2021. Citado na página 25.

MOZILLA. **What is CSS?** 2021. Disponível em: <https://developer.mozilla.org/en-US/docs/Learn/CSS/First_steps/What_is_CSS>. Acesso em: 24 de novembro de 2021. Citado na página 24.

MOZILLA. **What is JavaScript**. 2021. Disponível em: <https://developer.mozilla.org/en-US/docs/Learn/JavaScript/First_steps/What_is_JavaScript>. Acesso em: 5 de novembro de 2021. Citado na página 24.

MUNDO EDUCAçãO. **Ponte de Wheatstone**. 2021. Disponível em: <https://mundoeducacao.uol.com.br/fisica/ponte-wheatstone.htm>. Acesso em: 5 de novembro de 2021. Citado na página 20.

MóDULO HX711. **HX711**. 2021. Disponível em: <http://blog.baudaeletronica.com.br/conversor-hx711-para-balanca-eletronica/>. Acesso em: 8 de novembro de 2021. Citado na página 33.

NPM, INC. **About npm**. 2021. Disponível em: <https://docs.npmjs.com/about-npm>. Acesso em: 8 de novembro de 2021. Citado na página 42.

OPENJS FOUNDATION. **Write middleware for use in Express apps**. 2017. Disponível em: <https://expressjs.com/en/guide/writing-middleware.html>. Acesso em: 5 de novembro de 2021. Citado na página 26.

OPENJS FOUNDATION. **About Node.js**. 2021. Disponível em: <https://nodejs.org/en/about/>. Acesso em: 5 de novembro de 2021. Citado na página 25.

OPENJS FOUNDATION. **cors**. 2021. Disponível em: <https://expressjs.com/en/resources/middleware/cors.html>. Acesso em: 8 de novembro de 2021. Citado na página 52.

PETZ. **Acne felina: o que é, causas e tratamento**. 2020. Disponível em: <https://www.petz.com.br/blog/bem-estar/acne-felina/>. Acesso em: 24 de novembro de 2021. Citado na página 60.

PETZ. **Brasil já tem mais pets que crianças**. 2021. Disponível em: <https://blog.dotpet.com.br/>. Acesso em: 03 de novembro de 2021. Citado na página 14.

PETZ. **Descubra como calcular a quantidade de ração para gato**. 2021. Disponível em: <https://www.petz.com.br/blog/nutricao/quantidade-de-racao-para-gato/>. Acesso em: 8 de novembro de 2021. Citado na página 45.

PETZ. **Levar gato para viajar: cuidados com seu pet durante as férias**. 2021. Disponível em: <https://www.petz.com.br/>. Acesso em: 03 de novembro de 2021. Citado na página 15.

PRINTED CIRCUITS. **What is a Printed Circuit Board (PCB)?** 2021. Disponível em: <https://www.printedcircuits.com/what-is-a-pcb/>. Acesso em: 5 de novembro de 2021. Citado na página 22.

PROTOSUPPLIES. **DC-DC STEP-DOWN CONVERTER OVERVIEW**. 2021. Disponível em: <https://protosupplies.com/dc-dc-step-down-converter-overview/>. Acesso em: 5 de novembro de 2021. Citado na página 22.

PYTHON SOFTWARE FOUNDATION. **Applications for Python**. 2021. Disponível em: <https://www.python.org/about/apps/>. Acesso em: 5 de novembro de 2021. Citado na página 23.

PYTHON SOFTWARE FOUNDATION. **General Python FAQ**. 2021. Disponível em: <https://docs.python.org/3/faq/general.html>. Acesso em: 5 de novembro de 2021. Citado na página 23.

RASPBERRY PI. **Raspberry Pi Hardware**. 2021. Disponível em: <https://www.raspberrypi.com/documentation/computers/raspberry-pi.html>. Acesso em: 8 de novembro de 2021. Citado 2 vezes nas páginas 28 e 29.

RASPBERRY PI FOUNDATION. **Scheduling Tasks With Cron**. 2021. Disponível em: <https://www.raspberrypi.com/documentation/computers/using_linux.html>. Acesso em: 8 de novembro de 2021. Citado na página 56.

RASPBERRY PI FOUNDATION. **Welcome to Raspbian**. 2021. Disponível em: <https://www.raspbian.org/FrontPage>. Acesso em: 8 de novembro de 2021. Citado na página 54.

RPIMOTORLIB. **Bipolar Nema11 Stepper motor with DRV8825 Driver Carrier**. 2021. Disponível em: <https://github.com/gavinlyonsrepo/RpiMotorLib/blob/master/Documentation/Nema11DRV8825.md>. Acesso em: 8 de novembro de 2021. Citado na página 56.

SCIENCEDIRECT. **MUX – Multiplexador**. 2015. Disponível em: <https://www.embarcados.com.br/mux/>. Acesso em: 5 de novembro de 2021. Citado na página 21.

SCIENCEDIRECT. **Analog-to-Digital Converter**. 2021. Disponível em: <https://www.sciencedirect.com/topics/engineering/analog-to-digital-converter>. Acesso em: 5 de novembro de 2021. Citado na página 21.

SOLIDWORKS. **What is SOLIDWORKS?** 2019. Disponível em: <https://www.captechu.edu/blog/solidworks-mechatronics-design-and-engineering-program>. Acesso em: 5 de novembro de 2021. Citado na página 19.

TANIA RASCIA. **React Tutorial: An Overview and Walkthrough**. 2018. Disponível em: <https://www.taniarascia.com/getting-started-with-react/>. Acesso em: 5 de novembro de 2021. Citado na página 25.

TECHOPEDIA. **Single-Board Computer (SBC)**. 2017. Disponível em: <https://www.techopedia.com/definition/9266/single-board-computer-sbc>. Acesso em: 5 de novembro de 2021. Citado na página 22.

TECNOBLOG. **Como funciona uma impressora 3D**. 2018. Disponível em: <https://tecnoblog.net/240402/como-funciona-impressora-3d/>. Acesso em: 5 de novembro de 2021. Citado na página 19.

TECNOBLOG. **O que é um sistema operacional?** 2019. Disponível em: <https://tecnoblog.net/303055/o-que-e-um-sistema-operacional/>. Acesso em: 5 de novembro de 2021. Citado na página 23.

WOTIOM. **STEPPER MOTOR DATASHEET**. 2021. Disponível em: <shorturl.at/berDU>. Acesso em: 8 de novembro de 2021. Citado na página 29.

YUP. **Yup**. 2021. Disponível em: <https://www.npmjs.com/package/yup>. Acesso em: 8 de novembro de 2021. Citado na página 44.