UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ (UTFPR)
CURSO DE ENGENHARIA DE COMPUTAÇÃO

GABRIEL CARRICO GUERRERO
LUCAS FELIPE RIBEIRO
LEONARDO MURAROTO REIS

ROBÔ AUXILIAR PARA APRENDIZAGEM DE INGLÊS

OFICINA DE INTEGRAÇÃO 2 – RELATÓRIO FINAL

CURITIBA

2021

GABRIEL CARRICO GUERRERO
LUCAS FELIPE RIBEIRO
LEONARDO MURAROTO REIS

ROBÔ AUXILIAR PARA APRENDIZAGEM DE INGLÊS

Relatório Final da disciplina Oficina de Integração 2, do curso de Engenharia de Computação, apresentado aos professores que ministram a mesma na Universidade Tecnologica Federal do Paraná como requisito parcial para obtenção da aprovação na disciplina.

Orientador: Prof. Dr. César Manuel Vargas Benítez
Prof. Dr. Heitor S. Lopes

CURITIBA

2021

Este trabalho é dedicado a todas as vítimas da COVID-19, dentro e fora do Brasil. . .

AGRADECIMENTOS

Agradecemos ao corpo docente da UTFPR, sem os quais esse projeto não seria possível. Agradecemos também aos nossos familiares e amigos, que nos ajudaram com eventuais problemas do desenvolvimento, bem como construção da mecânica do projeto.

RESUMO

. ROBÔ AUXILIAR PARA APRENDIZAGEM DE INGLÊS. 58 f. Oficina de Integração 2 – Relatório Final – Curso de Engenharia de Computação, UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ (UTFPR). Curitiba, 2021.

O projeto relatado neste documento se refere ao desenvolvimento de um prótipo de robô que visa auxiliar crianças a terem um contato lúdico com o idioma inglês. O trabalho foca na comunicação entre a criança e o robô, para isso foram utilizados alguns componentes: para que o robô entenda a criança, foram acoplados um sensor de presença PIR e um sensor de cor e gesto, este segundo para os exercícios visuais, bem como uma Webcam com microfone embutido. Já para que a criança entenda o robô, foram acoplados uma tela de LCD e dois speakers. São disponibilizados rotinas de exercícios que podem ser disparados e customizados através de um aplicativo de celular. O robô ainda conta com quatro motores de passo, que são ativados por meio de comando de voz ao pedido da criança.

Palavras-chave: Raspberry Pi, Sistemas Embarcados, Aprendizagem Automatizada

ABSTRACT

. HELPER ROBOT FOR ENGLISH LEARNING. 58 f. Oficina de Integração 2 – Relatório Final – Curso de Engenharia de Computação, UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ (UTFPR). Curitiba, 2021.

This project consists of the development of a prototype of a robot that helps children to have a ludic contact with english. The project focuses on the communication between the child and the robot, for that it was used some components: for the robot to understand the child, it was coupled a PIR presence sensor and a sensor of gestures and colour, the latter for the visual exercises, it was also coupled a webcam with a mic. For the child to understand the robot, it was coupled a LCD screen and two speakers. The robot has routines of exercises that can be initiated and customized from a smartphone application. The robot also has for step motors, that are activated by voice from the child.

Keywords: Raspberry Pi, Embedded Systems, Automated Learning

LISTA DE FIGURAS

LISTA DE TABELAS

SUMÁRIO

1 INTRODUÇÃO

1.1 MOTIVAÇÃO

A ideia por trás do ZéColBerry se deu com o intuito de auxiliar na aprendizagem de idiomas para crianças de forma lúdica. O sistema seria utilizado por pais e/ou professores com ouma ferramenta auxiliar no aprendizado das crianças,ao mesmo tempo que proporciona um momento de lazer. Aproximar a língua estrangeira do contexto do aluno aumenta em muito o ritmo de aprendizagem, independentemente da idade.

1.2 OBJETIVOS

1.2.1 OBJETIVO GERAL

A proposta do projeto é fornecer uma série de exercícios e interações entre a criança e o robô, para que a mesma possa aprender inglês de forma mais divertida.

1.2.2 OBJETIVOS ESPECÍFICOS

O sistema deve fornecer dois tipos de interfaces e interações: a primeira entre o adulto responsável pela criança e o robô é feita unicamente através do aplicativo para celular, e a segunda entre a criança e o robô é feita através de estímulos visuais e sonoros. O aplicativo deve permitir uma conexão Bluetooth com o ZéColBerry. Também deve permitir ao adulto responsável selecionar a rotina de exercícios desejada, sua dificuldade e monitorar os níveis de acertos da criança. Deve ser permitido o cadastro de novas crianças dentro do aplicativo. A interface entre o robô e a criança deve permitir que a mesma responda aos exercícios fornecidos pelo robô através da fala e também através das formas geométricas que foram construídas e serão disponibilizadas. O robô deve, dependendo do nível de dificuldade do exercício, fornecer um auxílio extra por meio da tela de LCD. Também deve ser possível para a criança comandar que o robô mova um membro específico. Os exercícios planejados para o protótipo são: exercícios de cor, exercícios de forma, exercícios de contagem e exercícios de dia da semana.

1.2.3 REQUISITOS FUNCIONAIS

RF01 - O Hardware deve receber e processar um sinal da câmera.

RF02 - O Hardware deve receber e processar um sinal do microfone.

RF03 - O Hardware deve ler o sensor de proximidade.

RF04 - O Hardware deve ler o sensor de cor.

RF05 - O Hardware deve controlar os motores que movimentam os membros.

RF06 - O Hardware deve enviar um sinal para o speaker.

RF07 - O Aplicativo deve permitir o registro do nome da criança.

RF08 - O Aplicativo deve permitir a conexão com o Sistema através de Blutooth.

RF09 - O Aplicativo deve permitir a customização das rotinas de exercício.

RF10 - O Aplicativo deve permitir a seleção do nível de dificuldade dos exercícios.

RF11 - O Aplicativo deve permitir o acesso ao histórico de acertos da criança.

RF12 - O Aplicativo deve permitir a seleção dentre rotinas pré-programadas.

RF13 - O Aplicativo deve informar se a criança está pronta para avançar de nível

RF14 - O Firmware deve conectar com o aplicativo através de Bluetooth.

RF15 - O Firmware deve conectar com a rede através de Wi-Fi.

RF16 - O Firmware deve armazenar informações de acerto e erro da criança.

RF17 - O Firmware deve enviar as informações ao final de cada rotina de exercícios para o aplicativo.

RF18 - O Firmware deve hibernar até o sensor de movimento indicar a presença da criança.

RF19 - O Firmware deve reconhecer a criança através da câmera.

RF20 - O Firmware deve se comunicar com a criança através do speaker.

RF20.1 - O Firmware deve oferecer exercícios de contagem.

RF20.2 - O Firmware deve oferecer exercícios de cor.

RF20.3 - O Firmware deve oferecer exercícios de forma.

RF20.4 - O Firmware deve oferecer exercícios de dia da semana.

RF21 - O Firmware deve interpretar a resposta da criança e pontuar de acordo.

RF21.1 - Exercícios de contagem e dia da semana recebem respostas sonoras.

RF21.2 - Exercícios de cor e forma recebem resposta visual.

RF22 - O Firmware deve utilizar o sensor de cor para interpretar as respostas visuais.

RF23 - O Firmware deve mover o membro correspondente ao pedido pela criança.

RF24 - O Firmware deve responder a cumprimentos com o nome da criança.

RF25 - Na dificuldade ”Easy”o Firmware deve escrever a frase do exercício em português no LCD.

RF26 - Na dificuldade ”Normal”o Firmware deve escrever a frase do exercício em inglês no LCD.

RF27 - Na dificuldade ”Hard”o Firmware não deve escrever nada no LCD.

1.2.4 REQUISITOS NÃO FUNCIONAIS

RNF01 - O Sistema deve ser implementado utilizando uma Raspberry Pi 3 B.

RNF02 - O Sistema deve ser implementado utilizando um Sensor de Movimento HC-SR505.

RNF03 - O Sistema deve ser implementado utilizando um Sensor de Cor Apds-9960.

RNF04 - O Sistema deve ser implementado utilizando um Display LCD modelo KeyPad Shield 1602.

RNF05 - O Sistema deve ser implementado utilizando quatro Servo-motores para os membros.

RNF06 - O Sistema deve ser implementado utilizando uma Webcam USB.

RNF07 - O Sistema deve ser implementado utilizando um Speaker.

RNF08 - O Aplicativo deve ser desenvolvido com o Flutter.

RNF09 - O Aplicativo deve ser desenvolvido em Dart.

RNF10 - O Aplicativo deve fornecer os dados em uma dashboard.

RNF11 - O Aplicativo deve apresentar interface gráfica.

RNF12 - O Firmware deve ser implementado em uma Raspberry Pi.

RNF13 - O Firmware deve utilizar a solução da Azure para processar áudio e imagem.

RNF14 - O Firmware deve ser capaz de enviar informação em JSON.

RNF15 - O Firmware deve ser capaz de receber informação em JSON.

RNF16 - O Firmware deve ser desenvolvido em Python.

2 FUNDAMENTAÇÃO TEÓRICA

2.1 SENSOR PIR

O DYP-ME003 é um módulo baseado no detector de movimento BISS0001 PIR. O módulo em questão é utilizado para detectar o movimento. Ele tem como faixa de detecção entre 3 e 7 metros, que pode ser ajustado através do potenciometro de distância, localizado na parte inferior do módulo. O módulo ainda conta com outro potenciometro, este para regular o tempo de delay, os tempos variam entre 5 e 200 segundos. Ambos os potenciometros podem ser vistos na Figura 1.

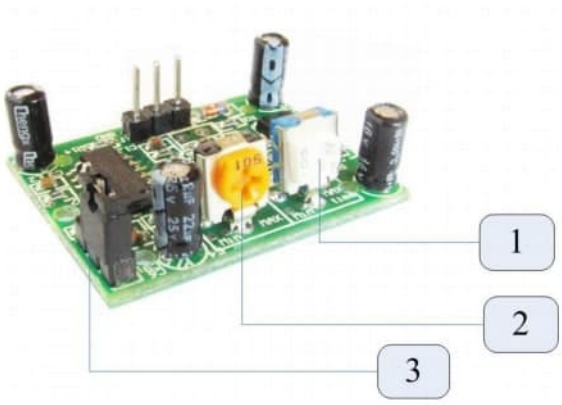

Figura 1: Sensor PIR, visão inferior. (1) Potenciometro de Tempo de Delay; (2) Potenciometro de Distância; (3) Jumper de Modo Fonte: (OPENIMPULSE,)

Além disso, o módulo é coberto por uma Lente de Fresnel, o que permite com que capte movimentos num ângulo de 110º. Além disso, o sensor possui um jumper para definir se utilizará disparo único de presença ou múltiplos disparos, o modo pardão, que foi utilzado neste projeto, é o modo de disparo único.

O Sensor PIR funciona captando através de um par de sensores piroelétricos a temperatura do ambiente ao redor. Todo ser vivo emite algum grau de raios infravermelhos, e é essa detecção que dispara o sensor. Portanto não é um sensor de posicionamento, o Sensor PIR apenas detecta que algo se moveu num dado ambiente. (COOK,)

2.2 SENSOR DE COR E GESTOS

O APDS-9960 foi o sensor de cor e gestos escolhido para o desenvolvimento do projeto. O sensor em questão utiliza a tecnologia I2C para comunicação entre o sensor e a Raspberry Pi.

O sensor apresenta detecção de gestos, proximidade, luz ambiente e sensor de cor. Para o desenvolvimento do projeto, as informações importantes eram de luz ambiente e cor.

A função de detecção de cor e luz ambiente provem dados de azul, vermelho, verde e intensidade de iluminação. Cada canal possui um filtro UV e IR e um conversor dedicado, provendo 16-bits de dados simultaneamente. A Figura 2 apresenta o diagrama de blocos do APDS-9960.

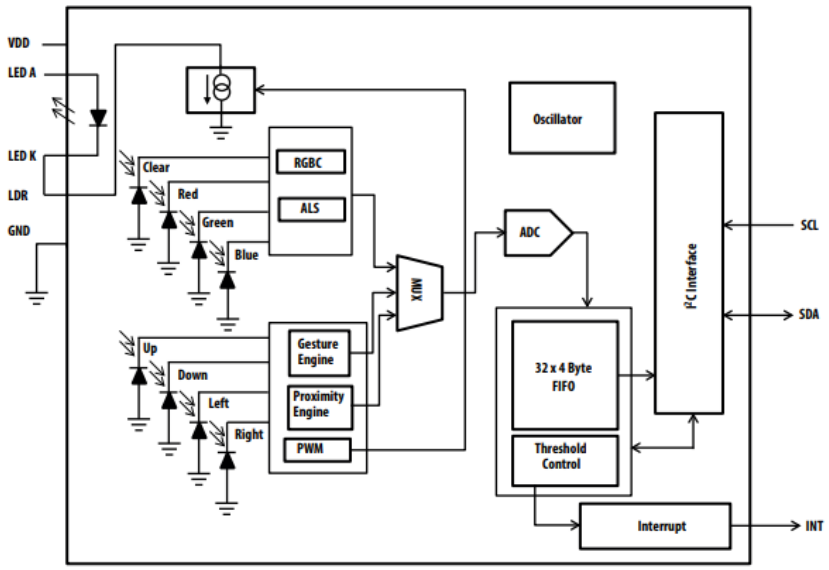

Figura 2: Diagrama de Blocos: APDS-9960 Fonte: (AVAGO TECHNOLOGIES, 2013)

Como dito anteriormente, o controle e o interfaceamento do sensor é feito através de um barramento I2C. Ele suporta um único endereço escravo hexadecimal 0x39 utilizando um endereçamento 7-bits.

O barramento pode ser utilizado no modo leitura, escrita ou combinado. Durante a operação de escrita, o primeiro byte é um byte de comando. Na operação combinada, o primeiro byte escrito é o byte de comando, seguido de uma série de leituras. No comando de leitura, o endereço do registrador do último comando será utilizado para acesso de dados. (AVAGO TECHNOLOGIES, 2013)

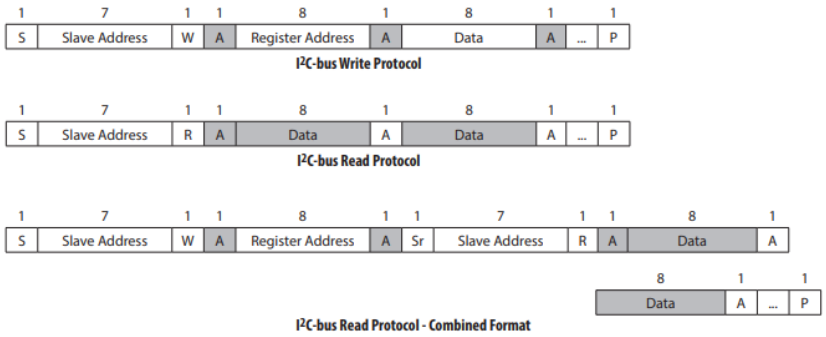

Figura 3: Barramento do Protocolo I2C Fonte: (AVAGO TECHNOLOGIES, 2013)

2.3 LCD

O display de LCD foi construído em um controlador LSI, o controlador possui dois registradores de 8-bits, um para instruções (IR) e outro para os dados (DR). O IR guarda instruções, tais como para limpar o display ou mover o cursor. O DR guarda temporariamente os dados que serão lidos ou escritos na DDRAM ou CGRAM. A DDRAM é utilizada para salvar os dados do display em códigos de caractere de 8-bits. Possui a capacidade para até 80 caracteres.

O dispay de LCD consegue gerar caracteres de 5x8 ou 5x10. A Figura 4 apresenta as possibilidades de caracteres gerados pelo LCD. O LCD precisa de uma alimentação de 5V. (CRYSTALFONTZ AMERICA, INC.,)

2.4 MOTOR DE PASSO

O motor de passo é um tipo de atuador elétrico cujo principio de funcionamento é a conversão de pulsos elétricos em movimentos mecânicos com variações angulares discretizadas. Os motores de passo são empregados sempre que o objetivo central de um projeto se focar na precisão, uma vez que esses motores não são tão velozes nem apresentam torques altos. Sendo assim, é bastante aplicado em implementações como brinquedos elétricos, o que serve para o projeto em questão.

O motor de passo funciona de forma com que existem solenóides alinhados dois a dois, que quando energizados atraem o rotor em sua direção. A movimentação do rotor faz com que ocorra uma variação no ângulo, assim causando o denominado passo. O motor de passo pode funcionar em full-step ou half-step. No caso do full-step, cada bobina é acionada em ordem individualmente ou dois a dois, mas sempre sem alternância. No caso do half-step, ocorre uma alternância, ora são duas bobinas acionadas, ora apenas uma, sempre também em uma mesma

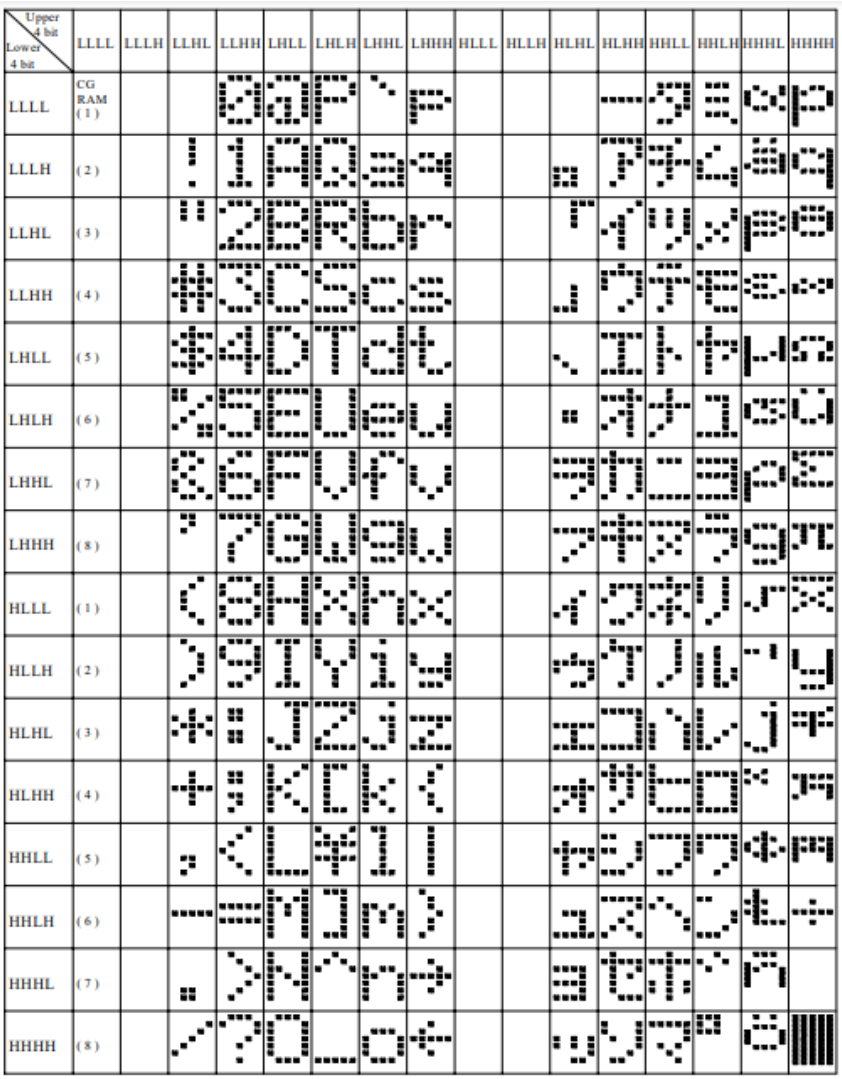

Figura 4: Tabela de Caracteres Fonte: (CRYSTALFONTZ AMERICA, INC.,)

ordem. O projeto conta com um motor bipolar, isto é, permite movimentação em ambos os sentidos, e os acionamentos em meio passo 5 e passo completo 1,2 estão nas tabelas abaixo. Por se tratar de um motor bipolar, a tabela pode ser percorrida no sentido crescente, bem como no sentido decrescente.

Referência completa para o motor de passo pode ser encontrada em (GONSALVEZ FELIPE E PUGA, 2008)

2.5 BLUETOOTH

”Bluetooth é o nome dado ao protocolo de rádio baseado em saltos de frequências de curto alcance (10 a 100 metros) que visa complementar ou substituir ás redes convencionais cabeadas, cujo meio físico de transmissão é o cabo de par trançado, cabo coaxial e fibra

Nº do Passo	B3	B2	B2	B0
1	1	0	0	0
2	0	1	0	0
3	0	0	1	0
4	0	0	0	1

Tabela 1: Passo Completo Um a Um Fonte: Autoria Própria

Nº do Passo	B3	B2	B2	B0
1	1	1	0	0
2	0	1	1	0
3	0	0	1	1
4	1	0	0	1

Tabela 2: Passo Completo Dois a Dois Fonte: Autoria Própria

Nº do Passo	B3	B2	B2	B0
1	1	0	0	0
2	1	1	0	0
3	0	1	0	0
4	0	1	1	0
5	0	0	1	0
6	0	0	1	1
7	0	0	0	1
8	1	0	0	1

Tabela 3: Passo Completo Fonte: Autoria Própria

óptica.”(JúNIOR,)

O protocolo utiliza modulação FHSS, com mudanças frequentes de frequência, para que assim seja difícil existirem dois dispositvos operando na mesma frequência ao mesmo tempo. O protocolo permite comunicação simétrica, podendo então ambos os dispositivos alternarem entre cliente e servidor. O protocolo está disponível em três versões, com mudanças ficando restritas às taxas de transmissão entre uma e outra.(JúNIOR,)

3 METODOLOGIA

O projeto utiliza um microcontrolador Raspberry Pi para controlar os motores de passo, ler o sensor PIR e a Webcam, exibir texto no LCD e reproduzir arquivos de áudio nos speakers. Ainda foi necessário que o Raspberry Pi se comunicasse com a Azure para processamento de dados em nuvem. O sistema foi construído utilizando a linguagem Python para o firmware. Para o aplicativo, foi utilizado a linguagem Dart, através do kit de desenvolvimento Flutter. O sensor PIR DYP-ME003 precisou de um divisor de tensão para funcionar corretamente, uma vez que ele foi produzido para ser utilizado com Arduíno, que possui tensão 5V em suas GPIO enquanto o Raspberry Pi possui tensão 3V3. O sensor de cor e gestos APDS-9960 e o LCD não precisaram de divisores de tensão, sendo fornecido para ambos dados através dos GPIO em 3V3.

3.1 VISÃO GERAL

O sistema ZéColBerry pode ser subdividido em algumas funções principais: a leitura dos sensores de cor e gestos e do sensor PIR; o interfaceamento com a Webcam USB e seu microfone emputido; o controle dos motores; a comunicação, tanto com a Azure utilizando a internet, quanto com o aplicativo utilizando Bluetooth; a comunicação com a criança através dos speakers e do LCD; e o controle dos motores. O diagrama em blocos do sistema como um todo pode ser visto na Figura 5.

Durante o desenvolvimento do projeto, foram alteradas partes significativas dessas funções. Tais mudanças serão prontamente abordadas no Capitulo 4. Mas apenas como visão geral, tivemos que abandonar para fins desse relatório a utilização do sensor de cor e gestos APDS-9960, bem como tivemos de fazer uma mudança na comunicação entre o Raspberry e o Aplicativo, deixando de lado o Bluetooth e investindo no protocolo HTTP.

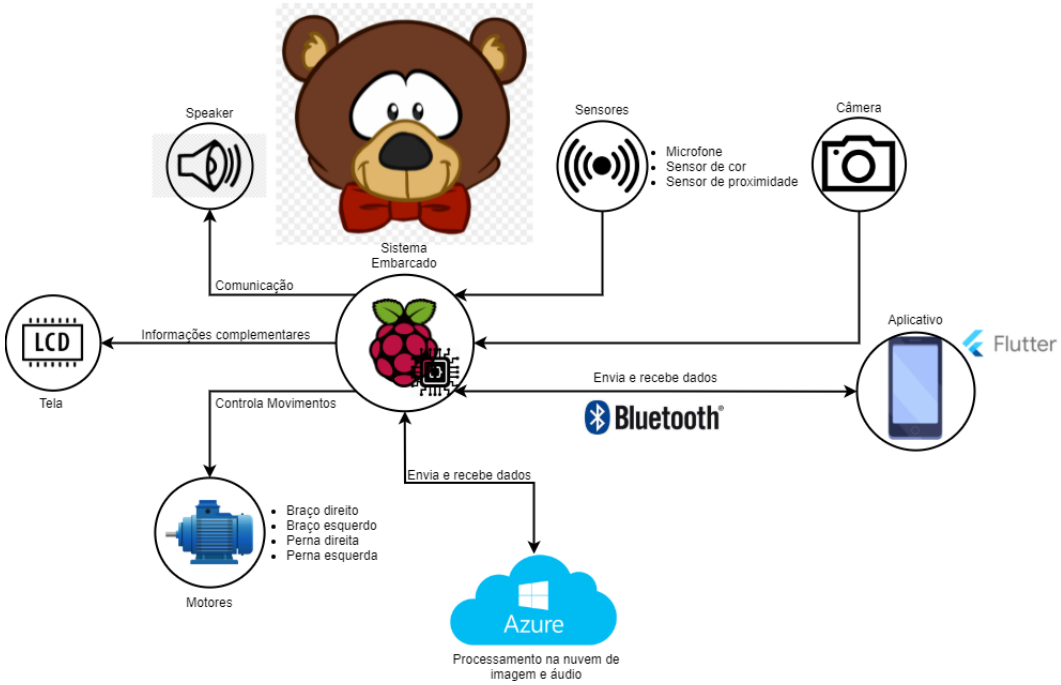

Figura 5: Diagrama de blocos Fonte: Autoria Própria

3.2 PROJETO MECÂNICO

O rascunho inicial da solução foi feito utilizando-se de um AutoCad de estudante cedido pela UTFPR e pode ser visto na Figura 6. Esse primeiro rascunho tinha como ideia ser construído de madeira. No entanto, como o pai do Gabriel possuí uma pequena fábrica de peças de alumínio, ocorreu uma mudança nos planos e o projeto final sofreu pequenas alterações como no seu material de fabricação.

Para a construção da carcaça do robô, foram utilizadas duas peças sobressalentes encontradas dentro da fábrica, as duas peças foram parafusadas unidas para formarem um único robô. As peças podem ser vistas nas Figuras 7 e 8.

As duas peças foram então furadas em locais específicos, Figura 9, para que fosse possível o acoplamento dos membros e motores, além do LCD. Além disso, as caixas de som foram desmontadas para poderem ser colocadas dentro do robô sem ocuparem todo o espaço interno.

Os quatro motores foram então aparafusados em seus respectivos lugares, já estando devidamente acoplados a carcaça para fins de solução final. A tela de LCD também já foi colocada no seu lugar final, embora ainda não tenha sido cabeada ou isolada da estrutura metálica.

Uma peça peitoral foi furada considerando as medidas do sensor de presença, para que

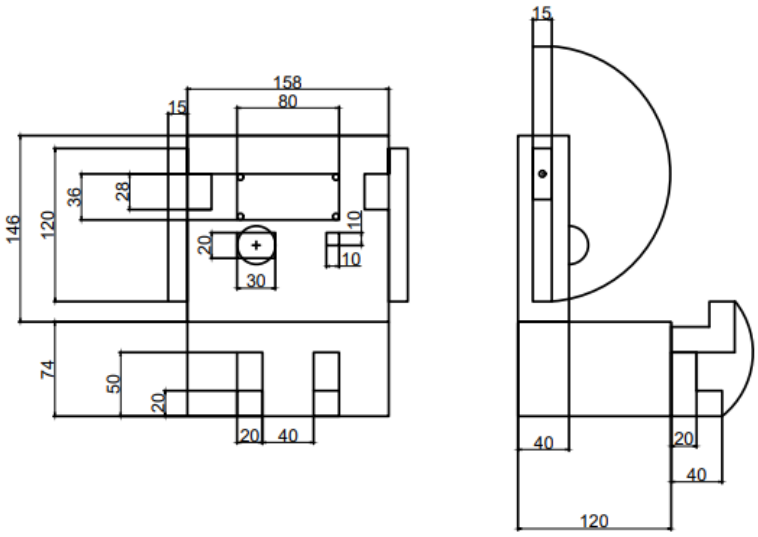

Figura 6: Diagrama do Projeto Mecânico Fonte: Autoria Própria

Figura 7: Carcaça Superior Fonte: Autoria Própria

Figura 8: Carcaças Empilhadas Fonte: Autoria Própria

Figura 9: Furação Peça Fonte: Autoria Própria

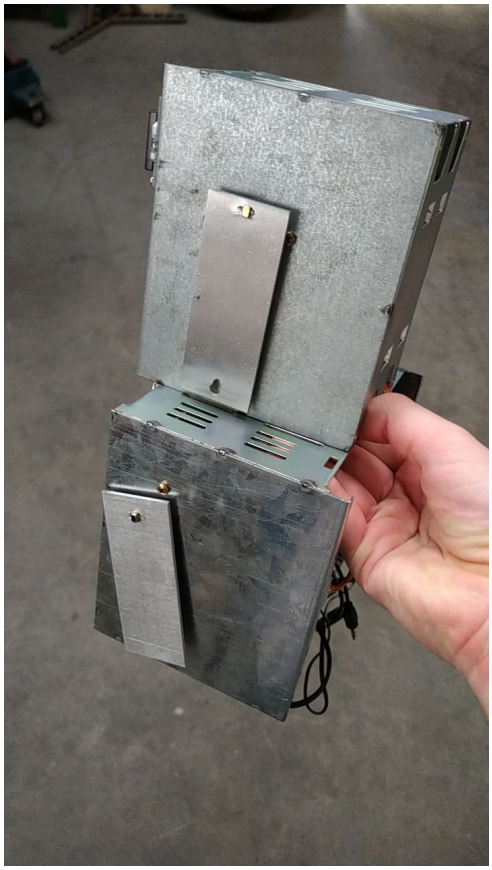

Figura 10: Membros Posicionados Fonte: Autoria Própria

esse possa posteriormente ser integrado a solução. No entanto o sensor ainda não foi acoplado a essa peça. Para fins de demonstração, a webcam foi inserida como cabeça do robô.

Placas de metal que estavam sobrando, além de terem sido usadas para o encaixe do sensor de presença, também acabaram servindo como os membros da solução. Essa é outra diferença importante entre o design inicial e a solução final, uma vez que o planejamento era de utilizar uma corda para mover as pernas, ao passo que a solução final apresenta o motor diretamente acoplado nos 4 membros, como pode ser visto na Figura 10.

Para a construção das formas geométricas, como se trata de um protótipo, foi utilizado EVA. Foram produzidas 4 formas, um circulo vermelho, um quadrado amarelo, um triângulo verde e um xis azul. As formas confeccionadas podem ser vistas na Figura 11

O plano original era confeccionar essas formas com madeira e pintá-las com tinta posteriormente, mas por restrições de tempo acabamos optando pela mudança. Para fins de teste o material das formas é indiferente, portanto não acreditamos que exista um impacto significativo no projeto por essa mudança.

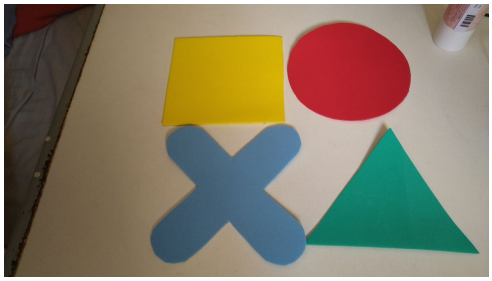

Figura 11: Formas Geométricas Fonte: Autoria Própria

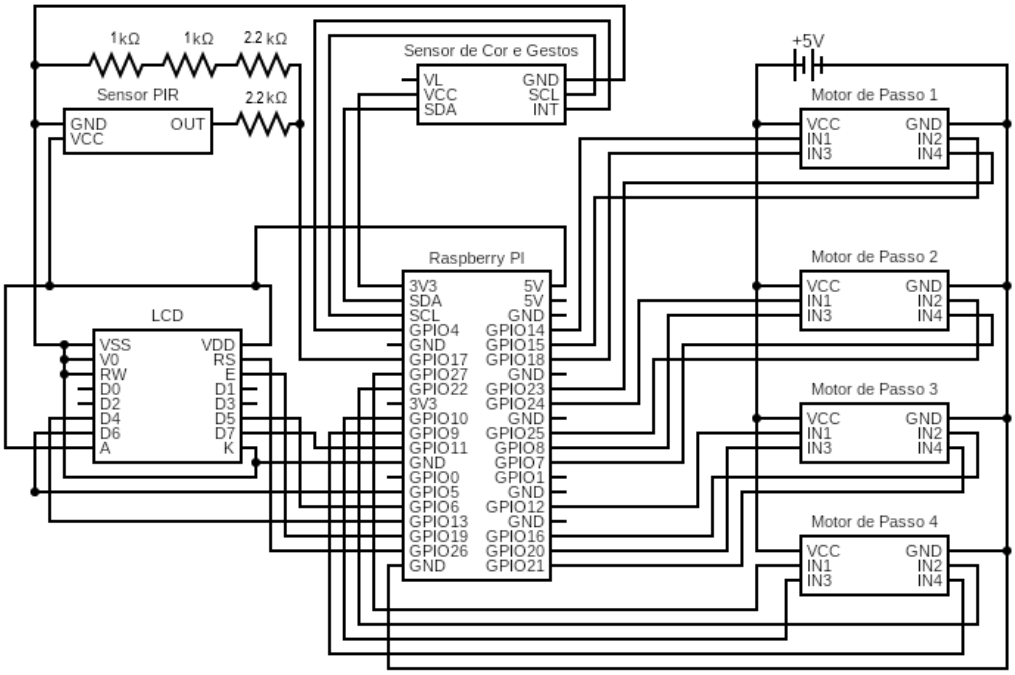

Figura 12: Diagrama Esquemático Fonte: Autoria Própria

3.3 PROJETO DE HARDWARE

O primeiro passo da definição do projeto de hardware foi a concepção do esquemático e do pinout do circuito. O coração do projeto de hardware se localiza em cima da Raspberry Pi, portanto ela foi determinante no planejamento do circuito. As figuras abaixo demonstram o esquemático, Figura 12, e o pinout, Figura 13.

O esquemático foi construído utilizando-se do Circuit Diagram Web Editor.

A instalação dos componentes foi feita juntamente a montagem da estrutura. Mais especificamente o acoplamento dos motores, Figura 14, e dos speakers, Figura 15, foi feito durante a montagem da estrutura. Ambos foram parafusados diretamente no esqueleto de metal.

O display de LCD passou por um processo de solda de seus jumpers, além disso foi

Raspberry Pi 3 B			
3V3	1	2	5V
SDA	3	4	5V
SCL	5	6	Ground
GPIO4	7	8	GPIO14
Ground	9	10	GPIO15
GPIO17	11	12	GPIO18
GPIO27	13	14	Ground
GPIO22	15	16	GPIO23
3V3	17	18	GPIO24
GPIO10	19	20	Ground
GPIO9	21	22	GPIO25
GPIO11	23	24	GPIO8
Ground	25	26	GPIO7
GPIO0	27	28	GPIO1
GPIO5	29	30	Ground
GPIO6	31	32	GPIO12
GPIO13	33	34	Ground
GPIO19	35	36	GPIO16
GPIO26	37	38	GPIO20
Ground	39	40	GPIO21
Speakers	USB2	USB1	Webcam
	USB3	USB4	
Speakers	P2	HDMI	

Motor de Passo 1	
5V	VCC
Ground	GND
GPIO14	IN1
GPIO15	IN2
GPIO18	IN3
GPIO23	IN4

Motor de Passo 2	
5V	VCC
Ground	GND
GPIO24	IN1
GPIO25	IN2
GPIO8	IN3
GPIO7	IN4

Motor de Passo 3	
5V	VCC
Ground	GND
GPIO12	IN1
GPIO16	IN2
GPIO20	IN3
GPIO21	IN4

Motor de Passo 4	
5V	VCC
Ground	GND
GPIO27	IN1
GPIO22	IN2
GPIO10	IN3
GPIO9	IN4

Sensor de Cor	
-	VL
Ground	GND
3V3	VCC
SCL	SCL
SDA	SDA
GPIO4	INT

Sensor de PIR	
Ground	GND
GPIO17	OUT
5V	VCC

Display LCD 16x2	
Ground	VSS
5V	VDD
Ground	V0
GPIO26	RS
Ground	RW
GPIO19	E
-	D0
-	D1
-	D2
-	D3
GPIO13	D4
GPIO6	D5
GPIO5	D6
GPIO11	D7
5V	A
Ground	K

Figura 13: Pinout Fonte: Autoria Própria

utilizada fita isolante, para evitar que parte dos circuitos da placa de circuito integrado do display tivessem contato direto com o alumínio, podendo causar assim mal funcionamento. O sensor PIR foi colado na estrutura utilizando-se de fita VHB da 3M, um procedimento semelhante foi feito para se afixar os 4 drivers dos motores de passo, bem como a webcam.

A construção do circuito final foi feita utilizando um protoboard. Essa decisão foi feita por restrição de tempo, bem como pelo fato de apenas um dos alunos, Gabriel, ter ficado responsável por toda a montagem mecânica e eletrônica do projeto. Apesar de não ser o ideal, dado os limitantes impostos pelo ensino a distância e considerando que o grupo construiu o projeto em cidades diferentes, acabou sendo definido que permaneceria assim.

Também foi feita uma fonte externa, uma vez que o Raspberry não se demonstrou capaz de fornecer a corrente desejada para a movimentação dos quatro motores. Foi comprada então uma fonte de 5V e corrente máxima de 2A, seu cabo foi cortado e soldado para que tivéssemos os conectores positivo e negativo da fonte separados e prontos para serem conectados ao protoboard. A fonte pode ser vista na Figura 16.

3.4 PROJETO DE SOFTWARE

3.4.1 DIAGRAMA DE CASOS DE USO

Damos início ao projeto de software apresentando o diagrama de casos de uso. Ele pode ser visto na Figura 17. Todos os diagramas desta sessão foram feitos utilizando a ferramenta online Draw.io.

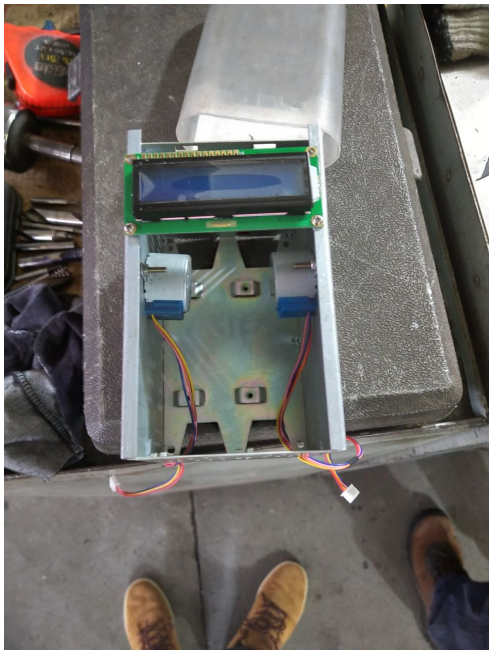

Figura 14: Acoplamento dos Motores e LCD Fonte: Autoria Própria

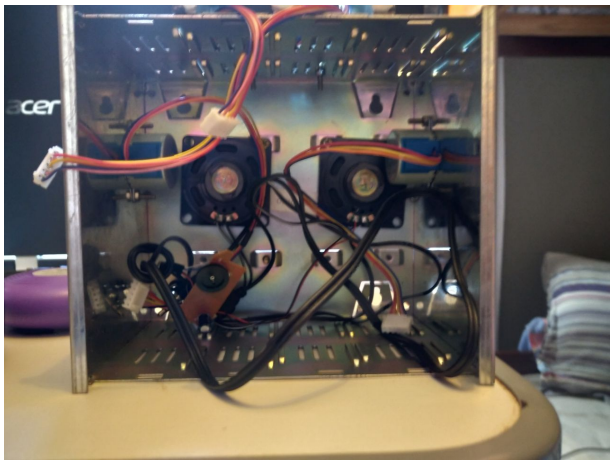

Figura 15: Acoplamento dos Speakers Fonte: Autoria Própria

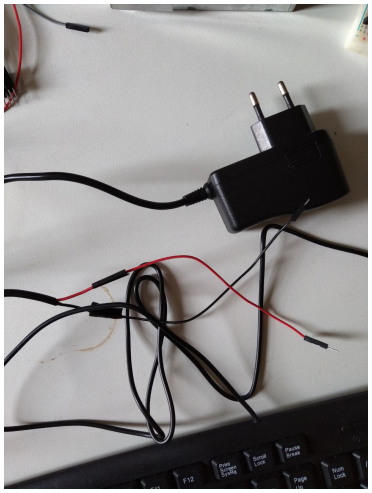

Figura 16: Construção da Fonte de Alimentação Fonte: Autoria Própria

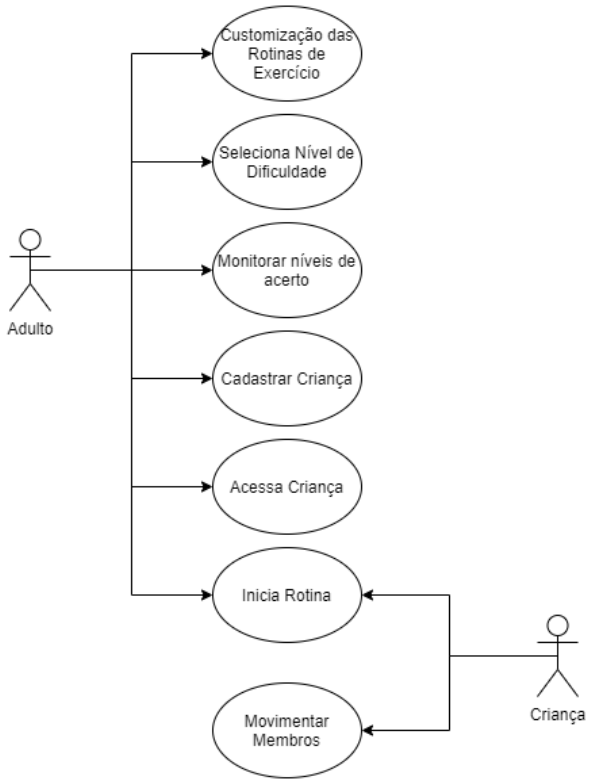

Figura 17: Diagrama de Casos de Uso Fonte: Autoria Própria

O ZéColBerry terá como interação dois atores: o adulto, que fará o acesso através do aplicativo de celular, e a criança que interagirá diretamente com o ZéColBerry.

Nos casos de uso temos que o Adulto pode customizar as rotinas de exercício (RF09), isso permite com que ele selecione quais exercícios serão feitos durante a rotina, dentre os quatro previamente planejados: contagem (RF20.1), cor (RF20.2), forma (RF20.3) e dia da semana (RF20.4). O caso de uso ainda prevê o salvamento da rotina (RF12).

O caso de uso 'Seleciona Nível de Dificuldade' permite ao adulto selecionar o nível de dificuldade de cada tipo de exercício individualmente (RF10). As opções de dificuldade são 'easy' (RF25), 'normal' (RF26) e 'hard' (RF27).

O caso de uso 'Monitora Níveis de Acerto' envolve o acesso ao histórico de acertos (RF11) bem como a informação de que a criança está pronta para avançar de nível (RF13).

O caso de uso de 'Cadastrar Criança' está mapeado com o requisito que o aplicativo deve permitir o registro do nome da criança (RF07). E caso de uso 'Acessa Criança' permite que uma criança já cadastrada continue sua aprendizagem de inglês.

O caso de uso 'Inicia Rotina' é o coração da nossa implementação. Ele é o caso mais complexo por envolver ambos os atores (Adulto e Criança) e a maior parte do processamento.

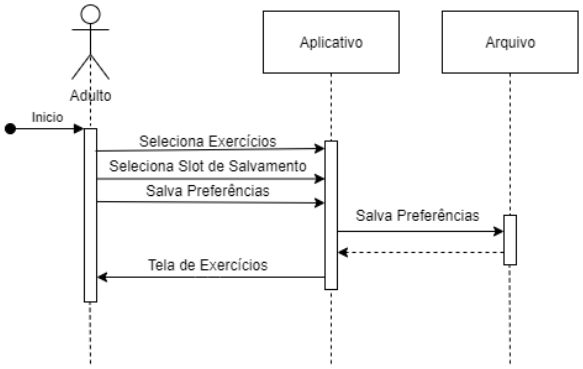

Figura 18: Customização das Rotinas de Exercício Fonte: Autoria Própria

O Inicia Rotina inicia com o pareamento entre o Aplicativo e o Raspberry através de Bluetooth (RF08, RF14). Após isso o Aplicativo envia a série de exercícios para o Raspberry. O Raspberry então aguarda a presença da criança (RF01, RF03, RF18, RF19). Então são iniciados os exercícios, o Raspberry deve utilizar o Speaker para informar a pergunta para a criança (RF20, RF06) e, dependendo da dificuldade, também o LCD (RF25, RF26, RF27). A leitura das respostas é feita de duas formas, através do Sensor de Cor (RF04, RF21.2, RF22)** ou através do Microfone (RF02). As respostas são então enviadas a Azure por Wi-Fi (RF15) e os resultados pontuados de acordo. Os dados históricos são então enviados de volta para o Aplicativo (RF16, RF17).

O último caso de uso, 'Movimentar Membros' diz respeito a possibilidade de a criança pedir ao ZéColBerry que mova algum dos quatro membros: 'braço direito', 'braço esquerdo', 'perna direita' ou 'perna esquerda'. O robô deve então mover o membro de acordo com o pedido (RF23).

3.4.2 DIAGRAMAS DE SEQUÊNCIA

Cada caso de uso da subsessão anterior foi mapeado então para um diagrama de sequência diferente, e estes podem ser vistos nas Figuras 18, 19, 20, 21, 22, 23 e 24.

3.4.3 MÁQUINA DE ESTADOS

A seguir temos a máquina de estados que representa o aplicativo, Figura 25. Ela tem seu início na 'Tela Inicial'. A partir dela temos duas transições, caso selecionada uma criança e clicar avançar, vamos para a 'Tela de Exercícios', caso clicar em cadastrar, vamos para 'Tela de Cadastro'.

A 'Tela de Cadastro' permite um retorno para a 'Tela Inicial' caso uma criança seja

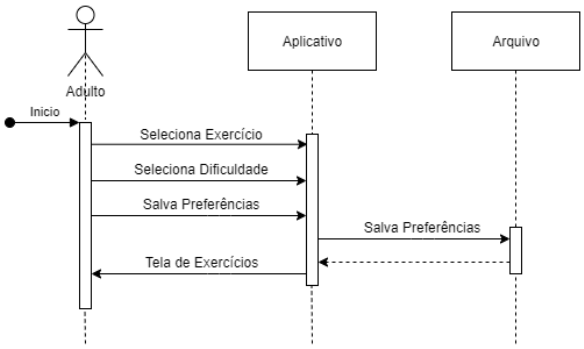

Figura 19: Customização das Rotinas de Exercício Fonte: Autoria Própria

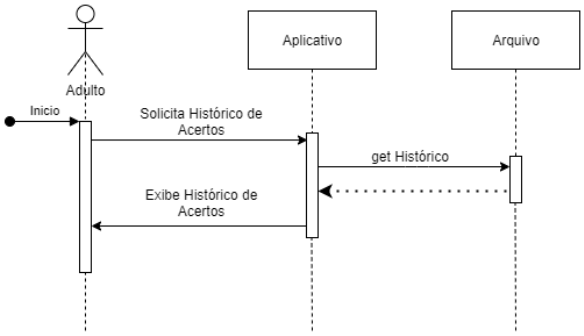

Figura 20: Monitorar Níveis de Acerto Fonte: Autoria Própria

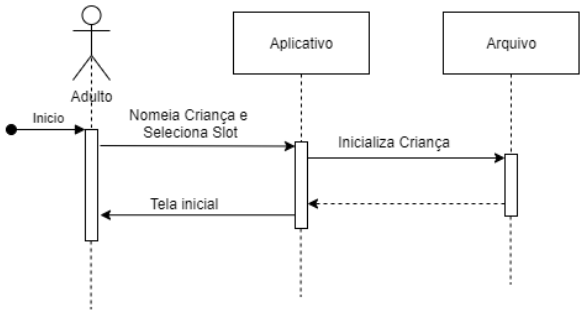

Figura 21: Cadastrar Criança Fonte: Autoria Própria

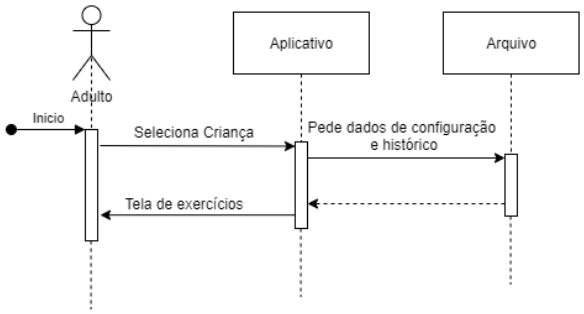

Figura 22: Acessar Criança Fonte: Autoria Própria

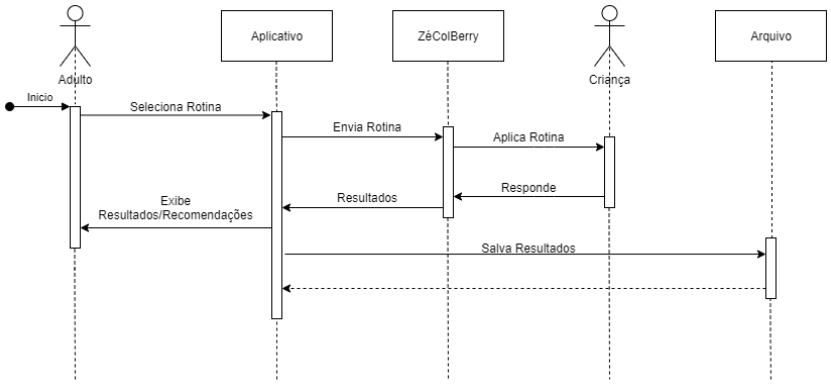

Figura 23: Iniciar Rotina Fonte: Autoria Própria

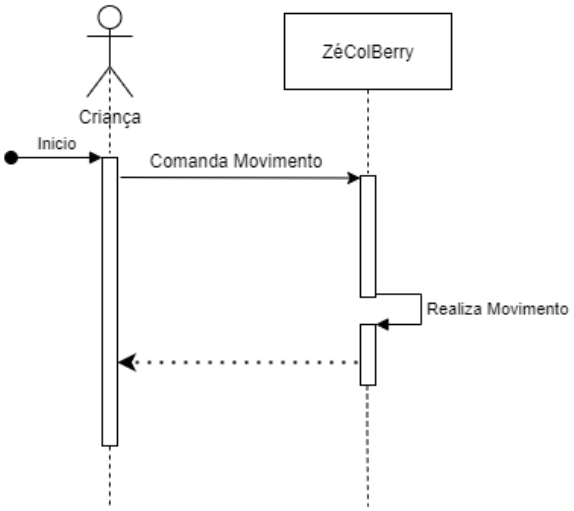

Figura 24: Movimentar Membros Fonte: Autoria Própria

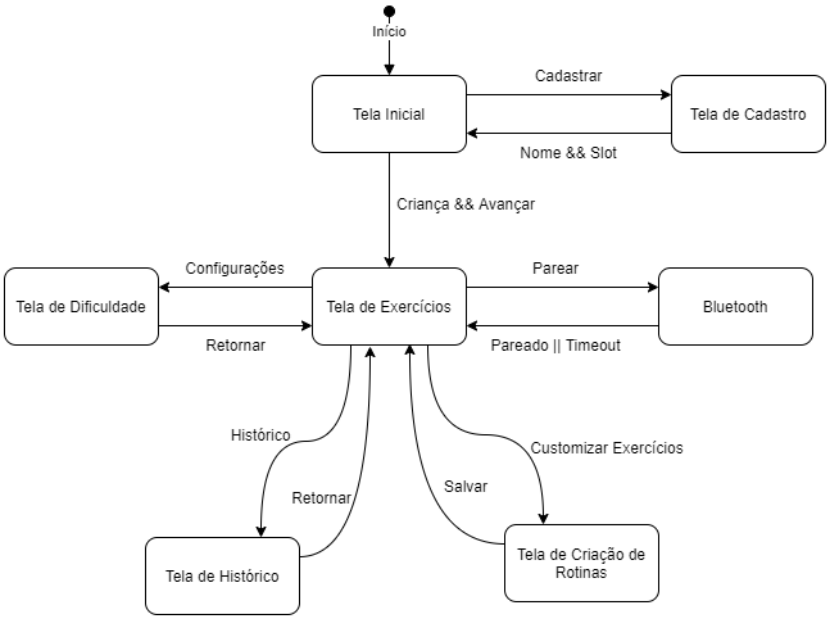

Figura 25: Máquina de Estados do Aplicativo Fonte: Autoria Própria

cadastrada. Para isso é necessário inserir um Nome para a criança e selecionar um Slot de salvamento. São 4 espaços para crianças disponíveis.

A 'Tela de Exercícios' é a tela central do Aplicativo. Com ela temos uma transição para 'Bluetooth' na tentativa de pareamento. Caso ocorra um pareamento ou caso o timeout seja atingido, retornamos para a 'Tela de Exercícios'.

Também podemos fazer uma transição para a 'Tela de Dificuldade'. Esta tela é a responsável por selecionarmos as dificuldades de cada tipo de exercício, como explicado previamente. Um botão de retorno está disponível nesta tela.

A 'Tela de Exercícios' também dá acesso a 'Tela de Histórico', na qual acessamos as pontuações da criança de rotinas passadas, divididas por cada tipo de exercício.

Por último temos a 'Tela de Criação de Rotinas', nesta tela podemos selecionar quais tipos de exercícios serão feitos numa dada rotina, bem como salvar essa rotina para uso posterior. Na versão atual, são 4 espaços de rotina disponíveis.

3.4.4 FLUXOGRAMA

Temos ainda o fluxograma para o Firmware, Figura 27. O funcionamento do fluxograma é dividido entre a 'Rotina de Exercícios' e o 'Movimenta Membros'.

A 'Rotina de Exercícios' tem seu inicio pareando com o Aplicativo. O ZéColBerry

então aguarda o envio de uma rotina por parte do Aplicativo. Enviada esta rotina, o ZéColBerry aguarda a criança ser captada pelo sensor de presença. Então a rotina começa, um exercício por vez. Cada exercício observa o nível de dificuldade, caso seja fácil, escreve no LCD a frase dita pelo ZéColBerry em português. Caso seja médio, escreve a frase dita pelo ZéColBerry em inglês. Caso seja difícil, não escreve no LCD. O ZéColBerry então aguarda pela resposta da criança, caso ela responda ou ocorra estouro do timer, passa então a tentar se conectar com a Azure e enviar os dados coletados. Quando receber o resultado do processamento, o ZéColBerry pontua a criança de acordo, e confere se chegou ao final da rotina. Caso tenha chego, envia os resultados para o Aplicativo e passa a executar o modo 'Movimenta Membros' até que uma próxima rotina seja enviada. Caso não seja o último exercício, passa para o próximo da fila.

O 'Movimenta Membros' permite a criança pedir que o ZéColBerry mova um membro solicitado. Ele permanece hibernando, até que o sensor de presença detecte a criança. A partir daí, o ZéColBerry capta a fala da criança, conecta-se com a Azure, e interpreta a fala. Caso a criança tenha feito um comando válido, o membro requisitado é movido pelo ZéColBerry. Caso contrário, o fluxo retorna ao começo da execução.

3.5 INTEGRAÇÃO

Para a integração fizemos uso de uma série de disciplinas vistas até aqui, bem como conhecimentos próprios dos alunos vistos em estágio ou cursos extra-curriculares.

Para o desenvolvimento mecânico, foram utilizados conhecimentos de Física 1, bem como aplicação de Desenho Técnico para operar o AutoCAD.

Para o desenvolvimento de hardware, foram utilizados conhecimentos de Circuitos Digitais em maior quantidade, já que o circuito é quase que totalmente digital, porém também foram utilizados conhecimentos de Eletrônica 1 e Física 3. Por se tratarem de conexões simples, não foi necessário a utilização de conceitos de Eletrônica 2. Todas as disciplinas voltadas para eletrônica também permitiram com que desenvolvessemos os esquemáticos e as montagens sem maiores problemas.

Para o desenvolvimento de software, foram utilizados conhecimentos de Sistemas Embarcados, Redes de Computadores para a comunicação HTTP e Programação para Dispositivos Móveis. Também foi importante lançar mão de disciplinas como Analise e Desenvolvimento de Sistemas e Engenharia de Software para projeto e documentação adequadas.

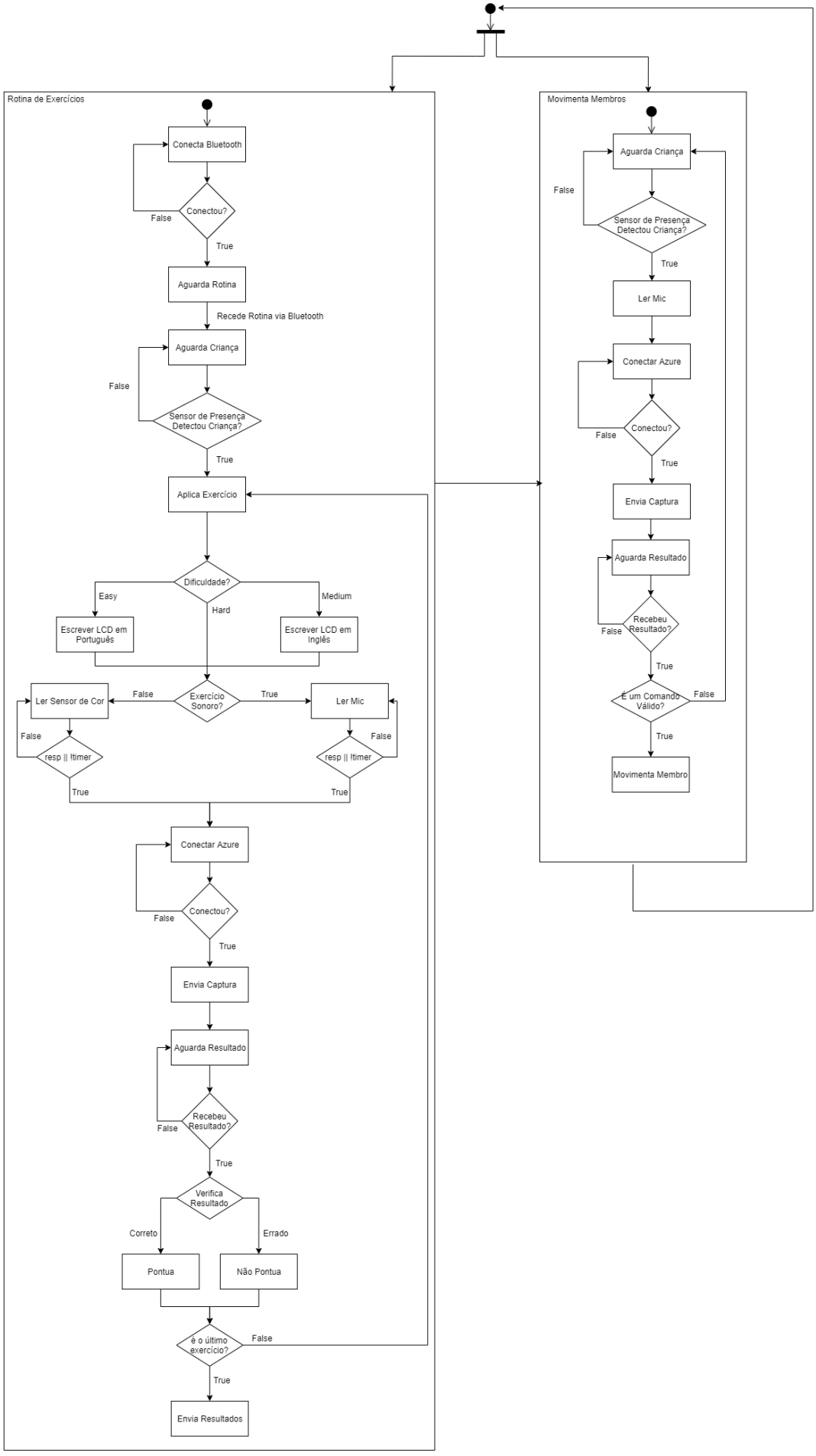

Figura 26: Fluxograma do Firmware Fonte: Autoria Própria

4 EXPERIMENTOS E RESULTADOS

O processo de experimentações e resultados foi desenvolvido ao longo de todo o semestre. Isso foi feito seguindo o cronograma da disciplina, que previa a apresentação de pequenos entregáveis semanais.

4.1 HARDWARE

Todos os componentes de Hardware funcionais possuem vídeos de seus testes que estão disponíveis no Blog da Disciplina (GUERRERO et al.,).

Os primeiros a serem testados foram os dois speakers, o projeto utiliza alimentação dos speakers através da porta USB do Raspberry e comunicação através da entrada P2. O teste demonstrou que os speakers funcionam de forma Plug'n Play, isto é, não necessitam de uma configuração extra por parte do Raspberry.

Para a Webcam também foi utilizado uma entrada USB da Raspberry, afim de termos acesso tanto ao áudio quanto ao vídeo da dita Webcam. Para a captura foi utilizada a FFmpeg, um programa que funciona por linha de comando que é composto de uma coleção de software livre e bibliotecas de código aberto.

O próximo teste a ser feito foi utilizando os motores já acoplados ao esqueleto. Era

Figura 27: Captura Webcam Fonte: Autoria Própria

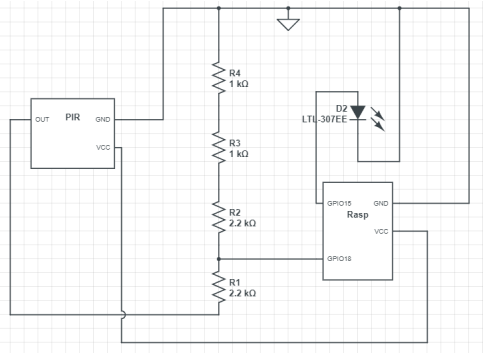

Figura 28: Circuito de Teste Sensor PIR Fonte: Autoria Própria

importante saber se o torque dos motores seria suficiente para mover os membros do robô. Com o teste, foi descoberto que sim. Também foi feito um teste de alimentar os 4 motores simultaneamente e mandar o Raspberry comandar 1 deles, para confirmar que a alimentação do mesmo seria suficiente. A partir deste segundo teste foi percebido que a Raspberry não daria conta, e uma fonte externa foi produzida e acoplada a solução, conforme já explicado no Capítulo 3.3 Projeto de Hardware.

O teste seguinte foi feito utilizando o display de LCD 16x2, para que o teste obtivesse sucesso, foi necessário o processo de solda dos jumpers ao LCD antes de seu uso. O código utilizado como exemplo para o teste do display de LCD pode ser encontrado em (ROTOTRON,).

Para o sensor PIR foi necessário o desenvolvimento de um pequeno circuito, afim de não queimarmos a entrada do Raspberry. Para isso foram utilizados 2 resistores de 2,2k Ohm e 2 resistores de 1k Ohm afim de fazermos um divisor de tensão e alimentarmos corretamente a porta do GPIO. Utilizamos ainda um segundo GPIO com um LED vermelho apenas para demonstrar o funcionamento do PIR. A Figura 28 apresenta o esquemático do circuito montado para teste.

As Figuras 29 e 30 são imagens ilustrativas dos testes, no entanto os vídeos deixam toda a operação muito melhor elucidada e a equipe encoraja fortemente que sejam assistidos.

Infelizmente durante os testes com o Sensor Cor Rgb E Gesto Apds-9960 não funcionou durante os testes. A conexão dele é feita através do I2C, mas o Raspberry Pi não conseguiu reconhecer a conexão do componente. Outro sensor foi encomendado, O sensor de cor foi reposto, mas os testes continuaram sem funcionar. Aventou-se a possibilidade de que o problema esteja na Raspberry Pi. Iremos redimensionar de forma que as cores serão descobertas através da imagem da webcam utilizando a solução da Azure.

Entrando um pouco mais no teste em si, utilizando-se das bibliotecas, é necessário usar

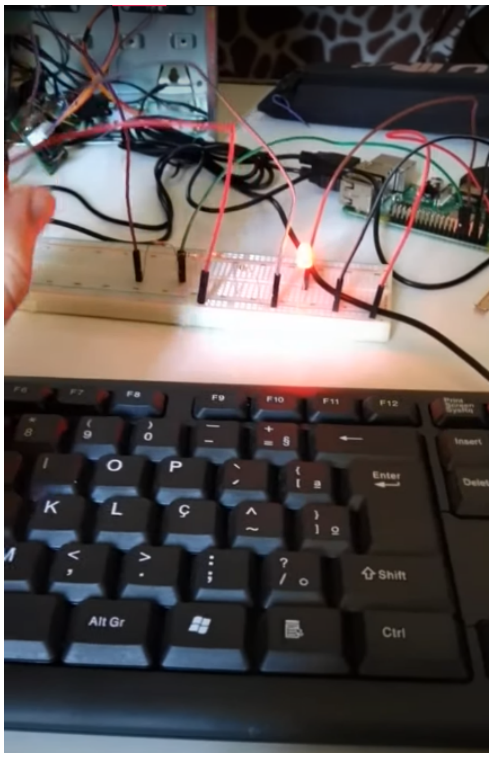

Figura 29: Teste de Hardware: Sensor PIR Fonte: Autoria Própria

Figura 30: Teste de Hardware: Display LCD Fonte: Autoria Própria

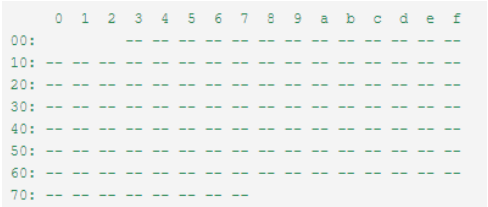

Figura 31: Teste de Hardware: Detecção I2C Fonte: Autoria Própria

o I2C para fazermos a leitura, mas conectando e reconectando os componentes não foi possível que o Raspberry detectasse através do i2cdetect, Figura 31. Sem essa detecção, não é possível fazer leituras num determinado endereço.

Um terceiro teste então foi realizado, com ele, foi comprado um conversor I2C para o display de LCD. Como o LCD já estava comprovadamente funcionando, poderiamos isolar a possibilidade de um problema no componente. Ainda era possível que o conversor em si viesse avariado, mas no tempo que tinhamos, não havia como nos certificarmos mais do que isso. O teste foi feito e ainda assim não conseguimos fazer a detecção através da Raspberry Pi do componente conectado ao I2C.

Com esse último resultado, tinhamos algumas opções para o erro: ou era um problema de hardware na Raspberry Pi utilizada pela equipe, ou um problema do sistema operacional Raspian. Foi aventado a possibilidade de se comprar ou conseguir um Arduino, bem como o empréstimo de outra placa Raspberry. Por problemas da pandemia, o projeto foi montado todo em São Paulo, o que dificultaria o empréstimo da placa, e nenhum membro da equipe possuia um Arduino. Acabou então que a questão ficou em aberto, e migramos a solução para os exercícios de cor e forma para a solução da Azure.

4.2 APLICATIVO

O software desenvolvido para o responsável por controlar os exercícios da criança consiste em um aplicativo para android. A Figura 32 representa o inicio do aplicativo, onde o usuário tem as opções de cadastrar e avançar, porém a opção de avançar só funcionará com pelo menos uma criça cadastrada e selecionada. Ao selecionar a opção cadastrar, o usuário é conduzido a tela de cadastro representado na Figura 33.

A tela de cadastro permite o usuário cadastrar o nome da criança e escolher o slot onde ela será salva, sendo possível escolher os slots de 1 a 4. Ao apertar o botão para cadastrar, o usuário será retornado a tela de inicio, mas agora com uma criança cadastrada, como na figura 34.

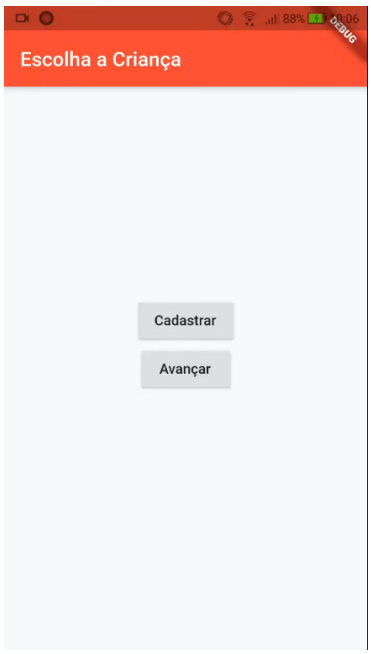

Figura 32: Tela inicial sem crinça cadastrada: Aplicativo Fonte: Autoria Própria

Figura 33: Tela Cadastro: Aplicativo Fonte: Autoria Própria

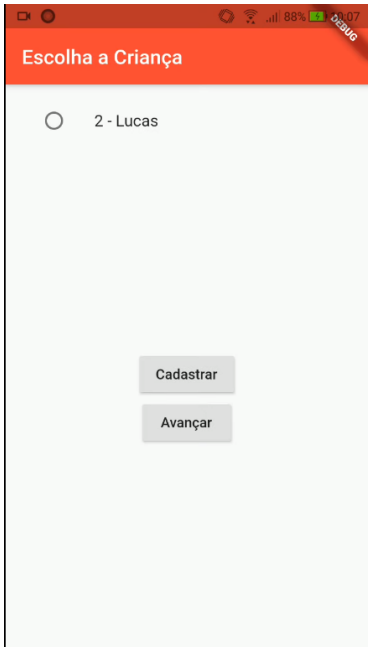

Figura 34: Tela inicial com criança cadastrada: Aplicativo Fonte: Autoria Própria

Após retornar a tela de inicio com a criança cadastrada, devemos seleciona-lá, como na figura 35, para que possamos avançar para a home, que será a próxima tela.

Chegando na Home é necessario se conectar ao raspberry usando o botão 'Conectar', com isso o app conseguirá identificar o ip a ser feito as requisições http, se tudo ocorrer corretamente, o botão deve sumir após ser pressionado. Após isso, o usuário se depara com varias opções, sendo elas configurações, para definir a dificuldade dos exercícios, customização de exercícios, histórico da criança e selecionar o exercício desejado. Como padrão, é disponibilizada uma rotina de exercícios contendo todos os quatro exercícios. Tudo isso pode ser observado na Figura 36.

A tela de configuração de dificuldades apresenta os quatro possíveis exercícios e as dificuldades a serem selecionadas para a criaça, como mostra a Figura 37. Para sair dessa tela é necessario pressionar o botão voltar e o usuário retornará a Home.

A tela de customização dos exercícios é apresentada de forma que o usuário ativa ou desativa cada um dos quatro exercícios. Assim como as crianças, os exercicios possuem 4 slots para serem salvos também, de forma que quando for selecionado um slot existente, a rotina de execício será sobrescrevido. A Figura 38 apesenta essa tela.

Por fim, ao lado de cada rotina de exercício há um botão para o início da mesma. Ao pressionar o botão, o app gera um request http para o Raspberry, começando assim os exercícios. O usuário é redirecionado para uma tela de espera, como mostrado na Figura X onde após

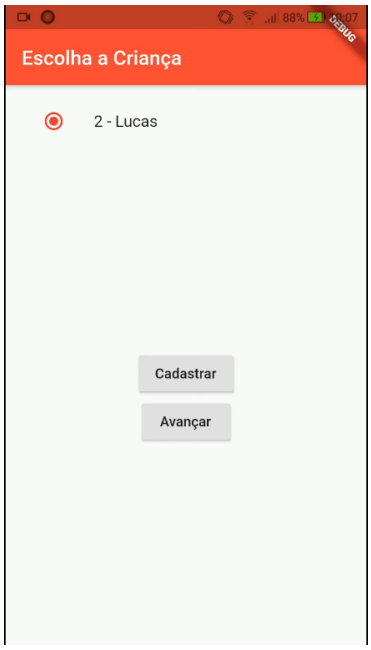

Figura 35: Tela inicial com criança selecionada: Aplicativo Fonte: Autoria Própria

Figura 36: Tela Home: Aplicativo Fonte: Autoria Própria

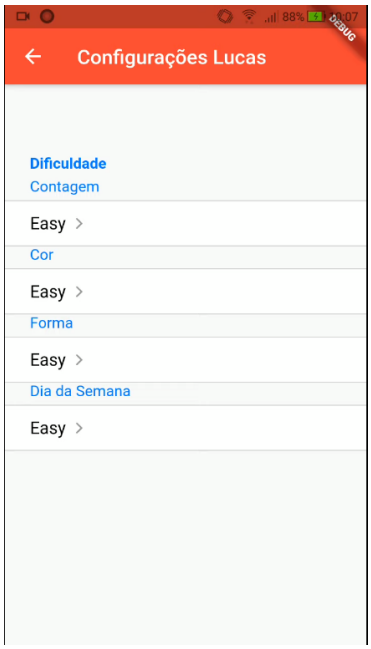

Figura 37: Tela de Seleção de Dificuldade: Aplicativo Fonte: Autoria Própria

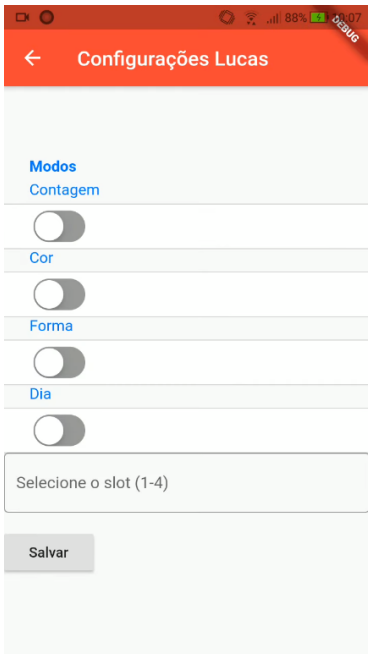

Figura 38: Tela de Customização da Rotina de Execício: Aplicativo Fonte: Autoria Própria

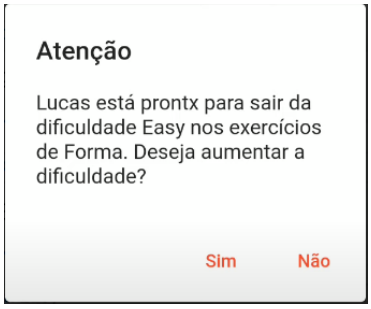

Figura 39: Popup para aumento da dificuldade: Aplicativo Fonte: Autoria Própria

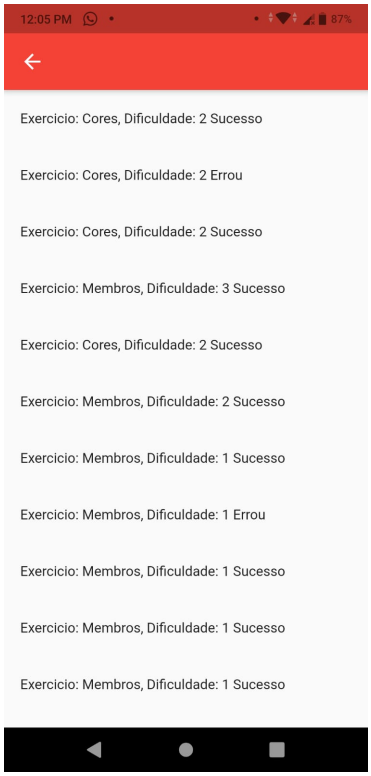

Figura 40: Tela Histórico: Aplicativo Fonte: Autoria Própria

terminada a rotina, será retornada a porcentagem de acertos da criança em cada exercício. Se a criança acertar mais de 60%, surgirá um popup perguntando se o usuário deseja aumentar o nivel de dificuldade, como mostrado na Figura 39.

Após qualquer exercício ser completado, o usuário poderá consultar o historico de exercícios feitos pela criança, com as informações de: Nome, Exercício, Dificuldade, Porcentagem de acerto e Data/Hora. A Figura 40 apresenta a tela de histórico.

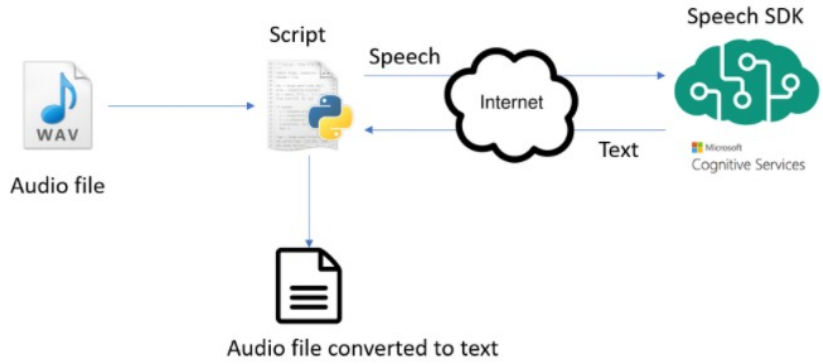

Figura 41: Azure Cognitive Services: Speech-to-Text Schema Fonte: Autoria Própria

4.3 AZURE

A Azure foi a solução selecionada para auxiliar no processo tanto de handling do reconhecimento visual, com formas e cores, quanto a função de speech-to-text, necessária para avaliar as skill de Speaking e Listening da criança.

Utilizando da Azure CLI, criamos resources pautados em módulos do Azure Cognitive Services como o módulo de Speech que utilizamos para transformar áudios de frases obtidos atráves do microfone no PI em texto para comparar com a resposta esperada. Como pode ser visto na Figura 41.

Outro módulo utilizado foi o de Custom Vision que nos permite treinar a detecção de padrões em imagens e mapear, dentro às tags associadas ao treinamento. Usando de datasets, tanto criados por nós com as imagens das formas, quanto coletados do kaggle.com de cor e forma para criar correlações entre os imagens coloridas e as tags, tanto de cor quanto de forma para conseguir identificar separadamente em diferentes endpoints da api. Figura 42.

Posteriormente através da API podemos mandar imagens obtidas do Raspberry PI a fim de avaliar uma predição do que representa aquela imagem - utilizando do módulo de Custom Vision - Predictions, Figuras 43 e 44.

Para garantir a comunicação segura com a API da Azure primeiramente temos o hardening das resources publicas no ambiente Cloud e posteriormente, seguindo as boas práticas, autenticação via RSA-keypair contra a API para validação das predições. A comunicação com a Azure é feita totalmente em HTTPS e por fim, com as respostas, tanto

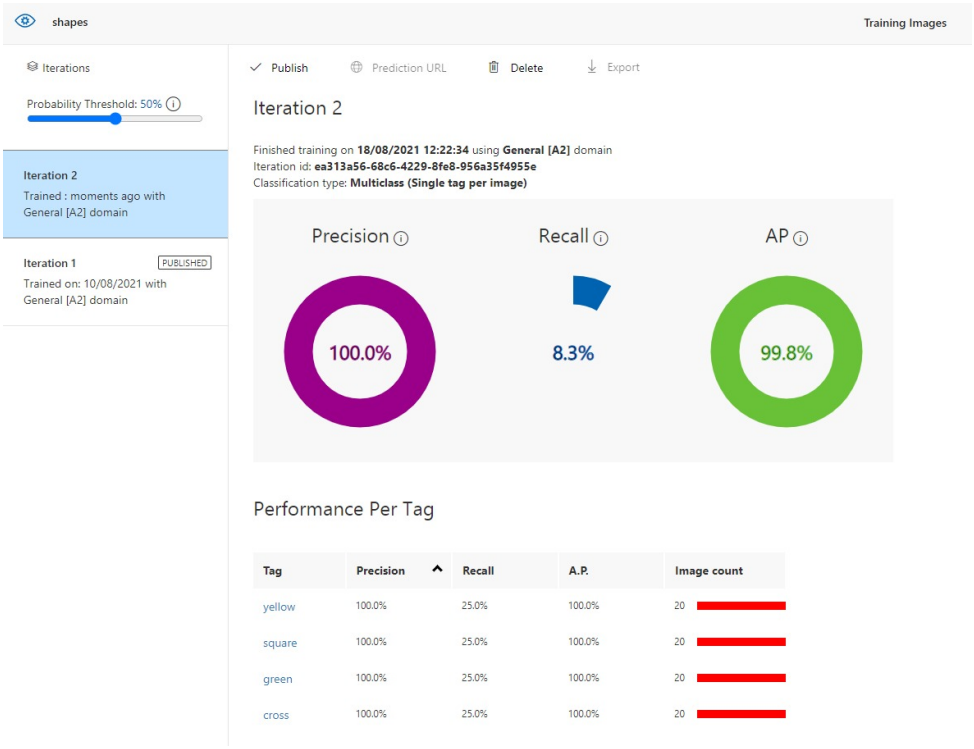

Figura 42: Azure Cognitive Services: Training Quality Fonte: Autoria Própria

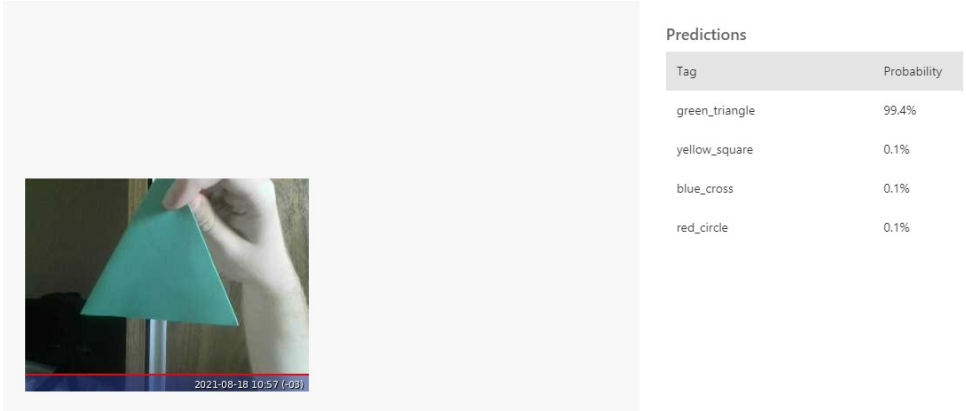

Figura 43: Azure Cognitive Services: Prediction Affinity Fonte: Autoria Própria

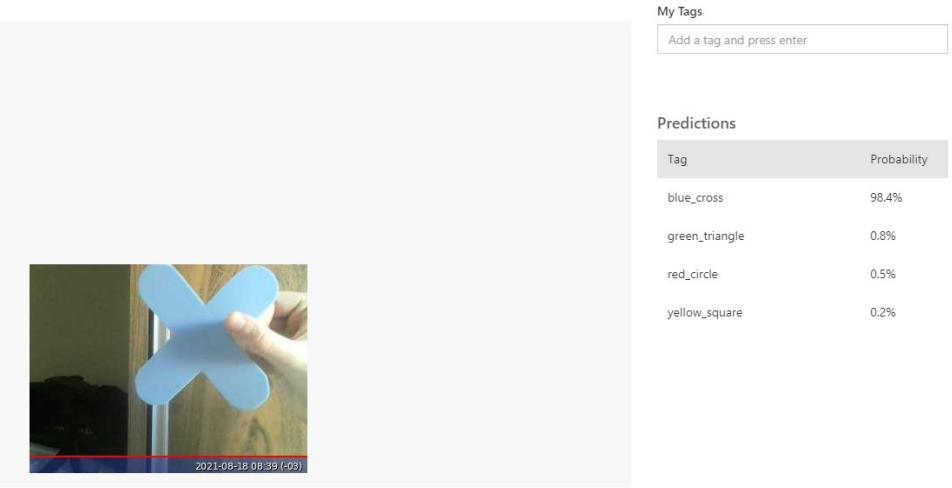

Figura 44: Azure Cognitive Services: Prediction Affinity Fonte: Autoria Própria

da computação visual como do reconhecimento de palavras abstraído para a núvem, usamos apenas essa interface para comparar as respostas obtidas da API com o resultado esperado da criança.Resultados com baixa precisão são ignorados e refeita a foto/avaliação.

4.4 INTEGRAÇÃO

A integração se iniciou com a finalização da montagem do robô. Como dito anteriormente, a montagem final ainda conta com o protoboard. Temos então na Figura 45 uma visão de cima da construção final, na Figura 46 uma visão traseira da construção final e na Figura 47 uma visão frontal.

Devido à falha da comunicação do aplicativo flutter com Bluetooth e a fim de reduzir uma conexão extra que não era necessária uma vez que tanto o ZéColberry a e o aplicativo se encontrarão sob a mesma rede Wi-fi foi feita uma comunicação via HTTPS entre o aplicativo e o Raspberry PI, que contrempla um servidor Flask para tomada de decisões e execução das rotinas O servidor do raspberry aguarda por conexões do aplicativo, explicitando qual atividade será executada e o Raspberry fica responsável em executar as diferentes funções controladoras das partes do ZéColberry.

Cada endpoint da API é segmentada através do tipo de exercício sendo possível indicar o nível de atividade na requisição e assim executar a procedure requisitada, e por fim, ao término da execução, reunir as informações de execução ou possíveis erros, e reunir as informações para informar o APP do progresso da atividade.

Na parte do aplicativo foi usado o mesmo feito para a simulação com algumas alterações para melhor demonstração. Ao invés de rotinas de exercícios, foi pré-definido um

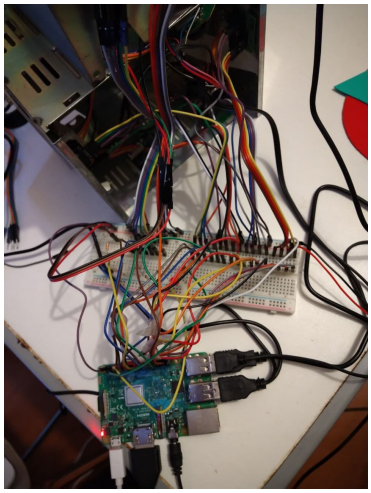

Figura 45: Construção Final: Visão de Cima Fonte: Autoria Própria

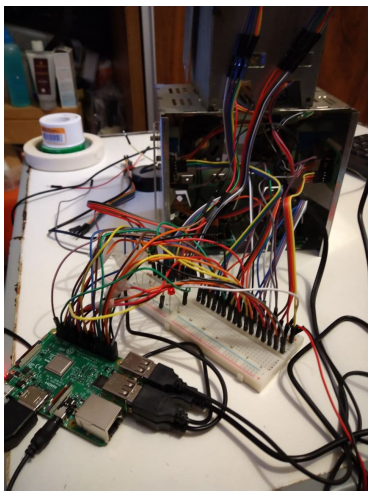

Figura 46: Construção Final: Visão Traseira Fonte: Autoria Própria

Figura 47: Construção Final: Visão Frontal Fonte: Autoria Própria

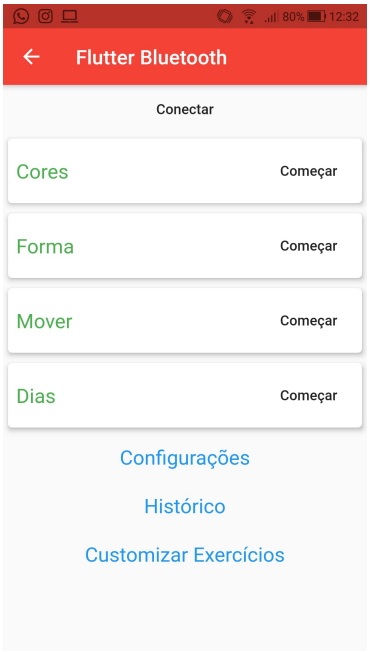

Figura 48: Tela Home após Integração: Aplicativo Fonte: Autoria Própria

exercício de cada para agilizar o processo, como pode ser observado na Figura 48.

Foi também adicionado uma tela para enviar a requisição http para o Raspberry com sua dificuldade, como mostrado na Figura 49

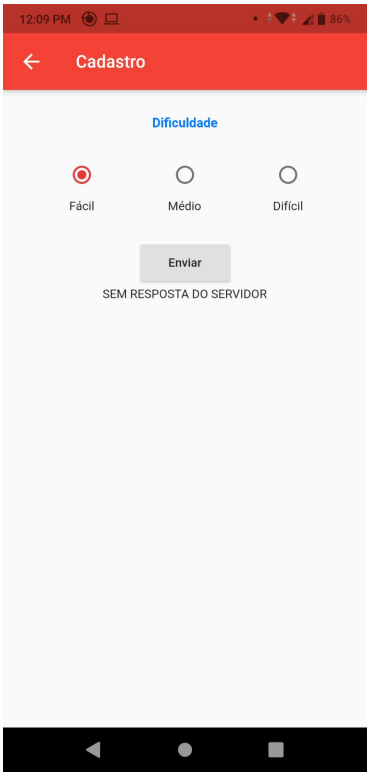

Figura 49: Tela de Envio: Aplicativo Fonte: Autoria Própria

5 CRONOGRAMA E CUSTOS DO PROJETO

5.1 CRONOGRAMA

Nesta sessão temos o cronograma devidamente subdividido nas sessões do projeto. O cronograma completo está disponível no blog do projeto (GUERRERO et al.,).

5.2 CUSTOS

Integrante	Tempo Previsto	Tempo Gasto
Gabriel	74	83
Leonardo	83	89
Lucas	80	67
Todos	54	55
Total	291	294

Tabela 4: Tempo

Atividade	Responsável	Assistente	Status	Duração	Erro	Total	Tempo Gasto	Data Início	Data Limite
Ideia Inicial	Todos		Feito	4	1,2	5,2	5	16/06	23/06
Especificação	Todos		Feito	8	2,4	10,4	10	23/06	30/06
Criação do Blog	Gabriel	Lucas	Feito	2	0,6	2,6	1	30/06	14/07
Requisitos	Gabriel	Leonardo	Feito	5	1,5	6,5	6	23/06	30/06
Análise de Riscos	Leonardo	Lucas	Feito	3	0,9	3,9	3	30/06	07/07
Criação de Cronograma	Todos		Feito	5	1,5	6,5	6	30/06	07/07
Orçamento	Lucas	Gabriel	Feito	3	0,9	3,9	2	30/06	07/07
Fazer Diagramas	Todos		Feito	6	1,8	7,8	5	23/06	07/07
Escrever Posts do Blog	Todos		Feito	4	1,2	5,2	2	30/06	07/07
Preparar a Apresentação	Lucas	Gabriel	Feito	5	1,5	6,5	4	16/06	07/07
Compra dos Materiais	Lucas	Leonardo	Feito	3	0,9	3,9	2	30/06	07/07
Pré-Projeto									
Total Time				48	14,4	62,4	46		

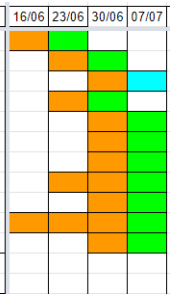

Figura 50: Cronograma: Pré-Projeto Fonte: Autoria Própria

Atividade	Responsável	Assistente	Status	Duração	Erro	Total	Tempo Gasto	Data Início	Data Limite
Construção do Esqueleto	Gabriel		Finalizado	5	1,5	6,5	6	07/07	21/07
Construção dos Membros móveis	Gabriel		Finalizado	3	0,9	3,9	3	07/07	21/07
Construção das Formas Geométricas	Gabriel		Finalizado	3	0,9	3,9	1	07/07	21/07
Pintura das Formas Geométricas	Gabriel		Finalizado	1	0,3	1,3	1	07/07	21/07
Teste da Webcam	Gabriel	Leonardo	Finalizado	2	0,6	2,6	3	14/07	28/07
Teste do Sensor de Presença	Gabriel	Leonardo	Finalizado	2	0,6	2,6	1	14/07	28/07
Teste do Sensor de Cor	Gabriel	Lucas	Finalizado	2	0,6	2,6	5	14/07	28/07
Teste dos Motores	Gabriel	Lucas	Finalizado	2	0,6	2,6	4	14/07	28/07
Teste dos Speakers	Gabriel	Lucas	Finalizado	2	0,6	2,6	1	14/07	28/07
Teste do LCD	Gabriel	Lucas	Finalizado	2	0,6	2,6	2	14/07	28/07
Acoplamento dos Motores	Gabriel		Finalizado	3	0,9	3,9	1	14/07	28/07
Definição da Pinagem	Gabriel		Finalizado	3	0,9	3,9	3	14/07	28/07
Definição da Alimentação	Todos		Finalizado	3	0,9	3,9	3	07/07	14/07
Implementação da Alimentação	Gabriel	Lucas	Finalizado	6	1,8	7,8	4	14/07	11/08
Produção da Configuração Final	Gabriel		Finalizado	7	2,1	9,1	7	14/07	11/08
Escrever Posts do Blog	Todos		Finalizado	4	1,2	5,2	5	07/07	28/07
Preparar a Apresentação	Lucas	Gabriel	Finalizado	5	1,5	6,5	5	21/07	28/07
Hardware/Mecânica									
Total Time				60	18	78	61		

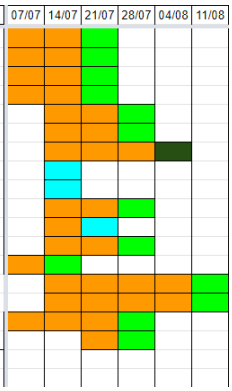

Figura 51: Cronograma: Mecânica e Hardware Fonte: Autoria Própria

Atividade	Responsável	Assistente	Status	Duração	Erro	Total	Tempo Gasto	Data Início	Data Limite
Total Time				60	18	78	61		
Instalação do Sistema Operacional	Gabriel	Leonardo	Finalizado	3	0,9	3,9	2	07/07	07/07
Estudo da Solução Azure	Leonardo		Finalizado	10	3	13	2	30/06	07/07
Criação dos diagramas para o Firmware	Gabriel		Finalizado	3	0,9	3,9	5	30/06	07/07
Implementação Inicial Reconhecimento Facial - Azure Setup	Leonardo	Lucas	Finalizado	10	3	13	8	07/07	14/07
Teste do Reconhecimento Facial na Rasp	Leonardo	Gabriel	Finalizado	3	0,9	3,9	2	14/07	21/07
Implementação Inicial Reconhecimento de Voz - Azure Setup	Leonardo	Lucas	Finalizado	10	3	13	8	14/07	21/07
Teste do Reconhecimento de Voz na Rasp	Leonardo	Gabriel	Finalizado	4	1,2	5,2	1	21/07	28/07
Implementação da Fala do Robô	Leonardo	Lucas	Finalizado	10	3	13	15	28/07	04/08
Teste da Fala do Robô	Gabriel	Leonardo	Finalizado	3	0,9	3,9	3	04/08	11/08
Refinamento da resposta da API Audio/Vid	Leonardo	Gabriel	Finalizado	8	2	10	12	28/07	11/08
Criação dos fluxos de Assertividade de Audio	Leonardo	Gabriel	Finalizado	1	.5	1.5	2	28/07	11/08
Criação dos fluxos de Assertividade de Video	Leonardo	Gabriel	Finalizado	1	.5	1.5	4	28/07	11/08
Teste e validação dos fluxos de audio	Leonardo	Lucas	Em Progresso	3	1	4	4	04/08	11/08
Teste e validação dos fluxos de Video	Leonardo	Lucas	Finalizado	3	1	4	4	04/08	11/08
Implementação dos sensores de cor e presença	Leonardo	Gabriel	Finalizado	4	1,2	5,2	3	04/08	11/08
Implementação da conexão Bluetooth	Leonardo	Lucas	Em Progresso	5	1,5	6,5	12	04/08	11/08
Implementação das Rotinas de Exercícios	Gabriel	Lucas	Finalizado	4	1,2	5,2	4	28/07	04/08
Implementação dos Níveis de Dificuldade	Lucas	Gabriel	Finalizado	3	0,9	3,9	2	28/07	04/08
Implementação da Resposta a Falta de Conexão	Leonardo	Gabriel	A fazer	5	1,5	6,5	5	04/08	11/08
Implementação do controle dos membros	Leonardo		Finalizado	3	0,9	3,9	4	14/07	21/07
Escrever Posts do Blog	Todos		A fazer	4	1,2	5,2	4	07/07	11/08
Preparar a Apresentação	Lucas	Gabriel	A fazer	5	1,5	6,5	2	11/08	11/08
Firmware									
Total time				105	31,5	136,5	108		

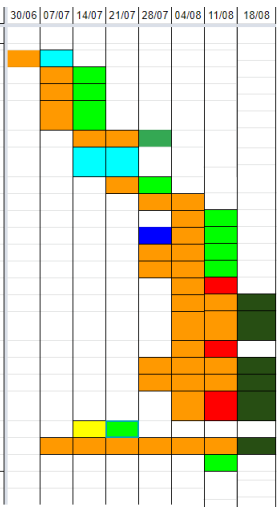

Figura 52: Cronograma: Firmware e Azure Fonte: Autoria Própria

Atividade	Responsável	Assistente	Status	Duração	Erro	Total	Tempo Gasto	Data Início	Data Limite	14/07	21/07	28/07	04/08	11/08	18/08
Estudo do Desenvolvimento de Aplicativos Flutter	Lucas		Feito	6	1,8	7,8	5	07/07	14/07						
Montar Telas e Esqueleto do Aplicativo	Lucas	Leonardo	Feito	3	0,9	3,9	4	14/07	16/07						
Fazer a Conexão via Bluetooth	Lucas	Gabriel	Feito	5	1,5	6,5	5	16/07	20/07						
Implementar Metodos e Variáveis de Cada Tela	Lucas		Feito	4	1,2	5,2	4	20/07	23/07						
Salvar Histórico das Crianças	Lucas		Feito	3	0,9	3,9	3	23/07	26/07						
Cadastrar as Rotinas de Exercícios Pré-Programadas	Lucas	Gabriel	Feito	6	1,8	7,8	2	26/07	29/07						
Possibilitar o Cadastro e Personalização das Rotinas de Exercício	Lucas		Feito	5	1,5	6,5	5	29/07	31/07						
Implementar Algoritmo que Informa se a Criança está apta a Aumentar Dificuldade	Lucas	Leonardo	Feito	7	2,1	9,1	4	31/07	04/08						
Escrever Posts do Blog	Todos		Feito	4	1,2	5,2	4	07/07	04/08						
Preparar a Apresentação	Lucas	Gabriel	Feito	5	1,5	6,5	4	04/08	04/08						
Integração Aplicativo -> Raspberry	Lucas	Leonardo	Feito	7	2,1	9,1	9	04/08	11/08						
Aplicativo															
Total Time				55	16,5	71,5	40								

Figura 53: Cronograma: Aplicativo Fonte: Autoria Própria

Atividade	Responsável	Assistente	Status	Duração	Erro	Total	Tempo Gasto	Data Início	Data Limite	28/07	04/08	11/08	18/08
Finalizar Testes com o Protótipo	Todos		Finalizado	8	2,4	10,4	8	11/08	18/08				
Finalizar Relatório Técnico	Gabriel		Finalizado	6	1,8	7,8	14	11/08	18/08				
Escrever Posts do Blog	Todos		Finalizado	4	1,2	5,2	3	11/08	18/08				
Preparar a Apresentação	Lucas	Gabriel	Finalizado	5	1,5	6,5	2	18/08	18/08				
Demonstração Protótipo/Relatório Técnico													
Total Time				23	6,9	29,9	27						

Figura 54: Cronograma: Demonstração Protótipo e Relatório Fonte: Autoria Própria

Componente	Custo Unitário	Unidades	Custo Total
Raspberry Pi 3 Modelo B	330,00	1	330,00
Servo-motor	20,00	4	80,00
Sensor de Cor e Gestos APDS 9960	29,00	2	58,00
Sensor de Movimento PIR DYP-ME003	18,90	1	18,90
Webcam USB com Microfone Integrado W18	64,99	1	64,99
Speaker P2	30,00	1	30,00
		Total	501,89

Tabela 5: Orçamento

6 CONCLUSÕES

6.1 CONCLUSÕES

Uma conclusão importante do projeto é que as dificuldades pandêmicas provaram-se muito mais desafiadoras do que o inicialmente previsto. Estarmos todos trabalhando remotamente, com centenas de quilometros de distância entre os membros se provou um desafio bastante importante.

Também é relevante apontar um problema de comunicação entre a equipe, que acabou por comprometer o andamento do projeto e atrasar alguns prazos e entregáveis. Outro destaque relevante foi a dificuldade de integração entre as partes, o desafio de se aplicar a solução da Azure, aliado ao Raspberry Pi e um aplicativo criado no Flutter se demonstrou maior do que a equipe estimava.

Esses e alguns problemas de componentes fez com que o projeto, embora funcional, não alcançasse todos os objetivos propostos, requerendo um tempo extra para que os problemas fossem corrigidos e o projeto concluído em sua totalidade.

No entanto, considerando a complexidade do projeto e as restrições impostas pela pandemia e o semestre reduzido, o grupo considera que o resultado final foi bastante positivo.

6.2 TRABALHOS FUTUROS

O primeiro destaque importante a ser feito para trabalhos futuros é a construção das placas de circuito integrado. O protótipo atual foi construído utilizando um protoboard, o que não representa um cenário ideal ou de projeto finalizado, uma vez que as fiações não estão fixas e podem acabar se soltando com o movimento.

Outro trabalho futuro importante é a aquisição ou empréstimo de um Arduino ou outra Raspberry Pi. Esses componentes seriam utilizados para que o projeto possa enfim utilizar, ou ao menos ter uma maior averiguação, sobre o problema com a interface I2C. Além disso, o

Arduino poderia ser utilizado para fazer a leitura do Sensor de Cor e Gestos, assim fazendo com que o projeto atendesse ao requisito inicial de fazer a leitura dos exercícios através dele ao invés da Webcam com Azure.

Um terceiro desenvolvimento importante seria a implementação dos exercícios de contagem. Os exercícios de contagem foram inicialmente propostos, mas acabaram não sendo desenvolvidos a tempo para a apresentação deste relatório e projeto final.

De implementações que acabaram fora do resultado final, ainda podemos destacar a implementação do Bluetooth. Por uma incompatibilidade entre o protocolo de Bluetooth da Raspberry Pi Model B e o utilizado pelo aplicativo Flutter, acabou sendo inviável a aplicação do protocolo na solução. A resposta para isso seria a compra do módulo bluetooth para a Raspberry e então teriamos o funcionamento correto desta parte do projeto.

Os algoritmos de detecção de fala, bem como de formas e cores poderiam ser refinados, com um filtro maior entre o que é aceito ou não. Por exemplo, na solução atual, maneirismos da fala são detectados como erros, num projeto real, seriam simplesmente desconsiderados, especialmente levando em conta que o público alvo são crianças.

REFERÊNCIAS

AVAGO TECHNOLOGIES. **APDS-9960: Digital Proximity, Ambient Light, RGB and Gesture Sensor**. [S.l.], 11 2013.

COOK, J. S. **The Right Tool for the Job: Active and Passive Infrared Sensors**. Disponível em: <https://www.arrow.com/en/research-and-events/articles/understanding-active-and-passive-infrared-sensors>.

CRYSTALFONTZ AMERICA, INC. **LCD-1602A**. [S.l.].

GONSALVEZ FELIPE E PUGA, V. Motor de passo petele. **Universidade Federal Fluminense Centro Tecnológico Escola de Engenharia Curso de Engenharia de Telecomunicações Programa de Educação Tutorial Grupo PET-Tele**, 2008.

GUERRERO, G.; REIS, L.; RIBEIRO, L. Disponível em: <https://www.notion.so/Z-ColBerry-17af6c4cd680403dba435f2bfbc22bec>.

JúNIOR, A. L. da C. **Redes Sem Fio: Bluetooth**. Disponível em: <https://www.teleco.com.br/tutoriais/tutorialredespbaid/>.

OPENIMPULSE. **DYP-ME003 PIR Sensor Module Datasheet**. Disponível em: <https://sicccIber.com.br/wp-content/uploads/2020/06/FTC-PIR.pdf>.

ROTOTRON. **LCD Display Tutorial for Raspberry Pi**. Disponível em: <https://www.rototron.info/lcd-display-tutorial-for-raspberry-pi/>.