INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DE MINAS GERAIS – *CAMPUS* FORMIGA
CURSO TÉCNICO INTEGRADO EM INFORMÁTICA
PROJETO ORIENTADO DE CURSO

GABRIELA COSTA DA CRUZ

PROTÓTIPO DE ALIMENTADOR AUTOMÁTICO DE PETS

FORMIGA-MG
2018

GABRIELA COSTA DA CRUZ

PROTÓTIPO DE ALIMENTADOR AUTOMÁTICO DE PETS

Trabalho de finalização de curso apresentado ao Conselho de Curso em informática do IFMG Campus Formiga, como parte dos requisitos para obtenção do diploma de técnica em informática.

Orientador: Otávio de Souza Martins Gomes.

FORMIGA-MG

2018

Dedico este trabalho aos amigos
e familiares que estiveram sempre presentes
em minha vida entendendo minha ausência
e de alguma forma me ajudando a acreditar
que eu era capaz de finalizar meu curso.

AGRADECIMENTOS

Agradeço primeiramente a Deus, pela minha vida e por tudo que já conquistei até o momento presente. Por me dar cada vez mais forças para superar o que vem pela frente e saúde para prosseguir caminhando de pé.

Aos meus pais, que estiveram sempre presentes me ajudando sempre que possível, me apoiando e dando conselhos em todas minhas decisões. Obrigada pelo tempo dedicado à minha educação que nunca deixou a desejar mesmo estudando sempre em escolas públicas foi a melhor que poderia ser dada.

Aos meus amigos Hene, Gisele, Marina, e Igor que fizeram parte da minha caminhada. Mesmo não estando todos presentes até o final, eles fizeram parte do meu curso e formaram uma segunda família devido o tempo que passamos juntos e as histórias construídas.

Ao meu orientador, Prof. Otávio, pelo tempo e paciência dedicados a mim, que permitiu a conclusão e o desenvolver deste trabalho.

E a todos que de alguma maneira, sendo direta ou indiretamente me encaminharam pelo que sou hoje e para essa tão sonhada finalização de curso, meu muito obrigada.

RESUMO

Este trabalho simula um protótipo de alimentador automático de pets utilizando *Arduino* e *Processing*. O simulador realiza o abastecimento de ração e água de forma planejada para que o animal não possua uma alimentação desregrada e seja alimentado de forma correta. Para controle do protótipo o proprietário do animal faz uso de um aplicativo desenvolvido na ferramenta no app inventor que possibilita escolher a quantidade de refeições e os horários em que o equipamento irá executar suas ações.

PALAVRAS-CHAVE: Alimentador. App inventor. Arduino.

LISTAS DE FIGURAS

LISTA DE SÍMBOLOS

C++ : Linguagem de Programação C++

CAD : Desenho assistido por computador

SUMÁRIO

INTRODUÇÃO

Assim como as pessoas, os animais sofrem de um dos maiores problemas da atualidade, que é causado pela alimentação desregrada, seja por comer em excesso, seja por comer de forma irregular. Desta maneira, uma alimentação incorreta em qualidade ou quantidade pode trazer diversos problemas para a saúde de um pet.

Em princípio, a maioria dos animais de companhia apresentam um aumento de peso e tudo indica que é devido a carência de monitoramento na quantidade de alimento servido ao animal (ASSIS, 2018). Na maioria dos casos se dá pois a ração fica disponível durante o dia todo, quando precisaria ser servida em horários planejados e em quantidade definida.

Pensando nisso, este trabalho é desenvolvido a fim de atenuar este contratempo. Criando um ambiente que simule o equipamento que servirá a ração de forma regrada e disponibilizará água quando a mesma estiver acabando. Para isso, é necessário um agendamento do horário em que a ração deve ser despejada e a quantidade da mesma. Considerando a carência de tempo, um alimentador automático faria com que os obstáculos encontrados sejam vencidos parcialmente.

Sendo assim, cães e gatos ao final deste projeto terão um simulador de protótipo de alimentador a disposição do proprietário funcionando com total acompanhamento para o animal de estimação. O proprietário terá acesso a programação, podendo modificar a quantidade de alimento ou o intervalo de tempo para reposição. e terá a comodidade da entrega do alimento e água no local destinado. Em síntese o projeto pretende controlar (de maneira simulada) a quantidade de alimento do animal, oferecendo a quantia indispensável e diminuindo a chance de doenças ou perda de ração.

1.1 OBJETIVO

O objetivo deste trabalho é o desenvolvimento de um simulador no qual um animal de estimação seja alimentado nos horários programados com a porção necessária de ração para precaver problemas de saúde ou a perda dessa ração. O simulador final contempla um *software* de controle e um simulador de alimentador automático para pets, possibilitando que o usuário possa escolher os horários e a quantidade de ração nos quais esse simulador irá executar suas ações. Foi utilizado o *App inventor* para a criação de um aplicativo e a plataforma de prototipação Arduino juntamente com o *Processing* a fim de dispensar a montagem total do projeto.

1.2 JUSTIFICATIVA

Pensando que diversos donos de animais apresentam falta tempo para alimentar os seus pets, pois possuem muitos compromissos como: trabalhar ou estudar. Faz-se necessário que o abastecimento da ração seja efetuada de forma constante e planejada para que o animal não possua uma alimentação desregrada e tenha uma boa saúde. Deste modo, é fundamental que ele não coma em excesso nem fique por muito tempo sem se alimentar. Surge assim, a ideia de desenvolver um equipamento que solucionará todos os problema apresentados e dará tranquilidade ao dono do animal e saúde ao animalzinho, esse projeto contribui com a construção de um simulador para esse protótipo.

2 REFERENCIAL TEÓRICO

Foram pesquisadas diferentes maneiras de criar o simulador. Inicialmente, foi visto um protótipo criado por três estudantes do ensino médio com materiais de baixo custo, formado por um galão de água onde a ração fica reservada e um recipiente que recebe a quantidade programada por um sistema. Em um programa feito no computador é configurado quantos gramas de ração devem ser liberados, quando e, com qual frequência e é feito um registro dos hábitos do animal no computador, bem como, os horários que o animal costuma comer (SOUZA, 2016). Vide Figura 1.

Figura 1 – Protótipo de alimentador (Tipo 1)

Fonte: (GUEDES, 2016)

Analisando outro protótipo, é possível notar que o criador também utilizou galões de água para reservar a ração e a água, mas, neste caso, ele utilizou de canos de pvc para ligar o reservatório aos potes localizados no fim do trajeto. Ele fez uso do *Arduino* juntamente com um relógio para guardar as informações dos horários da distribuição da ração, mostrado na Figura 2. Após todos os dados serem inseridos, o relógio se inicia checando se os horários descritos na agenda

são os mesmos da hora atual, quando eles são iguais é ligado o motor da ração, já no caso da água, o *Arduino* checa continuamente se é necessário reabastecê-la para que seu nível não diminua (SANTOS, 2015).

Figura 2– Protótipo de alimentador (Tipo 2)
Fonte: (SANTOS, 2015)

Atualmente, no mercado é possível encontrar várias máquinas capazes de realizar todas essas funções, um exemplo disso é a *Hoison* Babá Robô mostrada na Figura 3. Segundo o site *Petlove,* um site de vendas, a *Hoison* possui capacidade para até dois quilos de ração e um aplicativo para que o proprietário programe seus horários. Com ela é possível que o possuinte grave sua voz para se comunicar com o pet e monitore o animal através de uma câmera. Neste site é possível comprá-la por R$1.349,90, um preço consideravelmente alto em comparação com os protótipos apresentados, já que um protótipo similar pode ser feito com em média R$ 300,00.

Figura 3 – Máquina *Hoison* Babá Robô
Fonte: (https://goo.gl/NXsc)

3 MATERIAIS E MÉTODOS

Nesta seção serão apresentados os materiais utilizados na construção de protótipos de teste para o simulador e a metodologia utilizada, bem como o estudo realizado para a implementação dos códigos.

3.1 MATERIAIS UTILIZADOS

01 Placa *Arduino* Uno
01 *Protoboard*
01 *Buzzer*
02 *Leds*
01 Resistor
01 *Display LCD*
02 Sensor ultrassônico hc-sr04
02 Potenciômetros
01 Módulo de *bluetooth* HC-06
01 Motor de passo

3.1.1 ARDUINO

A plataforma *Arduino* abrange tanto discentes iniciantes, quanto pesquisadores evoluídos no ramo da eletrônica e vem ganhando progressivamente seu espaço dentro de universidades acadêmicas. Elaborada a fim de que todos possam ter em mãos um dispositivo simples de programar e com o preço atingível. A lista de possibilidades do que se fazer com ela é praticamente infinita, já que, a praticidade oferecida por ela é enorme podendo ser automatizada em casa, em sala de aula, ou até mesmo em um carro. Dependendo da criatividade de quem programa pode ser uma ferramenta poderosa de criação ou aperfeiçoamento.

A placa possui uma linguagem híbrida de C/C++ e consegue ser conectada rapidamente por via de uma IDE que pode ser alcançada no site oficial do próprio *Arduino*, permitindo assim a praticidade no desenvolver da programação. Normalmente conectado a diferentes componentes eletrônicos e a uma protoboard, podendo depois de programado ser usado de forma autônoma (THOMSEN, 2014). Dentre as possibilidades, foi escolhido a placa de *Arduino* UNO, já que ela possui o número de portas de entrada e saída necessárias para a aplicação, mostrado na Figura 4. A escolha foi justamente pela praticidade na hora de usar e pela experiência em sala de aula. Desta forma, foi definido que a plataforma de *Arduino* seria ideal para o projeto em questão.

Figura 4 – *Arduino* Uno

Fonte: (https://goo.gl/iU6L5R)

3.1.2 APP INVENTOR

De acordo com o site Tecmundo (2018), o *Arduino* é conhecido por ser fácil de programar e para iniciantes, o *App Inventor* é recomendado a quem planeja se iniciar na programação Android. Ele permite o desenvolvimento de aplicativos utilizando um navegador da Web, juntamente com um telefone *Android* baixando o aplicativo MIT AI2 Companion que permite testar as criações. Ademais, pode ser utilizado também o emulador que necessita ser instalado no computador de acordo com o sistema operacional *Windows, Mac OS X* ou *Linux*. De acordo com Tecmundo (2018),

> "Um aplicativo feito no *App Inventor* pode servir propósitos específicos para o seu uso pessoal ou profissional. Pode preencher a lacuna em que os aplicativos de terceiros pecavam. E pode levar usuários comuns a se interessarem em começar a programar para a plataforma, elevando assim o nível de qualidade dos aplicativos que são feitos e a quantidade de *softwares* disponíveis para a plataforma."

Tendo como base o que foi dito, o *App Inventor* pode ser usado tanto para iniciantes no mundo educacional, já que possui uma linguagem fácil e acessível, quando para profissionais.

3.1.3 TINKERCAD

Tinkercad é um simulador virtual online e gratuito, para quem não possui a placa de *Arduino* física em mãos ou para quem quer realizar toda a montagem do projeto virtualmente sem ter problemas de conexão ou mal contato. A ferramenta traz consigo diversos componentes que podem ser usados de diferentes formas dependendo da programação a ser feita. Ademais ela possui circuitos pré-montados para facilitar ainda mais a vida do programador que pode programar tanto em blocos como em texto. Além de simulador é uma ferramenta de *design*

de modelos 3D em CAD que vem tornando a impressão 3D mais acessível (PRADO, 2018).

3.1.4 PROCESSING

O *Processing* é um software desenvolvido por ex-membros do MIT media Lab com objetivo de educar pessoas no ramo da programação, não só de um computador mas também em um contexto de artes (GOMES, 2018). Desta forma, ele ficou conhecido como um caderno de esboços de *software* flexíveis para aprendizado (GIZMODO, 2008). Acrescenta-se também que, a ferramenta permite a integração com a plataforma *Arduino* o que possibilita seu uso no projeto.

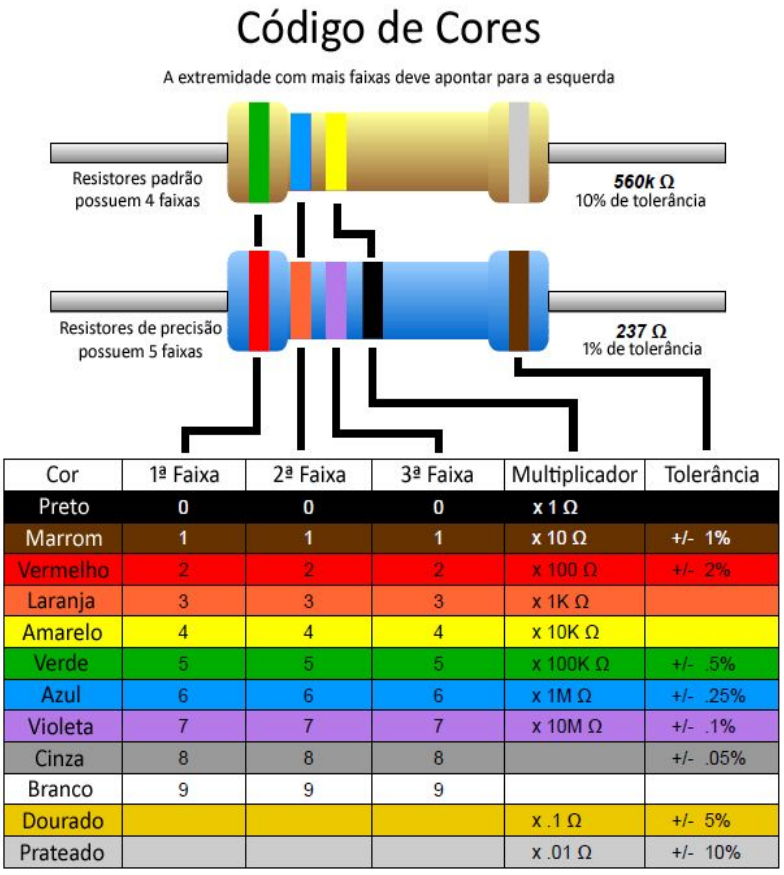
3.1.5 MÓDULO DE CONEXÃO BLUETOOTH

Como o modelo de placa *Arduino* utilizada no trabalho não possui um módulo de *Bluetooth* integrado, será necessário a utilização do módulo de *Bluetooth* HC-06 que conectado a placa possibilitará a conexão *Bluetooth* com a placa *Arduino* e o *App Inventor*. De acordo com *Buildbot* (2018) uma loja de robótica,

> "O alcance do módulo segue o padrão da comunicação *bluetooth*, que é de aproximadamente 10 metros. Esse módulo funciona apenas em modo *slave* (escravo), ou seja, ele permite que outros dispositivos se conectem à ele, mas não permite que ele próprio se conecte à outros dispositivos *bluetooth*."

3.1.6 RESISTOR

Os resistores são usados mormente a fim de limitar correntes e evitar a queima de componentes como os *leds* e os *displays.* Para usá-lo, é necessário saber o valor de suas cores. Vide Figura 5.



Cor	1ª Faixa	2ª Faixa	3ª Faixa	Multiplicador	Tolerância
Preto	0	0	0	x 1 Ω	
Marrom	1	1	1	x 10 Ω	+/- 1%
Vermelho	2	2	2	x 100 Ω	+/- 2%
Laranja	3	3	3	x 1K Ω	
Amarelo	4	4	4	x 10K Ω	
Verde	5	5	5	x 100K Ω	+/- .5%
Azul	6	6	6	x 1M Ω	+/- .25%
Violeta	7	7	7	x 10M Ω	+/- .1%
Cinza	8	8	8		+/- .05%
Branco	9	9	9		
Dourado				x .1 Ω	+/- 5%
Prateado				x .01 Ω	+/- 10%

Figura 5 – Código de cores

Fonte:(https://goo.gl/66ZP6h)

Segundo o *Arduino* e *Cia* (2013), apesar deste método funcionar, é necessário realizar a soma dos valores de cada cor do resistor, uma maneira mais rápida e simples de saber quantos ohms ele possui, é utilizar uma calculadora online como a do site *Searching tabs,* mostrado na Figura 6.

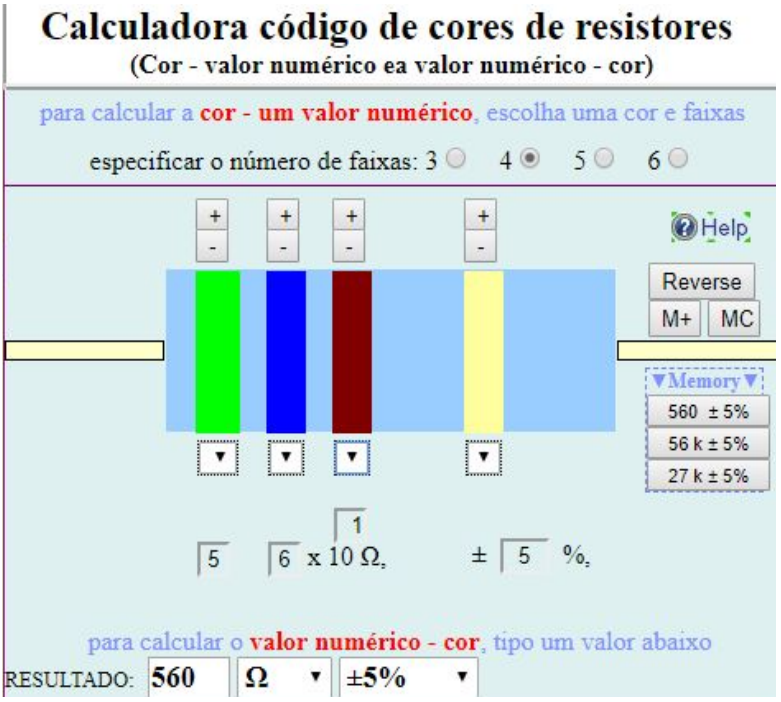


Figura 6 – Calculadora código de cores de resistores

Fonte: (http://www.searchingtabs.com/rcolor/rescolor_port.htm#cl)

Tendo em base ainda no site *Arduino* e *Cia* (2013), a calculadora pode aparentar mais difícil do que é, pois o layout é confuso olhando uma primeira vez. Mas ela é ótima, já que, possibilita especificar a quantidade de faixas e escolher as cores de cada uma, mostrando no fim, o resultado sem nenhum esforço. Além do mais, é possível fazer o caminho inverso, adicionando um número no campo resultado a página será carregada novamente e mostrará as novas cores correspondentes do resistor.

3.1.7 MOTOR DE PASSO

De acordo com Dicionário de Cambridge, motores são mecanismos que transformam energia elétrica em energia cinética para fazer uma máquina atuar. Sendo assim, segundo o site Kalarec Automação (2018), o motor de passo transforma impulsos elétricos em movimentos discretos mecânicos . Ele é usado em projetos que necessitam de movimentos precisos, já que, ele possibilita um comando através da velocidade, do ângulo de rotação e da posição. As vantagens

em se usar um motor de passo são inúmeras e como ele se posiciona em ângulos, definir seu posicionamento se torna fácil. Como ele se movimenta através do estímulo de suas bobinas, basta simplesmente diminuir ou aumentar a sua frequência e o motor se alinha rapidamente a elas.

3.1.8 LED

Segundo Mattede (2018), os *leds* são componentes polarizados e por isso necessitam ser conectados corretamente. Como ele é um emissor de luz, se conectado corretamente ele possibilita que o fluxo de energia levado nele o acenda. Por fim, ele possui dois polos, um positivo e um negativo, sendo que, o positivo se localiza na perna maior e o negativo em sua perna menor. Vide Figura 7.

Figura 7 – *LED*
Fonte: (https://goo.gl/Ftpz2G)

3.1.9 *PROTOBOARD*

Na visão do site *Robocore* (2017), a *protoboard* é um aparelho de alta importância para montagens de circuitos e para eletrônica. Ela permite a montagem desses circuitos sem necessitar a soldagem de seus componentes, tornando assim, o desenvolvedor capaz de realizar diferentes projetos em apenas

uma placa devido a sua praticidade de ser montada e desmontada rapidamente. Uma *protoboard* é dividida em duas partes, a primeira parte é a parte central onde se realiza os encaixes dos componentes a serem utilizados. A segunda parte consiste em duas linhas que se localizam nas suas laterais, nessas linhas é realizada a distribuição e alimentação elétrica positiva e negativa da placa. Vide Figura 8.

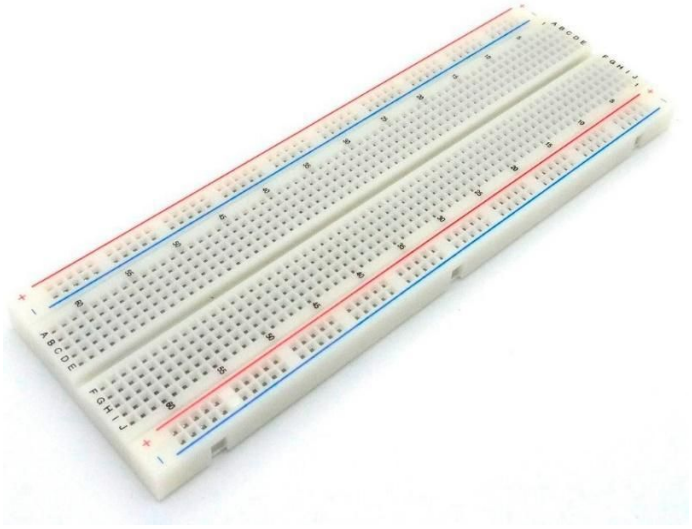


Figura 8 – *Protoboard*
Fonte: (https://goo.gl/CnkMns)

3.1.10 *DISPLAY* LCD

Displays LCD são interfaces de comunicação permitindo a interface visual do programador com a máquina. Ela é muito usada, já que, é bem simples de ser usada e seu preço é bem acessível (CIRIACO, 2009). Existem diferentes modelos e os mais usados em *Arduino* são o 16×2 (16 caracteres x 2 linhas) ou 20×4 (20 caracteres x 4 linhas). Segundo o blog Eletrogate (2017),

> "O mostrador é formado de duas placas acrílicas transparentes. Entre essas placas está o cristal líquido. Esse cristal líquido altera o seu comportamento cristalino, dependendo da tensão aplicada entre ele. Os displays, como dá para ver, são formados de vários pontinhos. Cada pontinho pode ficar claro ou escuro, dependendo da polarização da eletricidade de cada um. Sob as placas transparentes, existem uma

> matriz invisível de conexões que controlam todos esses pontinhos. Quem faz isso, são os chips controladores que ficam por trás do *display*."

Eles estão em quase todos os equipamentos de comunicação pessoa e máquina, e neste projeto será utilizado o 16x2 como mostrado na Figura 9.

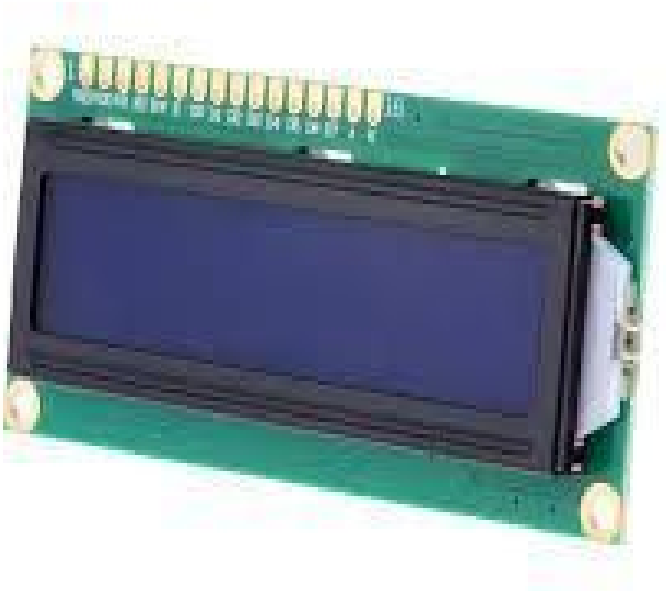


Figura 9 – *Display LCD*
Fonte: (https://goo.gl/VhBDQY)

3.1.11 SENSOR DE FLUXO

Segundo a USINAINFO (2016) um blog de ferramentas e eletrônica, o sensor de fluxo de água é apropriado para: a verificação de consumo de água, o intervalo de tempo em que há recebimento desta água e a quantidade em litros por minuto da mesma. Na maioria das vezes ele trabalha junto com o *display* LCD, que irá passar essas informações ao programador. Conforme dito por USINAINFO (2016) ,

> "O sensor de fluxo de água trabalha com uma turbina, no seu interior existe um tipo de hélice que ao receber o atrito da água gira, junto a esta hélice temos fixado um íman, e paralelamente ao íman temos um sensor de efeito Hall, esse sensor detecta cada volta dada pelo íman."

E assim é possível a realização de diferentes projetos relacionados a verificação de água pelo sensor de fluxo. Ele pode ser usado com diferentes plataformas de prototipagem e *Arduino* é uma delas.

3.1.12 POTENCIÔMETRO

Segundo o site Como Fazer as Coisas (2014), o potenciômetro é um componente eletrônico que normalmente possui três terminais e um eixo giratório que permite o controle de fluxo de sua corrente elétrica. A sua resistência é medida em ohms e o valor informado nele é a sua resistência máxima. De acordo com o Como Fazer as Coisas (2014),

> "Os potenciômetros são utilizados em circuitos de baixa tensão e corrente, devido a sua baixa potência que normalmente vai de 0,25w a 1w. Se você necessitar de um pouco mais de potência pode usar um potenciômetro de fio, que pode suportar comumente 4w, ou um reostato."

Existem diferentes formatos de potenciômetros, mas o mais comum e utilizado neste projeto é o de eixo giratório. Vide Figura 10.

Figura 10 – Potenciômetro
Fonte:(https://goo.gl/CCBmYS)

3.2 METODOLOGIA

Primeiramente, foi feita uma reunião informal entre o orientador e a orientada, para que os dois realizassem o planejamento de um esboço inicial para ser seguido (*brainstorm*[1]). As ideias foram desenhadas no quadro a fim de um maior entendimento com a ilustração. A Figura 11 mostra exatamente isso, nela é possível notar que foi definido em primeira instância que a orientada faria dois reservatórios, um para água e outro para ração, os dois seriam ligados ao *arduino* e executariam suas funções.

No caso da ração, teria um motor de passo que se encarregaria de reabastecer a ração quando o sensor determinasse que a mesma estaria acabando, com a quantidade e horários escolhidos pelo dono do animal no aplicativo feito no *App Inventor*. Já no caso da água ela seria reabastecida sempre que o sensor determinasse um nível baixo de sua quantidade. Foi feita uma pesquisa com Médicos Veterinários da cidade de Formiga e eles determinaram que o recomendado para a alimentação de cada animal, é 2% de seu peso, sendo assim, o aplicativo já foi programado pensando nessa porcentagem. Uma observação feita pelos veterinários é que a porcentagem pode variar quando o animal é filhote ou necessita de uma dieta regrada.

[1] SIGNIFICADO. **Significado de Brainstorming**. Disponível em: <https://www.significados.com.br/brainstorming/>. Acesso em: 20 nov. 2018.

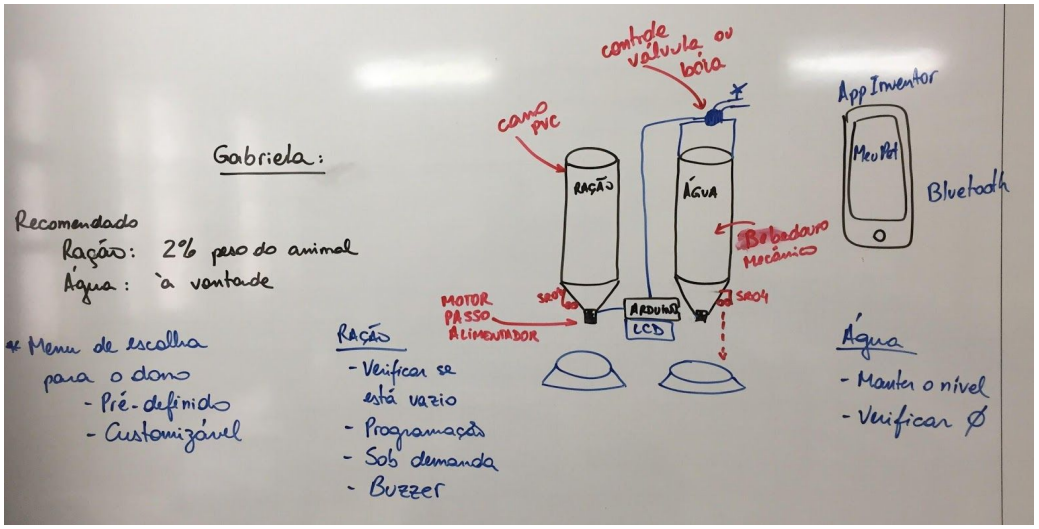


Figura 11- Esboço inicial

Fonte: A autora.

3.2.1 PESQUISA E IMPLEMENTAÇÃO

Inicialmente foram oferecidos pelo professor orientador dois cursos de *Arduino*, sendo eles "*Arduino*- curso completo e prático[2]" e "Aprenda *Arduino* com uso de simulador[3]". Os cursos oferecidos são da *Udemy*[4] um site educativo, que é um *marketplace* de ensino e aprendizado online com diversos cursos e milhões de alunos. A *Udemy* possui diferentes cursos de programação e Arduino está entre eles.

O curso Aprenda *Arduino* com uso de simulador do instrutor Roni Shigueta, mostrou como se cria projetos em plataformas online, para alunos que não possuem a placa física em mãos. Tendo como objetivo capacitar o aluno no uso de um simulador para criar projetos em Arduino, ajudando o aluno conectar e receber dados de sensores e enviar dados aos atuadores. Realizar acionamentos usando controle remoto e muito mais. O curso possui 45 aulas de aproximadamente 6 minutos cada resultando no total de 4,5 horas de aprendizado.

---

[2] MAZUCO, Vitor. Arduino - **Curso Completo e Prático**. Disponível em: <https://www.udemy.com/arduino-na-pratica/learn/v4/overview>. Acesso em: 20 nov. 2018.

[3] SHIGUETA, Roni. **Aprenda Arduino com uso de simulador**. Disponível em: <https://www.udemy.com/aprenda-arduino-com-uso-de-simulador/learn/v4/overview>. Acesso em: 20 nov. 2018.

[4] Disponível em : < https://www.udemy.com/mobile/ipad/ >. Acesso em : 20 nov. 2018.

O curso tem o propósito de instruir os alunos nos diferentes conteúdos que o *Arduino* proporciona. O instrutor repassou seu conhecimento sobre as características dos resistores e mostrou como identificar os componentes da placa e como usar os *LEDs, buzzers*, sensores de temperatura, *sensor de ultra-som*, *LED RGB,* sensor de movimento, *micro-servo*, *display LCD*, controle remoto, motores DC*, relés*. Mostrou também como realizar a transferência do código do simulador para a placa *Arduino* física, e como acionar as lâmpadas com *relés* utilizando controle remoto. E por fim apresentou o sistema de temperatura que pode ser utilizado em diferentes aplicações. Ao concluir o curso, a Udemy disponibilizou o diploma certificando sua finalização. Vide Figura 12

Figura 12- Certificado de Conclusão Curso 1

Fonte: Udemy.

Já o curso *Arduino* - Curso Completo e Prático do instrutor Vitor Manzuco tem como foco não apenas quem procura conhecimentos básicos em Arduino, mas também quem busca conhecimentos avançados. É um curso completo que promete ajudar e capacitar qualquer aluno criar seus projetos. O curso mostrou como surgiu o *Arduino,* os primeiros passos para a programação, como instalar o IDE e também como conectar uma placa fazendo uso de um computador. Mostrou de forma detalhada diversos projetos com *LED*s, sensores variados, sons, motores, servo motores e projetos especiais que unem todos os projetos em

apenas um. O curso possui 88 aulas de aproximadamente sete minutos e meio cada resultando no total de 11 horas de aprendizado.

O curso tem o propósito de instruir os alunos nos diferentes conteúdos que o *Arduino* proporciona, sendo não apenas um curso-palestra, mas também tendo objetivo de um aprendizado prático. O instrutor repassou seu conhecimento sobre as características dos resistores e os mostrou como criar projetos com ferramentas acessíveis pensando em quem não possui condições de possuir controladores sofisticados tendo como objetivo fazer com que seja fácil o aprendizado tanto amador ou profissionais. Ao finalizar o curso, a Udemy disponibilizou o diploma certificando conclusão. Vide Figura 13.

Figura 13- Certificado de Conclusão Curso 2

Fonte: Udemy.

4 DESENVOLVIMENTO

4.1. IMPLEMENTAÇÃO NO *TINKERCAD*

Inicialmente, foi montado o circuito, conforme Figura 14. Foram utilizados dois sensores, um para medir a distância da ração e outro da água, dois *leds* representando os motores de passo, um *buzzer* que é acionado quando a ração é despejada e um *display lcd.* Os sensores trabalham junto com o display *lcd* medindo a distância e mandando para a tela do display. Quando a distância for pequena, o *led* acende mostrando que o motor está girando, e quando o recipiente está cheio o *buzzer* é acionado para alertar o animal que a ração está a sua disposição.

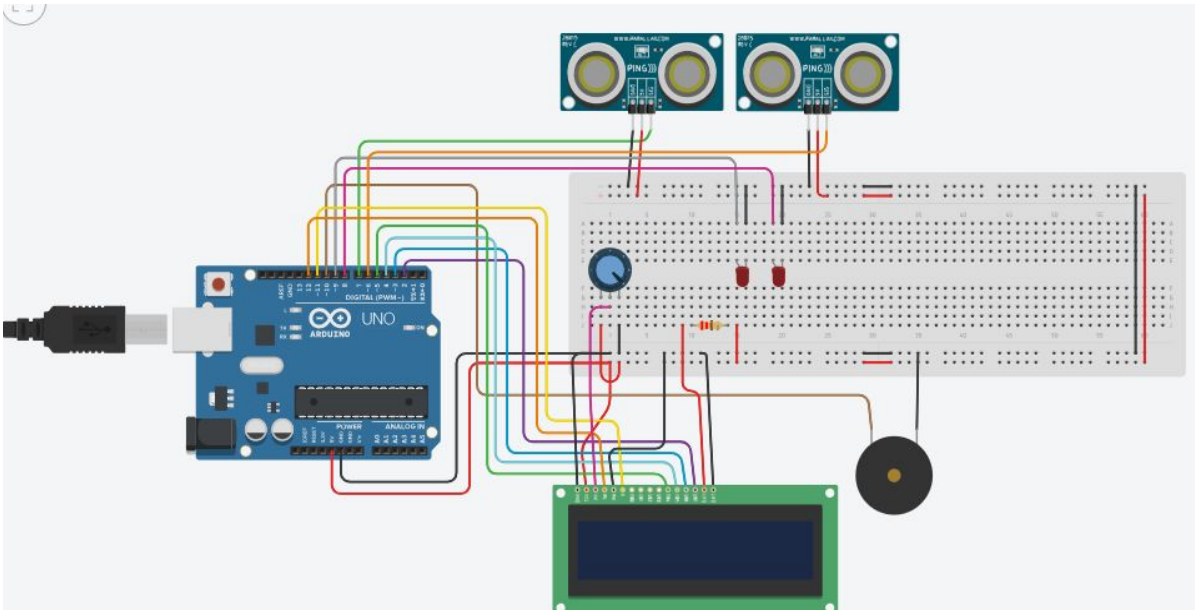


Figura 14– Circuito *tinkercad*

Fonte: A autora.

Após a montagem do circuito na plataforma online, iniciou-se o desenvolvimento dos códigos. De início, foi necessário incluir a biblioteca usada para o *display lcd* e criar uma constante para o *buzzer*. Na Figura 15 é mostrado

como a distância foi calculada e como o *ping* é preparado por um pulso *HIGH* de 2 microssegundos. Depois disso é feito um pulso *LOW* para que o pulso *HIGH* seja seguro e não aconteça falhas. Em continuidade é usado o mesmo pino para ler o sinal do *PING* em que *HIGH* é representado pela duração em microssegundos que levará desde o envio do som até que o eco retorne. Para o desenvolvimento dos códigos, foram utilizadas referências disponíveis em *Arduino* (2008).

```
#include <LiquidCrystal.h>

LiquidCrystal lcd(12, 11, 5, 4, 3, 2);

const int buzzer = 10;

// medir distância
long centimetrosSensor(int pingPin) {
  long duration;

  //HIGH de 2 microssegundos
  // pulso LOW evidenciando HIGH
  pinMode(pingPin, OUTPUT);
  digitalWrite(pingPin, LOW);
  delayMicroseconds(2);
  digitalWrite(pingPin, HIGH);
  delayMicroseconds(5);
  digitalWrite(pingPin, LOW);
  // Ler o sinal ping
  //envio do som até a volta do eco
  pinMode(pingPin, INPUT);
  duration = pulseIn(pingPin, HIGH);

  return duration / 29 / 2;
```

Figura 15 – Medição da distância

Fonte: A autora

Logo em seguida, é configurado os pinos do *buzzer* e dos *leds* como pinos de entrada(*INPUT*) e saída(*OUTPUT*), e é usado também o comando *begin()* que trabalha junto com a biblioteca *liquidCrysta*l configurando o modelo do *display LCD* que no caso deste trabalho é 16x2 (LEÃO, 2014). Como o projeto foi pensado para o abastecimento de alimento e água do animal o sensor de distância irá

trabalhar medindo essa distância e quando ela for menor que 50 cm será acionado o *Led* que representa o motor de passo já que a plataforma não o possui em seus componentes. O código foi implementado duas vezes já que existem dois sensores e ocorreram mudanças apenas onde foram conectados o sensor e o *led.* A Figura 16 mostra exatamente está codificação.

```
void setup(){

  pinMode(buzzer, OUTPUT);
  // 16 colunas x 2 linhas LCD
  lcd.begin(16, 2);
  //leds 8 e 9
  pinMode(8,OUTPUT);
  pinMode(9,OUTPUT);
  digitalWrite(8, LOW);
  digitalWrite(9, LOW);
}
void loop() {
  // Se no sensor a distância for menor que 50
  // é ligado o Led e o Buzzer
   if (centimetrosSensor(7) < 50) {
         digitalWrite(8, HIGH);
  //Ligando o buzzer com uma frequência de 1500 hz.
    tone(buzzer, 15);
   // se a distância for maior que 50,
   // o buzzer e o Led são desligados
  } else {
         digitalWrite(8, LOW);
    //Desligando o buzzer.
         noTone(buzzer);
  }

  if (centimetrosSensor(6) < 50) {
         digitalWrite(9, HIGH);
     tone(buzzer, 15);
  } else {
         digitalWrite(9, LOW);
    noTone(buzzer);
  }
```

Figura 16 – Configuração de pinos e funcionamento do sensor

Fonte: A autora

Por fim, foi feito um código para jogar a distância em que a ração e a água estão para o *LCD.* Foram usados os comandos *lcd.setCursor* e *lcd.print*, que também trabalham junto com a biblioteca *liquidCrystal*. Segundo o Baú da eletrônica (2017), o *setCursor* posiciona o cursor na linha e coluna indicada no comando e o *lcd.print* escreve o dado no display. Por último, no comando de condição *if*, se os centímetros do sensor forem menor que cinquenta, o *led* acende, se não forem ele permanece apagado ou apaga. Vide Figura 17.

```
//ração
  // jogar a distância do objeto para o LCD
  lcd.setCursor(0, 0);
  lcd.print("Distancia");
  lcd.setCursor(0, 1);
  lcd.print(centimetrosSensor(7));

  if (centimetrosSensor(7) < 50) {
        digitalWrite(8, HIGH);
  } else {
        digitalWrite(8, LOW);
  }*/

 //agua

  lcd.setCursor(0, 0);
  lcd.print("Distancia");
  lcd.setCursor(0, 1);
  lcd.print(centimetrosSensor(6));

  if (centimetrosSensor(6) < 50) {
        digitalWrite(9, HIGH);
  } else {
        digitalWrite(9, LOW);
  }
}
```

Figura 17 – Código ração e água *Arduino*

Fonte: A autora

4.2 IMPLEMENTAÇÃO NO *ARDUINO*

4.2.1 IMPLEMENTAÇÃO *ARDUINO* E *APP INVENTOR*

Para que a soma da quantidade de alimentação que o animal terá fosse completada, foi necessário a inversão dos valores do *App Inventor* para letras já que no aplicativo não foi possível o envio de números. Ao enviar um número *do App Inventor* para o *Arduino,* as informações passadas eram perdidas e apenas com letras essa conexão foi exata. Sendo assim, foram usadas as letras “a”, “b” e “c”, substituindo respectivamente os valores 2,3 e 4.

Na codificação do Arduino, foi usado um laço de repetição *while,* que possui a função de iniciar sua função assim que alguma informação chegar até ele (quando a informação for igual a “a”, “b” ou “c”). Ele começará a execução lendo a quantidade de KG adicionado no aplicativo, e como as letras equivalem a números, foi preciso desenvolver uma cláusula condicional para cada uma delas, especificando o valor de cada letra com a ajuda da variável “op”. Após isso, é realizada a soma da quantidade de KG que o animal possui. Este valor é multiplicado por 0,02, que é a quantidade necessária de alimentação que ele necessita. Este valor deve ser dividido pela opção do número de vezes que ele se alimentará por dia. Por fim, como o valor da variável “recebido” foi declarada como um *char*, foi necessário realizar a inversão desse valor para *int* novamente e segundo o site *Arduino* (2008), basta subtrair o valor adicionado por 48 já que um número em *char* equivale a 49. O código a seguir pode ser analisado na Figura 18.

```
int ledPin = 13;

int val_pot01 = 0;
int val_pot02 = 0;

char recebido;
int txtKg = 0;
int op = 0;

float total;
```

```
void setup() {
   pinMode(ledPin, OUTPUT);
   Serial.begin(9600);
}

void loop() {

  while (Serial.available()){

          recebido = Serial.read();

          val_pot01 = analogRead(A1);
          val_pot01 = map(val_pot01, 0, 1023, 0, 5);

          val_pot02 = analogRead(A2);
          val_pot02 = map(val_pot02, 0, 1023, 0, 5);

          if (recebido=='a'){
                  op = 2;
          }
          else if (recebido=='b'){
                  op = 3;
          }
          else if (recebido=='c'){
                  op = 4;
          }
          else {
                  txtKg=(int(recebido)-48);
          }

          total = txtKg;
          total = total*0.02;
          total = total/op;

          Serial.print(total);
          Serial.print(',');
          Serial.print(val_pot01);
          Serial.print(',');
          Serial.println(val_pot02);

          delay(100);
  }
}
```

Figura 18 – Código *Arduino*

Fonte: A autora

4.2.2 IMPLEMENTAÇÃO *ARDUINO* E *PROCESSING*

A implementação do *Processing* no *Arduino* contou com dois potenciômetros, um para aumentar e diminuir a quantidade da ração e outro para

a água. Primeiramente, foi necessário a criação algumas variáveis que servirão para guardar os valores da quantidade da ração e da água. Logo depois, como o potenciômetro permite que seu sinal varie de 0 até 1023 foi feita uma codificação com base no site Arduino (2008) para compatibilizar o sinal fazendo com que ele varie apenas entre 0 e 5. Neste caso, quando o reservatório estiver em estado crítico ele estará em 0, e quando ele estiver em 5 ele estará cheio e quando estiver entre esses valores, ele estará com meia carga . Vide Figura 19.

```
int ledPin = 13;

int val_pot01 = 0;
int val_pot02 = 0;

int recebido;

int txtKg = 0;
int op = 0;

float total;

void setup() {
   pinMode(ledPin, OUTPUT);
   Serial.begin(9600);
}

void loop() {

          val_pot01 = analogRead(A1);
          val_pot01 = map(val_pot01, 0, 1023, 0, 5);

          val_pot02 = analogRead(A2);
          val_pot02 = map(val_pot02, 0, 1023, 0, 5);

          total = 4.0; //txtKg
          total = total*0.02;
          total = total/op;

       Serial.print(total);
       Serial.print(',');
       Serial.print(val_pot01);
       Serial.print(',');
       Serial.println(val_pot02);
          delay(2000);
}
```

Figura 19 – Codificação potenciômetro

Fonte: A autora

4.3 IMPLEMENTAÇÃO NO *APP INVENTOR*

4.3.1 CRIAÇÃO DA LOGO NO *APP INVENTOR*

Para a elaboração do logo, foi usado o *FreeLogoServices*, um site online que cria diferentes logos gratuitos. Ele disponibiliza diversos modelos ao empreendedor e é usado atualmente por mais de 25 milhões de empresas no mundo já que o logo possui a responsabilidade de remeter à conexão entre o aplicativo e o protótipo deixando claro sua função. Vide Figura 20.

Figura 20 – Logo *AppInventor*
Fonte: A autora

Para a elaboração da interface, foi levado em conta a bagagem ganha ao longo do curso, as matérias de POO e Banco de Dados, ministradas respectivamente pelos professores, Paloma Oliveira e Fernando Paim do IFMG Campus Formiga. Vide Figura 21.

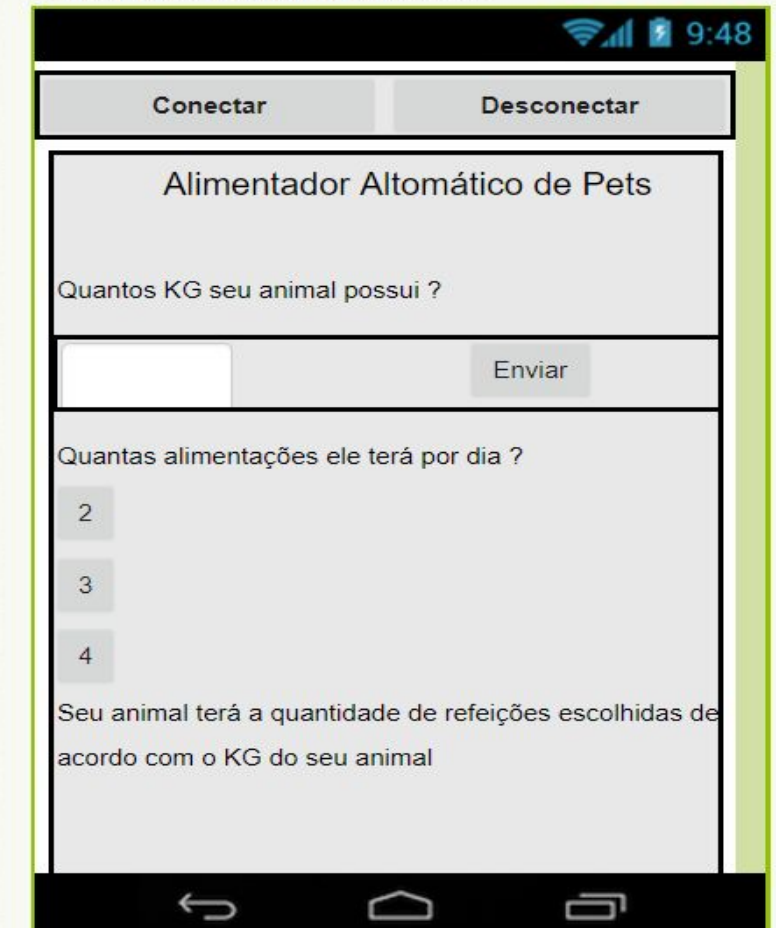


Figura 21 – Interface *AppInventor*

Fonte: A autora

4.3.2 IMPLEMENTAÇÃO DO *BLUETOOTH* NO APP INVENTOR

Toda implementação do *bluetooth* no app inventor foi embasada no vídeo “Ligando um *led* com *AppInventor, Arduino* e módulo *Bluetooth*” visto no *Youtube* (2014). A implementação se inicializa com uma verificação para saber se o *bluetooth* está ativo ou não. Se não estiver ativo é chamado duas mensagens, uma de ok e outra de sair, quando clicado nas mensagens o evento *AfterChoosing* é chamado, se a pessoa clicar em ok ela irá continuar no aplicativo e uma mensagem à avisará para ligar o *bluetooth.* Caso clicar em sair o aplicativo será fechado. Se a pessoa continuar no aplicativo e mesmo após ler a mensagem para a ativação do *bluetooth* não ativá-lo outra mensagem é enviada. Vide Figura 22.

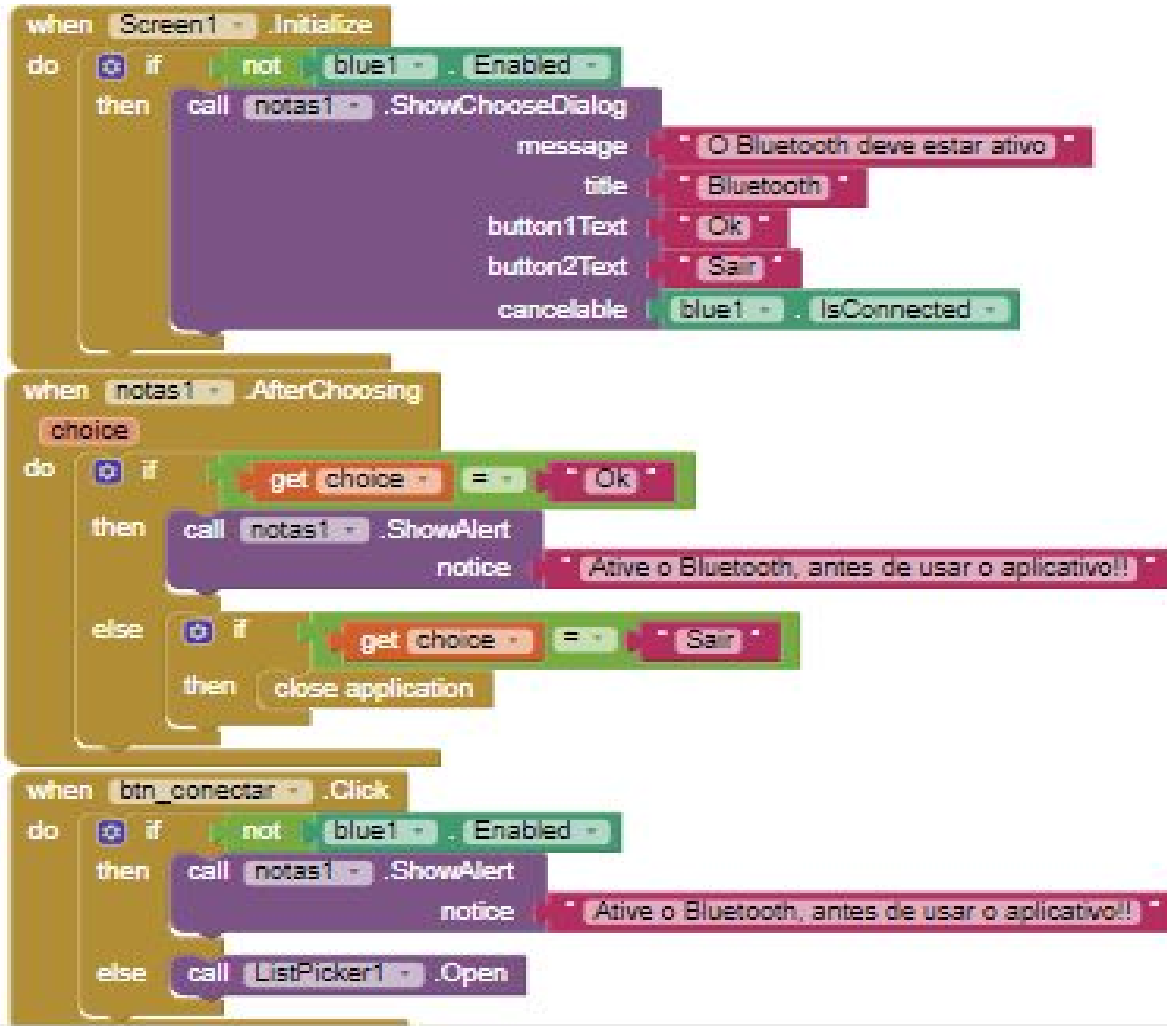


Figura 22 – Ativando *Bluetooth* 01

Fonte: A autora

Se o *bluetooth* estiver ativado, o *listPicher* é aberto e seu evento é acionado para o preenchimento da lista que corresponde ao *bluetooth*, ali será passado o endereço correspondente do *bluetooth* como mostrado na Figura 23.

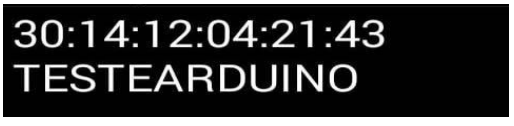


Figura 23 – *ListPicher*

Fonte: A autora

Após clicado em algum dos endereços é verificado a conexão, se ele não estiver conectado aparece uma mensagem de erro na tentativa de se conectar, e

se a conexão der certo acenderá o *led* da placa *Arduino* que está representado pelo “A” já foi configurado no código do IDE no *Arduino.* Se a pessoa quiser se desconectar é mandado um “a” e o *led* se apaga. Vide Figura 24.

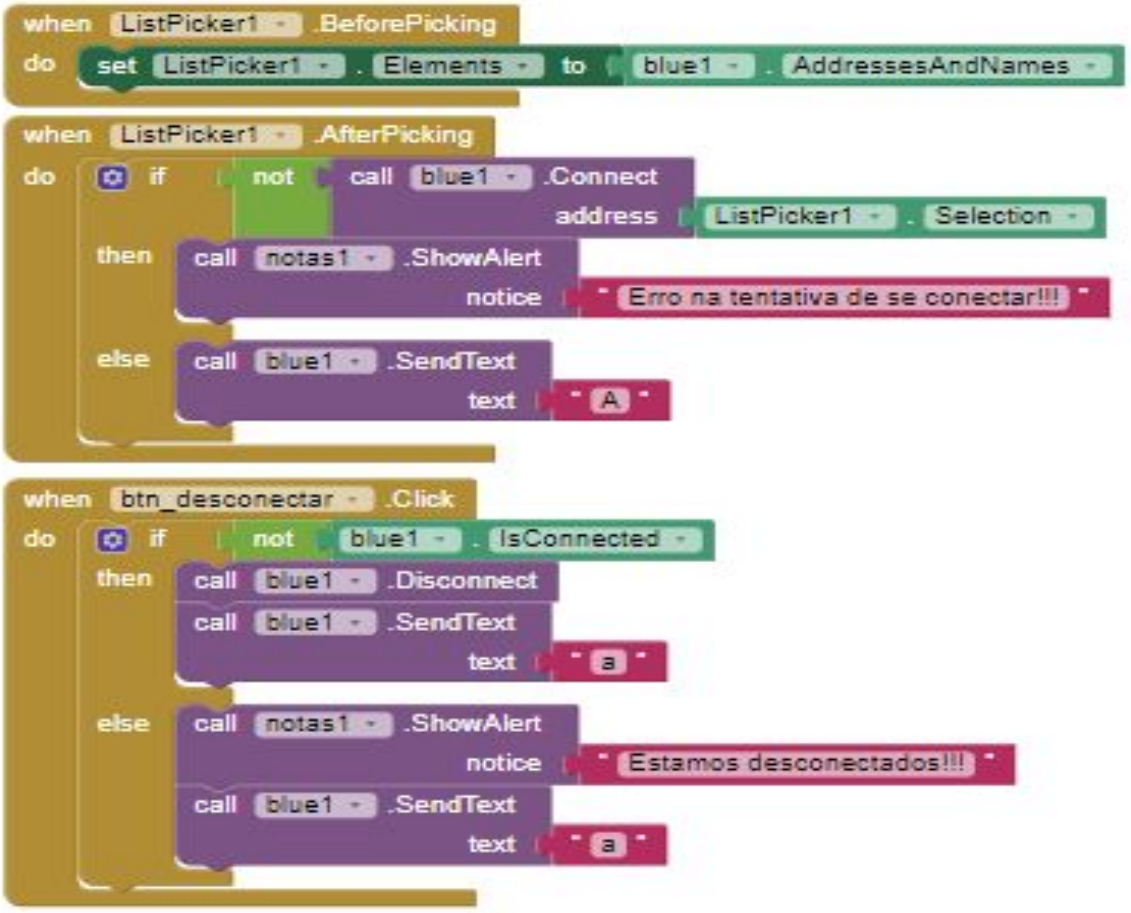


Figura 24 – Ativando Bluetooth 02

Fonte: A autora

O circuito do projeto é apresentado na Figura 25.

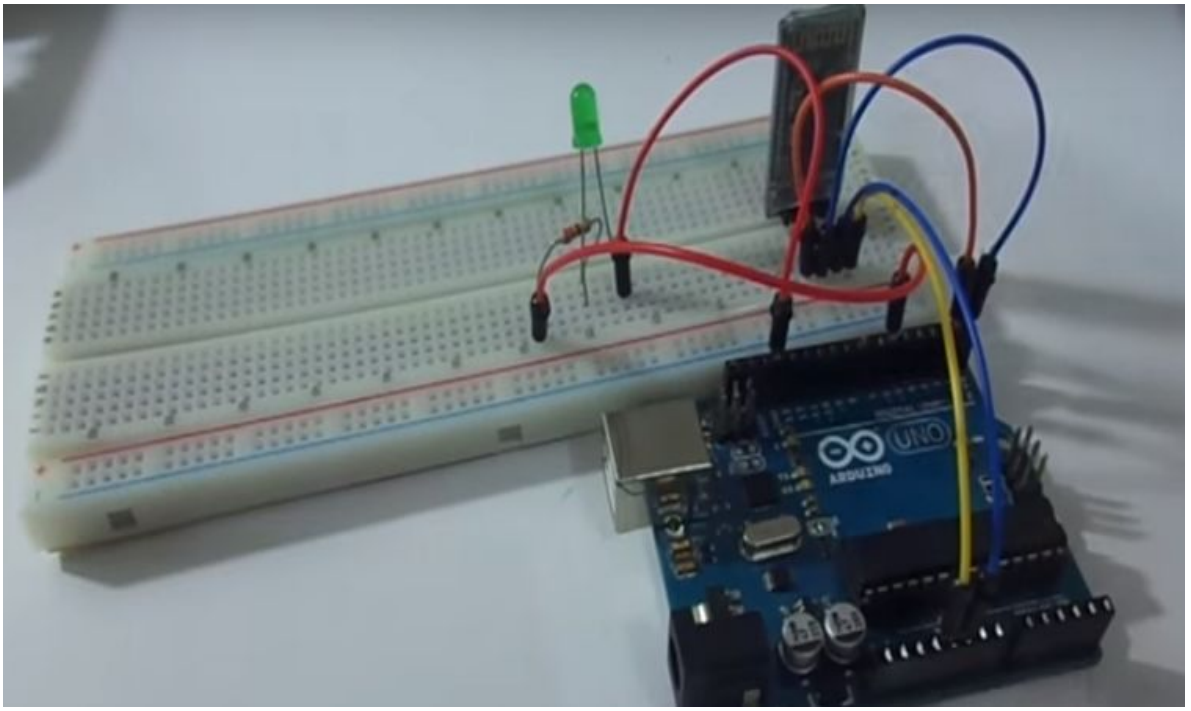


Figura 25 – Circuito *Bluetooth*
Fonte:(https://goo.gl/VZM2kf)

4.3.3 IMPLEMENTAÇÃO DO CÓDIGO DA QUANTIDADE DE RAÇÃO

Para a criação dos códigos a seguir, foi realizado um estudo no vídeo do internauta Vitor Santos (2016). Como possuem quatro botões para o dono do animal escolher qual será a quantidade de vezes que o animal se alimentará por dia, foi necessário a implementação um código que enviará a informação de cada botão para o *Arduino*, e como não foi possível criar a implementação usando números, foi necessário o uso de letras.

Já para enviar kg do animal, pressionando o botão enviar é feito uma checagem para saber se o *bluetooth* está ativado e conectado, se ele estiver conectado e ativado, o texto é enviado para o *Arduino* e logo após é limpado a caixa de texto do aplicativo. Após ter escolhido a quantidade que o animal se alimentará por dia e ter digitado qual seu peso, a codificação juntamente com o *Arduino* irá multiplicar esse peso pelos 2% indicados e dividir por quatro, três ou dois , dependendo da escolha do dono do animal. Vide Figura 26.

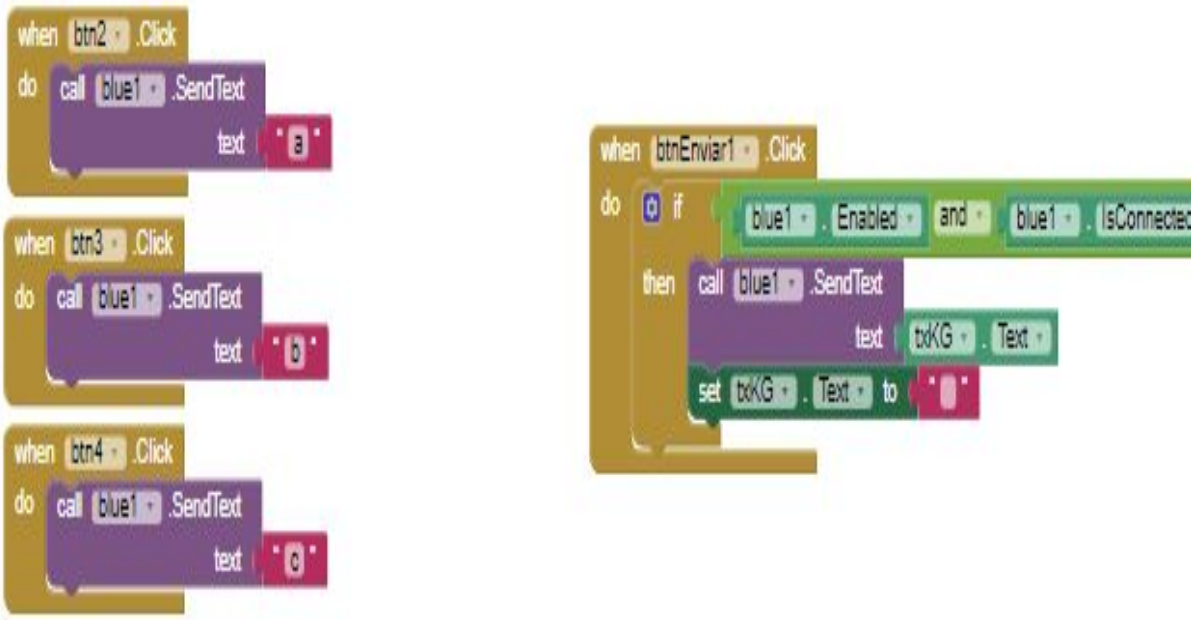


Figura 26 – Código ração *App Inventor*

Fonte: A autora

4.4 IMPLEMENTAÇÃO NO PROCESSING

No *Processing* a implementação se inicializa com a montagem da interface do protótipo. Primeiramente, foram feitos dois retângulos, um azul e um marrom. O azul representa o reservatório de água, e o marrom o de ração. Em seguida foram feitas marcações nos reservatórios a fim de marcar a quantidade de substâncias existente em cada um e por último, foi adicionado imagens de quadriláteros na ponta dos reservatórios e potes. Os quadriláteros foram colocados na ponta de cada reservatório para simular a parte da saída do alimento e da água e os potes onde os mesmos serão depositados. Vide Figura 27.

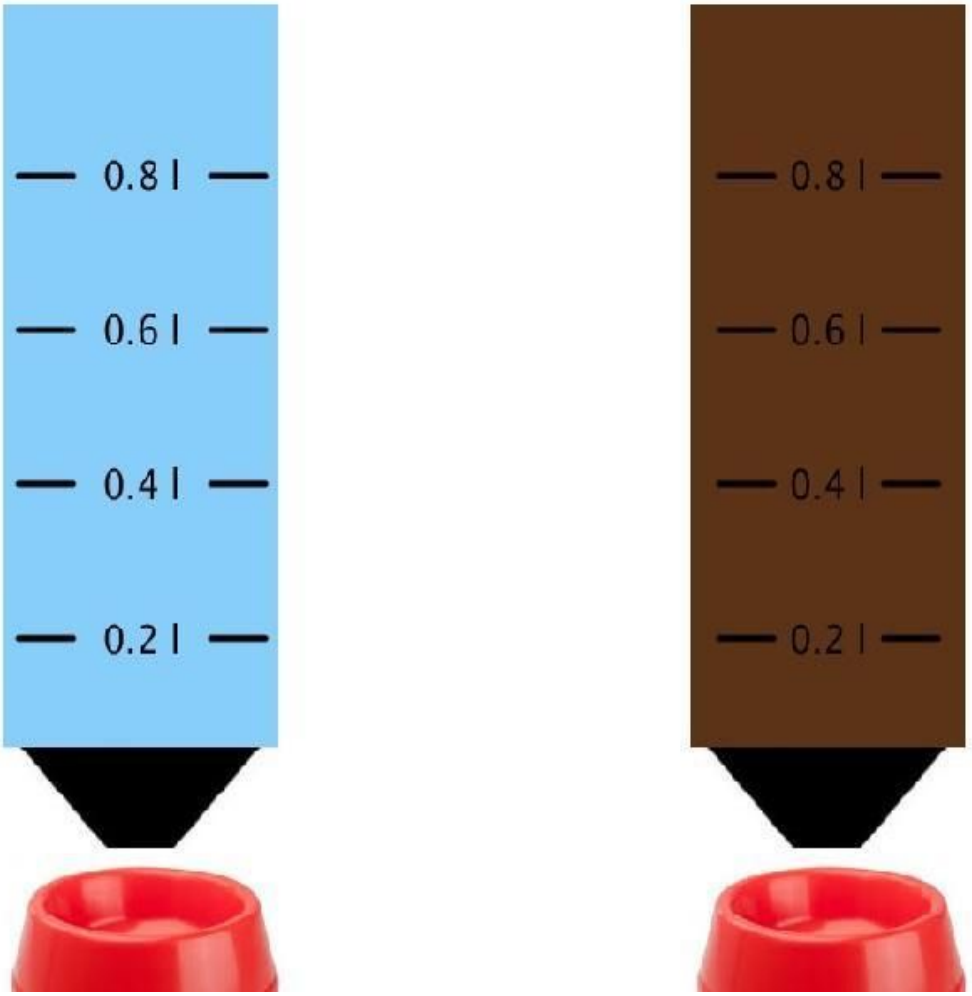


Figura 27 – Imagem Protótipo

Fonte: A autora

4.5 PRINCIPAIS PROBLEMAS ENFRENTADOS

Inicialmente foi pensado junto ao orientador Otávio Gomes, em fazer a manutenção e revisão de laboratórios de informática em escolas públicas. O objetivo geral era proporcionar aos alunos da mesma, um contato efetivo com a informática, auxiliando-os na educação formal através da manutenção dos ambientes onde a prática e o aprendizado de informática seria realizado. O que não foi possível pois, com o passar do tempo, a proposta não foi aceita e a mudança do projeto foi necessária.

Com isso, começou a correria para encontrar um projeto interessante, foi iniciado diferentes pesquisas e a implementação do protótipo atual que teve um início tardio. Sendo assim o principal problema enfrentado foi o tempo e o custo, já que, a proposta inicial contava com um protótipo físico, o que não terminaria dentro do tempo e gastaria mais do que o planejado o que acarretou numa mudança do resultado final. Foi usada assim, uma ferramenta capaz de transformar as ideias do protótipo físico em um protótipo virtual.

5 RESULTADOS

Após o processo de desenvolvimento, foi desenvolvido um simulador que pode ser utilizado como prova de conceito para o desenvolvimento de um protótipo, que pode ser feito com canos pvc e pode fornecer água e ração simultaneamente. No início do projeto foi pensado em montar o protótipo com seus equipamentos físicos, mas no decorrer do trabalho surgiu a necessidade de fazer uso do *Processing*, e ele possibilitou que o projeto fosse comprovado efetivamente sem a necessidade da montagem do protótipo.

O protótipo se inicia com a conexão do aplicativo com o *bluetooth* e após ter realizado a conexão é possível adicionar o peso do animal e fazer a escolha de quantas alimentações ele terá por dia. Essas informações do aplicativo são enviadas para o *Arduin*o, onde é feita a codificação das informações para que o animal receba suas refeições no horário e na quantidade necessária. O *Arduino* envia as informações para o *Processing* que mostra em sua interface como funciona o armazenamento da ração e da água. Segue abaixo o diagrama explicativo sobre o funcionamento do projeto, vide figura 28.

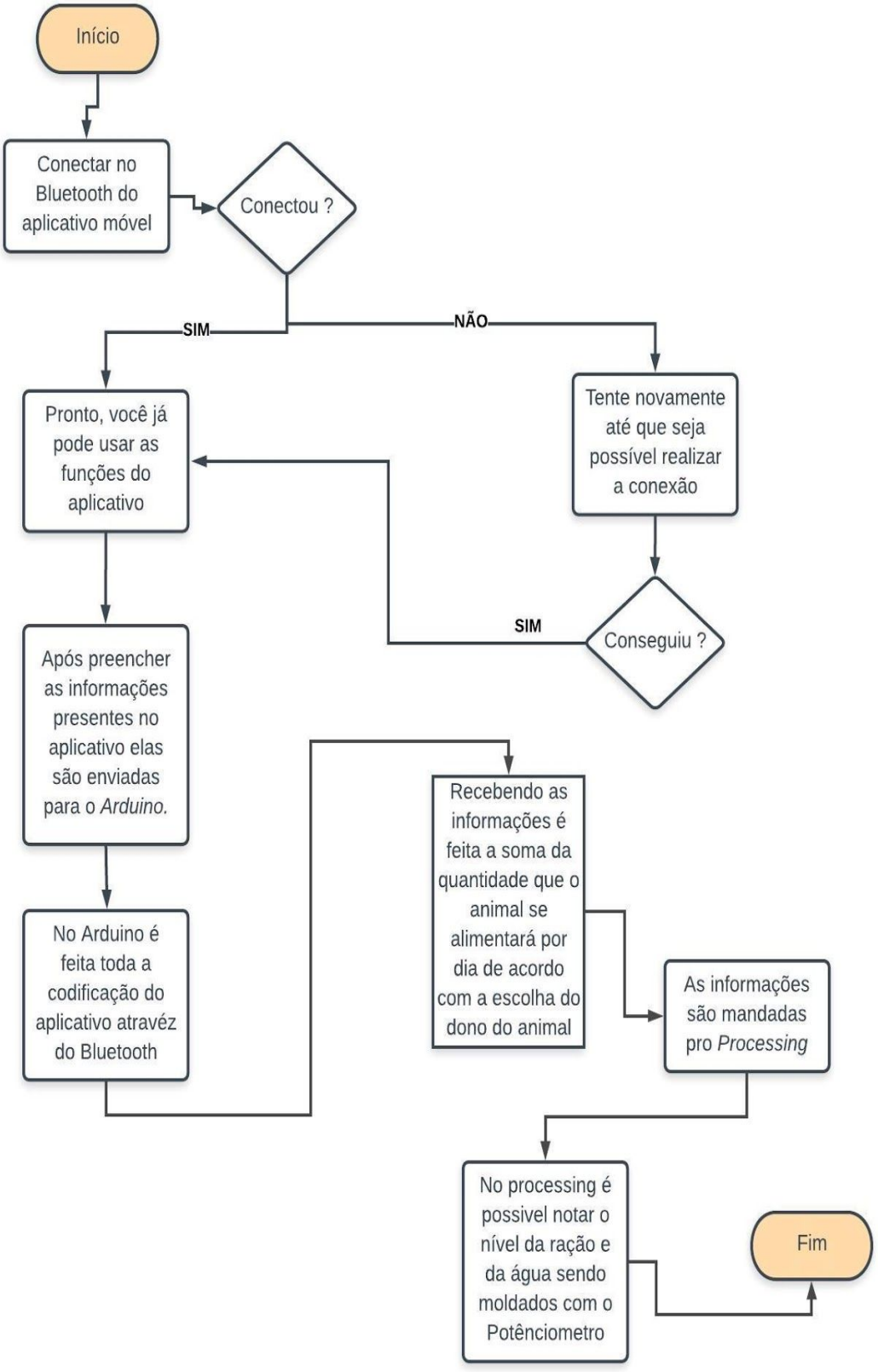


Figura 28 – Diagrama do projeto

Fonte: A autora

6 CONCLUSÃO

Em virtude do que foi mencionado, o simulador teve sucesso nos testes visto que, sua montagem não foi realizada mas não se vê a necessidade da mesma já que a efetividade do *processing* é notada e o tempo dado não seria o bastante para a montagem da placa física.

De acordo com o relatório feito de informações acumuladas em hotéis qualificados para receberem os pets na cidade de Formiga-MG, a diária para um cachorro de porte médio é de R$ 30,00. Ademais, um preço estipulado para a montagem do projeto tendo em vista o preço dos equipamentos usados é de R$ 350,00. Sendo assim, é possível notar que a montagem do projeto vale a pena já que o protótipo pode ser usado diariamente e seu valor pode ser atingido em apenas onze dias em um hotel de *pets*.

7 TRABALHOS FUTUROS

As possibilidades para os trabalhos futuros são imensas, indo desde a montagem do protótipo físico até o aprimoramento e a inserção de tecnologias capazes de deixar a máquina com um grande diferencial se comparada com as existentes no mercado. A criação da máquina física, fazendo o uso de um motor de passo é uma ótima opção, já que, ele é vastamente usado em projetos onde há a necessidade de sua precisão. É vasta a quantidade de projetos que o utiliza já que com uma codificação simples pode-se definir sua velocidade, a contagem de seus passos por rotações e se o motor girará em sentido horário ou anti-horário.

Outrossim, seria muito interessante a inserção de uma balança para medir quantos quilos o animal possui com exatidão e mandar a informação para o *Arduino*, assim o aplicativo ficaria ainda mais simples. Outra possibilidade seria inserir baias individuais para pessoas que possuem mais de um *pet* em casa. Pensando nisso, seria essencial a utilização de um RFID[5] na coleira do pet ou a criação de uma visão computacional inteligente para identificar qual animal está prestes a se alimentar, ou se existe um mais "guloso" que outro.

Ademais, seria interessante utilizar um *log* com uma variação do peso do animal com alertas para um maior monitoramento do animal proporcionando o proprietário saber da variância do kg do seu *pet* . Além disso, o *log* também ajudaria saber qual o tempo que o animal fica sem se alimentar e quando o nível do reservatório está baixo é muito relevante já que avisando ao proprietário do pet ele pode reabastecer o reservatório sem que o nível de ração acabe totalmente.

Por fim, é muito importante criar uma espécie de banco de dados que armazene a memória com os horários que o animal se alimenta deixando o sistema totalmente autônomo e precise apenas ser reabastecido para que seu funcionamento ocorra com o dono do animal presente ou não.

---

[5] CIRIACO, Douglas. **Como funciona a RFID?**. 2009. Disponível em: <https://www.tecmundo.com.br/tendencias/2601-como-funciona-a-rfid-.htm>. Acesso em: 20 nov. 2018.

REFERÊNCIAS BIBLIOGRÁFICAS

CPS . **Alimentador automático para animais de estimação é invenção de alunos da Etec Bento Quirino**. 2016. Disponível em: <https://www.cps.sp.gov.br/alimentador-automatico-para-animais-de-estimacao-e-invencao-de-alunos-da-etec-bento-quirino/>. Acesso em: 20 mar. 2018.

SANTOS, Bruno Feu De Brito. **ALIMENTADOR AUTOMÁTICO PARA ANIMAIS UTILIZANDO ARDUINO**. 2015. Disponível em: <https://repositorio.unesp.br/bitstream/handle/11449/139072/000864824.pdf?sequence=1&isAllowed=y>. Acesso em: 20 mar. 2018.

JUNIOR, Douglas Rossi Buogo Joselino Xavier. **Protótipo de alimentador automático para animais domésticos - Gingapets**. 2017. Disponível em: <http://joinville.ifsc.edu.br/~bibliotecajoi/arquivos/tcc/mecind/180273.pdf>. Acesso em: 20 mar. 2018.

THOMSEN, Adilson. **O que é Arduino?**: Como nasceu o Arduino? Para que serve um Arduino? Quais as vantagens? Como eu começo a programar? Nesse tutorial vamos apresentar um resumo sobre o que é Arduino e como você pode utilizá-lo em seus projetos.. 2014. Disponível em : <http://HTTPS://WWW.FILIPEFLOP.COM/BLOG/O-QUE-E-ARDUINO/>. Acesso em: 23 mar. 2018.

GUISS, Alexandre. **Google App Inventor: o criador de apps para Android para quem não sabe programar.** 2011. Disponível em: <http://HTTPS://WWW.TECMUNDO.COM.BR/GOOGLE/11458-GOOGLE-APP-INVENTOR-O-CRIADOR-DE-APPS-PARA-ANDROID-PARA-QUEM-NAO-SABE-PROGRAMAR.HTM>. Acesso em: 23 mar. 2018

PRADO, Thiago Pereira do. **Tinkercad: ferramenta online e gratuita de simulação de circuitos elétricos.** 2018. Disponível em: < https://www.embarcados.com.br/tinkercad/>. Acesso em: 19 maio 2018.

THOMSEN, Adilson. **Tutorial Módulo Bluetooth com Arduino**. 2015. Disponível em: <https://www.filipeflop.com/blog/tutorial-modulo-bluetooth-com-arduino/>. Acesso em: 19 maio 2018.

ARDUINO E CIA, Arduino e Cia. **Código de cores de resistores**. 2013. Disponível em: <https://www.arduinoecia.com.br/2013/08/codigo-de-cores-de-resistores.html>. Acesso em: 19 maio 2018.

SEARCHING TABS. **Calculadora código de cores de resistores**: (Cor - valor numérico ea valor numérico - cor). s.d. Disponível em: <http://www.searchingtabs.com/rcolor/rescolor_port.htm#cl)>. Acesso em: 20 maio 2018.

LEMOS, Manoel. **PROJETO 12 - MOTOR DE PASSO CONTROLADO POR ARDUINO**. 2013. Disponível em: <http://facacomarduino.info/projeto-12-motor-de-passo-controlado.html>. Acesso em: 20 maio 2018.

KALATEC AUTOMAÇÃO. **O que é um motor de passo?**. s.d. Disponível em: <http://www.kalatec.com.br/O-QUE-E-UM-MOTOR-DE-PASSO/>. Acesso em: 20 maio 2018.

MATTEDE, Henrique. **O que é um LED?**. s.d. Disponível em: <https://www.mundodaeletrica.com.br/o-que-e-um-led/>. Acesso em: 20 maio 2018.

ROBO CORE. **Como utilizar uma Protoboard**. 2017. Disponível em: <https://www.robocore.net/tutoriais/como-utilizar-uma-protoboard.html>. Acesso em: 20 maio 2018.

CIRIACO, Douglas. **Como funcionam as telas de LCD:**. 2009. Disponível em: <https://www.tecmundo.com.br/televisao/2058-como-funcionam-as-telas-de-lcd-.htm>. Acesso em: 25 maio 2018.

MURTA, Gustavo. **Guia completo do Display LCD – Arduino.** s.d. Disponível em: <http://HTTP://BLOG.ELETROGATE.COM/GUIA-COMPLETO-DO-DISPLAY-LCD-ARDUINO/>. Acesso em: 25 maio 2018.

EQUIPE BAÚ DA ELETRÔNICA. **Conhecendo a Biblioteca LiquidCrystal.** s.d. Disponível em: <http://HTTP://BLOG.BAUDAELETRONICA.COM.BR/CONHECENDO-BIBLIOTECA-LIQUIDCRYSTAL/>. Acesso em: 25 maio 2018.

USINAINFO. **SENSOR DE FLUXO DE ÁGUA PARA ARDUINO 1-30 L/MIN.** 2016. Disponível em: <http://HTTP://BLOG.USINAINFO.COM.BR/SENSOR-DE-FLUXO-DE-AGUA-PARA-ARDUINO-1-30-LMIN/>. Acesso em: 25 maio 2018.

LABORATÓRIO DE GARAGEM. **Tutorial: Como utilizar o Sensor de Fluxo de Água - G 3/4 com Arduino.** 2014. Disponível em: <http://labdegaragem.com/profiles/blogs/tutorial-como-utilizar-o-sensor-de-fluxo-de-agua>. Acesso em: 25 maio 2018.

REIS, Fábio dos. **Arduino – Conhecendo as funções pinMode, digitalRead e digitalWrite.** s.d. Disponível em: <http://HTTP://WWW.BOSONTREINAMENTOS.COM.BR/ELETRONICA/ARDUIN

O/ARDUINO-CONHECENDO-AS-FUNCOES-PINMODE-DIGITALREAD-E-DIGITALWRITE/>. Acesso em: 08 jun. 2018.

THOMSEN, Adilson. **Configuração do módulo bluetooth HC-06 com Arduino.** 2015. Disponível em: <http://buildbot.com.br/blog/configuracao-do-modulo-bluetooth-hc-06-com-arduino/>. Acesso em: 06 jul. 2018.

THOMSEN, Adilson. **Tutorial Módulo Bluetooth com Arduino.** 2015. Disponível em: <https://www.filipeflop.com/blog/tutorial-modulo-bluetooth-com-arduino/>. Acesso em: 06 jul. 2018.

ARDUINO E CIA, Arduino e Cia. **Módulo Bluetooth JY-MCU - HC-06 – Configuração.** 2013. Disponível em: <https://www.arduinoecia.com.br/2013/03/modulo-bluetooth-jy-mcu-configuracao.html>. Acesso em: 06 jul. 2018.

PIOVANI, Lucas. **Módulo Bluetooth HC-06 com Arduino.** 2016. Disponível em: <https://www.youtube.com/watch?v=Z57sLToEW5s>. Acesso em: 06 jul. 2018.

PROFJR. **Ligando um led com AppInventor, Arduino e módulo Bluetooth.** 2014. Disponível em: <https://www.youtube.com/watch?v=VSUwpigQ4s0>. Acesso em: 06 jul. 2018.

COMO FAZER AS COISAS. **Potenciômetro, o que é, para que serve, tipos, aplicações e como funciona.** s.d. Disponível em: <http://www.comofazerascoisas.com.br/potenciometro-o-que-e-para-que-serve-e-como-funciona.html>. Acesso em: 10 set. 2018.

COMO FAZER AS COISAS. **Potenciômetro o que é, para que serve, tipos e aplicações.** s.d. Disponível em:

<https://www.youtube.com/watch?time_continue=2&v=JYY9vuhwk5c>. Acesso em: 10 set. 2018.

GIZMODO. **Lindos visuais criados com a linguagem Processing**. 2008. Disponível em: <https://gizmodo.uol.com.br/lindos-visuais-criados-com-linguagem-processing/>. Acesso em: 10 set. 2018.

SPARKFUN. **Connecting Arduino to Processing.** s.d. Disponível em: <https://learn.sparkfun.com/tutorials/connecting-arduino-to-processing>. Acesso em: 10 set. 2018.

ARDUINO E CIA, Arduino e Cia. **Processing : mostre as informações do Arduino no seu computador.** 2014. Disponível em: <https://www.arduinoecia.com.br/2014/09/tutorial-arduino-processing.html>. Acesso em: 10 set. 2018.

ASSIS, LUIZA CERVENKA DE. **Pesquisa aponta que a maioria dos cães e gatos estão obesos**. 2018. Disponível em: <https://emais.estadao.com.br/blogs/comportamento-animal/pesquisa-aponta-que-a-maioria-dos-caes-e-gatos-estao-obesos/>. Acesso em: 11 out. 2018.

PETLOVE. **Alimentador Eletrônico Hoison Babá Robô para Pet**. s.d. Disponível em: <https://www.petlove.com.br/alimentador-eletronico-hoison-baba-robo-para-pet/p>. Acesso em: 11 out. 2018.

SANTOS, Vitor. **Monitor Serial - bluetooth - App Inventor 2** - Android #03.s.d. Disponível em: <https://www.youtube.com/watch?v=x1NqRMmsTyM>. Acesso em: 25 out. 2018.

STACKOVERFLOW. **How do I convert char to int on Arduino serial read()?.** s.d.Disponível em: <https://stackoverflow.com/questions/26811300/how-do-i-convert-char-to-int-on-arduino-serial-read>. Acesso em: 25 out. 2018.

8 APÊNDICES

APÊNDICE A - Tabela de preços

Quantidade	Material	Preço
01	Placa *Arduino UNO*	R$:45,90
01	*Protoboard*	R$:30,00
01	*Buzzer*	R$:19,00
02	*Leds*	R$:12,00
01	Resistor	R$:9,40
01	*Display LCD*	R$:27,49
02	Sensor ultrassônico hc-sr04	R$:12,79
02	Comedouros (ração e água)	R$:28,90
01	Módulo de *Bluetooth* HC-06	R$:30,00
01	Motor de passo	R$:115,00
02	Canos PVC	R$:18,90

APÊNDICE B - Código de criação dos protótipos virtuais e conexão com *Arduino*

```
import processing.serial.*;
Serial minhaPorta;

float total = 0.0;
int pot01 = 0;
int pot02 = 0;

int x_pot01 = 0;
int x_pot02 = 0;

PImage foto;
PImage foto1;

void setup(){
   size(800, 1000);
   textAlign(CENTER);
   textSize(26);
   String portName = Serial.list()[0];
   minhaPorta = new Serial(this, portName, 9600);
   smooth(2);
   foto=loadImage("triangulo.jpg");
   foto1=loadImage("racao.jpg");
}

void serialEvent (Serial myPort){

   String myString = myPort.readStringUntil('\n');

   if (myString != null) {
      myString = trim(myString);

      int values[] = int(split(myString, ','));

     if (values.length >= 3) {

      total = (float)values[0];
      pot01 = values[1];
      pot02 = values[2];

      println(total +" "+ pot01 +" "+ pot02);
     }
   }

   if (pot01 <= 0) {
     x_pot01 = -5;
   }
   else if (pot01 == 1){
```

```
    x_pot01 = -70;
  }
  else if (pot01 == 2){
    x_pot01 = -170;
  }
  else if (pot01 == 3){
    x_pot01 = -270;
  }
  else if (pot01 == 4){
    x_pot01 = -370;
  }
  else if (pot01 >= 5){
    x_pot01 = -470;
  }


  if (pot02 <= 0) {
    x_pot02 = -5;
  }
  else if (pot02 == 1){
    x_pot02 = -70;
  }
  else if (pot02 == 2){
    x_pot02 = -170;
  }
  else if (pot02 == 3){
    x_pot02 = -270;
  }
  else if (pot02 == 4){
    x_pot02 = -370;
  }
  else if (pot02 >= 5){
    x_pot02 = -470;
  }

}

void draw() {

  background(255);
  fill(255, 255, 255);
  text("Arduino Level Monitor", 225, 50);
  text("Status", 350, 270);

  // Criacao do nível de água
  fill(135, 206, 250);
  noStroke();
  rect(50, 540, 200, x_pot01);

  // Criacao do medidor de água
  noFill();
  stroke(0,0,0);
  rect(50, 60, 200, 480);



  image(foto1,40,560);
```

```
image(foto,60,540);

// Marcacoes no medidor de água
fill(0);
text("0.2 l", 150, 480);
text("0.4 l", 150, 380);
text("0.6 l", 150, 280);
text("0.8 l", 150, 180);
fill(0);
stroke(0);
strokeWeight(4);


// Linhas da esquerda (medidor de água)
line(50, 170, 100, 170);
line(50, 270, 100, 270);
line(50, 370, 100, 370);
line(50, 470, 100, 470);

// Linhas da direita (medidor de água)
line(190, 170, 240, 170);
line(190, 270, 240, 270);
line(190, 370, 240, 370);
line(190, 470, 240, 470);
noStroke();


// Criacao do nível de ração
fill(#5C3317);
noStroke();
rect(550, 540, 200, x_pot02);

// Criacao do medidor de ração
noFill();
stroke(0,0,0);
rect(550, 60, 200, 480);


// Marcacoes no medidor de ração
fill(0);
text("0.2 l", 650, 480);
text("0.4 l", 650, 380);
text("0.6 l", 650, 280);
text("0.8 l", 650, 180);
fill(0);
stroke(0);
strokeWeight(4);

// Linhas da esquerda (medidor de ração)
line(570, 170, 610, 170);
line(570, 270, 610, 270);
line(570, 370, 610, 370);
line(570, 470, 610, 470);

// Linhas da direita (medidor de ração)
line(690, 170, 730, 170);
line(690, 270, 730, 270);
```

```
  line(690, 370, 730, 370);
  line(690, 470, 730, 470);


  // Imagens dos potes de água e de ração)
  noStroke();
  image(foto1,540,560);
  image(foto,560,540);

  while (minhaPorta.available() > 0) {
    serialEvent(minhaPorta);
  }

  // Esperar 1 segundo
  delay(250);
}
```