CENTRO PAULA SOUZA
ESCOLA TÉCNICA PROFESSOR MASSUYUKI KAWANO
Técnico em Redes de Computadores

Enzo Cruz Modena
Gabriel Roberto Rodella de Assis
Gustavo Mendes de Oliveira
Samuel Julio do Nascimento Batista

ARDUNANNY: DESENVOLVIMENTO DE UM SISTEMA DE MONITORAMENTO UTILIZANDO TECNOLOGIA ESP32 E BLUETOOTH PARA AUXILIAR PAIS COM DEFICIENCIA AUDITIVA

Tupã - SP
2023

CENTRO PAULA SOUZA
ESCOLA TÉCNICA PROFESSOR MASSUYUKI KAWANO
Técnico em Redes de Computadores

Enzo Cruz Modena
Gabriel Roberto Rodella de Assis
Gustavo Mendes de Oliveira
Samuel Julio do Nascimento Batista

ARDUNANNY: DESENVOLVIMENTO DE UM SISTEMA DE MONITORAMENTO UTILIZANDO TECNOLOGIA ESP32 E BLUETOOTH PARA AUXILIAR PAIS COM DEFICIENCIA AUDITIVA

Trabalho de Conclusão de Curso apresentado ao Curso Técnico em Técnico em Redes de Computadores da Etec PROFESSOR MASSUYUKI KAWANO, orientado pelo Prof. Caroline de Oliveira Ferraz, como requisito parcial para obtenção do título de técnico em Redes de Computadores;

Menção do Trabalho: MB

Tupã - SP

2023

GOVERNO DO ESTADO
DE SÃO PAULO

CENTRO PAULA SOUZA
ESCOLA TÉCNICA PROFESSOR MASSUYUKI KAWANO
Técnico em Redes de Computadores

Enzo Cruz Modena
Gabriel Roberto Rodella de Assis
Gustavo Mendes de Oliveira
Samuel Julio do Nascimento Batista

ARDUNANNY: DESENVOLVIMENTO DE UM SISTEMA DE MONITORAMENTO UTILIZANDO TECNOLOGIA ESP32 E BLUETOOTH PARA AUXILIAR PAIS COM DEFICIENCIA AUDITIVA

Apresentação para a Banca em caráter de validação do título de Técnico em Redes de Computadores.

BANCA EXAMINADORA:

Prof. Esp. Caroline de Oliveira Ferraz
Orientadora

Prof. (a).
Avaliador (a) Anderson Tukiyama Berengue

Prof. (a).
Avaliador (a) Joel Coutinho de Souza

Tupã, 20 de Junho de 2023

AGRADECIMENTOS

Em primeira instância, agradecemos aos pais com deficiência auditiva por demonstrarem imensa compreensão e boa vontade ao fornecerem informações sobre suas necessidades. Isso nos permite abordar um problema recorrente ao meio dessas pessoas, assim trazendo uma possível solução rentável.

Agradecemos à professora e orientadora Caroline de Oliveira Ferraz, pela paciência e pela orientação durante toda a elaboração do trabalho.

Agradecemos ao professor Joel Coutinho de Souza, pelos ensinamentos apresentados ao longo do curso, pelas ideias e pelos importantes conselhos que serviram para o nosso aprendizado.

Agradecemos ao professor e coordenador do Redes, Anderson Tukiyama Berengue, pelos aprendizados durante todo o curso.

A todos que contribuíram diretamente e indiretamente para a conclusão do curso.

EPÍGRAFE

“O desejo que me guia em tudo o que faço é o desejo de aproveitar as forças da natureza a serviço da humanidade”.

(Nikola Tesla)

RESUMO

ARDUNANNY: DESENVOLVIMENTO DE UM SISTEMA DE MONITORAMENTO UTILIZANDO TECNOLOGIA ESP32 E BLUETOOTH PARA AUXILIAR PAIS COM DEFICIENCIA AUDITIVA

A deficiência auditiva é uma condição que afeta milhões de pessoas no Brasil e no mundo, gerando dificuldades em várias atividades diárias, inclusive no cuidado com os filhos. Diante dessa situação, este trabalho de automação tem como objetivo desenvolver uma solução inovadora para auxiliar pais e/ou responsáveis com deficiência auditiva no monitoramento de bebês. A proposta consiste em um sistema de monitoramento em tempo real baseado em luzes, em vez de sons, utilizando um microcontrolador ESP32 e o protocolo MQTT. Adicionalmente, será estabelecida a comunicação entre os microcontroladores ESP32 por meio do protocolo MQTT, utilizando o broker do Adafruit. Essa integração permitirá o envio e recebimento de informações cruciais para o monitoramento dos bebês. Além disso, os dados serão passíveis de serem analisados no Dashboard do Adafruit e estarão disponíveis no aplicativo móvel. Ao adotar essa abordagem, o sistema proporciona benefícios significativos aos pais, oferecendo uma ferramenta acessível e de baixo custo. Tendo isso em vista, o objetivo principal é detectar e notificar a ocorrência de choro por parte da criança, mesmo quando os pais estão ausentes ou dormindo. Dessa forma, o sistema fornece uma solução simples, eficiente e que se integra facilmente aos dispositivos móveis, como celulares. A metodologia aplicada neste trabalho é qualitativa, com destaque para a realização da entrevista com os pais deficientes auditivos conhecidos por um membro do grupo que enfrentam o desafio de não saberem quando seus filhos precisam de atenção. Com essa pesquisa e desenvolvimento, espera-se contribuir para a melhoria da qualidade de vida dos pais com deficiência auditiva, fornecendo-lhes uma ferramenta de monitoramento eficaz, acessível e fácil de usar, promovendo, assim, a inclusão e a autonomia dessas famílias.

Palavras – chave: Internet das Coisas; ESP32, Comunicabilidade; Tecnologia Assistiva.

LISTA DE FIGURAS

LISTA DE ABREVIATURAS E SIGLAS

ESPE	Eletronic Stability Program
MQTT	Message Queuing Telemetry Transport
IBGE	Instituto Brasileiro de Geografia e Estatística
LED	Light-Emiting Diode
MAC	Media Access Control
IOT	Internet das Coisas
IDE	Ambiente de desenvolvimento integrado
GND	Graduated neutral density filter
USB	Universal Serial Bus
APK	Android Package Kit
VCC	Tensão em corrente contínua
LCD	Liquid Crystal Display

Sumário

1 INTRODUÇÃO

O objetivo desta iniciativa é solucionar problemas comuns enfrentados por casais com filhos e deficiência auditiva. Quando os pais estão dormindo, como durante a noite, eles não conseguem ouvir seus filhos chorando, tornando-se necessária a existência de um sistema que alerte quando os bebês precisam de atenção. De acordo com dados do Censo do IBGE de 2010 (IBGE, 2010), aproximadamente 8% dos brasileiros, o que totaliza cerca de 45 milhões de pessoas, possuem algum tipo de perda auditiva. No entanto, as soluções disponíveis atualmente no mercado são caras e inacessíveis para a maioria dos casais.

> No caso dos surdos, a maioria dos produtos desenvolvidos para eles tentam trazê-los de alguma forma a “normalidade”, a realidade das pessoas que ouvem (exemplo dos aparelhos auditivos); quando seus sentidos mais apurados (visão, tato, a expressão corporal) e sua forma de expressão – a língua de sinais – acabam sendo deixadas em segundo plano. Incluir a pessoa surda é também valorizar sua cultura, sua forma de expressão, e, os designers, ao desenvolverem produtos podem enriquecê-los com essas características. (Lopez et al, 2016).

Diante desse problema, o principal objetivo deste projeto é desenvolver uma solução mais acessível e econômica para pais surdos. Para alcançar esse objetivo, o grupo de pesquisa conduziu estudos, identificou componentes de baixo custo e optou por usar o ESP32 juntamente com outros componentes acessíveis. Essa escolha foi fundamental para viabilizar um projeto final com um custo mais baixo, tornando-o mais acessível para casais com menor poder aquisitivo.

Além disso, foi realizada uma análise dos componentes utilizados, incluindo a análise do código correspondente, para resolver incompatibilidades que surgiram durante o desenvolvimento do projeto. Por fim, foi realizada a montagem física do projeto com os ajustes necessários para garantir seu correto funcionamento.

Considerando a prevalência da perda auditiva e a necessidade de soluções acessíveis, é justificado propor um monitor de bebês de baixo custo que se conecte ao celular dos pais surdos. A implementação desse projeto pode contribuir significativamente para melhorar a qualidade de vida dessas famílias, permitindo que fiquem tranquilos ao saberem quando seus filhos precisam de atenção, mesmo com a limitação auditiva.

Dessa forma, esta pesquisa contribui para a área de tecnologia assistiva, ao oferecer uma solução inovadora e acessível para um problema enfrentado por uma

parcela significativa da população. Além disso, ao promover a inclusão e autonomia dos pais com deficiência auditiva, o projeto busca trazer benefícios tanto para a sociedade como um todo quanto para a comunidade científica interessada em tecnologias inclusivas.

Portanto, a estrutura deste trabalho está organizada da seguinte forma: inicialmente, será apresentado o embasamento teórico sobre os principais aspectos que compõem o projeto final, como o Protocolo MQTT e outros. Em seguida, será apresentada a metodologia utilizada e os recursos utilizados para a confecção do projeto final. Posteriormente, serão apresentadas as características do protótipo inicial utilizado como modelo, o desenvolvimento do aplicativo mobile e o projeto final em si, juntamente com os resultados e considerações finais obtidos ao longo de todo o processo. Com essa estrutura, espera-se abordar de forma abrangente e sistemática os objetivos propostos, contribuindo para o avanço do conhecimento na área e oferecendo uma solução viável e acessível para casais deficientes auditivos com filhos.

2 OBJETIVOS

O objetivo central da presente pesquisa é desenvolver um mecanismo capaz de detectar e notificar a ocorrência de choro por parte de uma criança, quando os pais com deficiência auditiva estiverem ausentes ou dormindo. A iniciativa proposta visa suprir uma necessidade específica desse grupo de pais, que enfrentam uma limitação na detecção de sons e podem ter dificuldades para perceber o choro de seus filhos. Com o desenvolvimento desse dispositivo, busca-se oferecer uma solução prática e eficiente para garantir a segurança e o bem-estar da criança, bem como proporcionar aos pais deficientes auditivos maior tranquilidade e confiança no cuidado com suas crianças.

2.1 Objetivo geral

Desenvolvimento de um sistema de detecção de som utilizando dois microcontroladores ESP32, onde um microcontrolador com módulo de som conectado ao roteador captura a alerta sonoro e envia um sinal para o segundo microcontrolador ESP32, conectado à uma lâmpada.

2.2 Objetivos específicos

- Elaboração de um sistema simples, com custo reduzido e de fácil conexão.
- Criação de um aplicativo móvel para receber alertas sonoros.
- Realização da integração do ESP32 com o aplicativo móvel utilizando Bluetooth, facilitando o acesso dos pais aos alertas que serão emitidos para o aplicativo quando o ESP32 receptor captar um alerta sonoro.
- Estabelecer a comunicação entre os ESP32 utilizando o broker do ADAFRUIT, permitindo assim o envio e recebimento de informações por meio do roteador para a análise do Dashboard do Adafruit e do aplicativo mobile.

3 EMBASAMENTO TEÓRICO

Neste capítulo, serão apresentados os principais conceitos básicos e necessários para o entendimento e elaboração deste trabalho.

3.1 Internet das Coisas

A Internet das Coisas (IoT) desempenha um papel fundamental ao conectar objetos comuns à Internet, concedendo-lhes capacidades de comunicação, processamento e sensores/atuadores. Essa interconexão possibilita o controle remoto e o acesso a uma ampla gama de serviços, impulsionando assim o contínuo crescimento e evolução da IoT. É essencial ressaltar a importância da padronização das tecnologias nesse cenário em constante desenvolvimento, pois, segundo Da Silva e Junior (2018, p. 3):

> [...] considera-se a IoT como uma rede de dispositivos físicos conectados, que permitem a interação entre si e com objetos externos, através de interfaces de controle e sensoriamento, possibilitando grande quantidade de dados e diversas formas de interação entre o mundo virtual e o real.

3.2 Protocolo MQTT

Protocolo MQTT (*Message Queuing Telemetry Transport*) é um protocolo de transporte de mensagens no formato Cliente/Servidor, que possibilita a comunicação entre máquinas (*Machine to Machine* – M2M) e é amplamente usado para conectividade de IoT (*Internet of Things*). Ele é um protocolo aberto e leve, que pode ser executado com TCP/IP ou em outros protocolos de rede, capturando informações enquanto o Servidor administra o envio e recebimento de dados.

> Ele utiliza o modelo de publicação/subscrição e o TCP (*Transmission Control Protocol*) na camada de transporte, proporcionando uma solução leve e eficiente para o envio de mensagens, otimizando o uso de largura de banda na Internet. Devido a estas características, o MQTT tem sido amplamente utilizado em aplicações IoT (Naik, 2017, tradução nossa).

Em resumo, o protocolo foi desenvolvido para minimizar a largura de banda da rede e atender a outros requisitos restritos, proporcionando confiabilidade e maior eficiência de entrega. Em seu funcionamento, ele foi projetado para transferir mensagens, utilizando um modelo de publicação e inscrição que permite enviar mensagens para um ou vários clientes. Seu funcionamento é similar ao de uma TV, onde uma emissora faz a transmissão de um programa utilizando um canal específico. Em seguida, os espectadores sintonizam nesse canal para visualizar a transmissão.

O protocolo se destaca pela simplicidade e por aperfeiçoar o trânsito de mensagens em redes não confiáveis e de latência baixa. Por usar a arquitetura *Publish/Subscribe*, tem elevada grande escalabilidade nas aplicações, permitindo que vários dispositivos façam publicações e vários assinantes possam consumir esses dados. De acordo com o Cope (2018) e o Banks (2014, tradução nossa), isso é possível devido ao projeto do MQTT, que foi concebido para operar usando tópicos, afim de evitar perdas as informações são subscrevidas, além de não perder as características de um servidor. A outro ponto que reforça sua segurança é abordagem Cliente/Servidor permite que as mensagens sejam redirecionadas do servidor para o cliente. Em outras palavras, um servidor de mensagens MQTT recebe as mensagens dos clientes, as organiza de acordo com seus respectivos tópicos e, em seguida, as distribui aos clientes que se inscreveram em cada tópico, garantindo assim um fluxo eficiente e confiável de comunicação.

3.3 Adafruit

O ecossistema da plataforma Adafruit IO inclui componentes-chave, como o *broker* MQTT para comunicação entre dispositivos IoT e a plataforma. Os *WebSockets* facilitam a transferência de dados em tempo real. A plataforma também fornece APIs para integração com outros serviços e aplicativos externos.

Além do mais, a plataforma apresenta um painel intuitivo que permite aos usuários visualizar e interagir com dados coletados de dispositivos IoT. Isso inclui recursos para monitorar, controlar e configurar dispositivos conectados.

Para simplificar a integração de hardware e plataforma, a Adafruit disponibiliza o protocolo MQTT e bibliotecas específicas compatíveis com diversas plataformas e microcontroladores como Arduino e ESP32, isso facilita a implementação do protocolo MQTT em seus projetos de IoT.

O suporte MQTT do Adafruit IO permite que dispositivos IoT enviem e recebam dados em tempo real. Ele pode ser configurado para publicar informações sobre tópicos específicos ou assinar tópicos para receber atualizações e comandos da plataforma. Essa comunicação bidirecional permite que informações sejam trocadas entre o dispositivo e a plataforma Adafruit IO.

4 METODOLOGIA

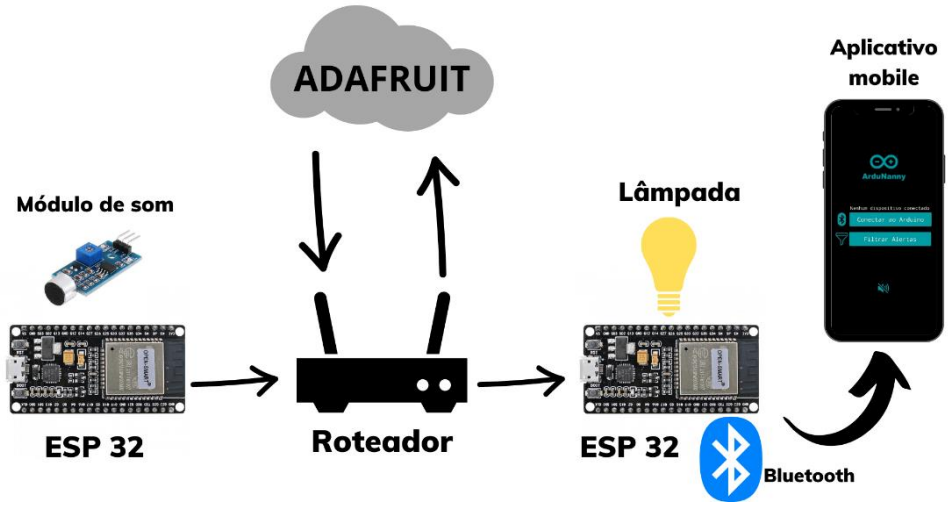
O presente artigo apresenta uma metodologia com o objetivo da criação de um sistema capaz de avisar aos pais deficientes auditivos quando o bebê estiver chorando. A Metodologia é composta por um estudo de caso para a criação de um sistema que cumpra as necessidades dos pais deficientes auditivos, a fim de garantir um mecanismo fácil de ser utilizado, com um custo baixo e de rápido acesso, de forma altamente intuitiva e prática.

Primeiramente, foi-se realizado uma entrevista com pais deficientes auditivos, que passam pelo problema de não saberem quando seu(s) filho(os/as) está chorando, pelo fato de estarem necessitando de ajuda para suas devidas urgências, como por exemplo, fome.

Tendo isso em vista, foi realizado um protótipo que servisse como base para o projeto final – ambos serão apresentados ao decorrer – que consistia em um simples sistema de Arduino Uno em que, ao captar um determinado volume sonoro, pisca três vezes um LED, que estava conectado a uma Protoboard.

Com o protótipo feito, deu-se início ao projeto final, que consiste em um sistema simples, acessível e muito eficaz, em que a forma de comunicação acontece através da rede WiFi, gerenciados por um roteador, a partir de dois microcontroladores ESP32, que em suas funções apresenta a possibilidade de conexão via WiFi e Bluetooth. Para o início da comunicação entre os dois ESP32, o primeiro ESP32 apresenta um módulo de som que capta o som do ambiente, que no caso seria o choro do bebê, e está conectado ao WiFi (roteador), e o outro ESP32, também conectado a rede WiFi, apresenta somente uma lâmpada, ligada através de um módulo Relé, que será acesa quando o outro ESP32 captar um alerta sonoro. O segundo microcontrolador estará conectado através do Bluetooth com um aplicativo mobile que precisará ser instalado no celular dos pais, e será necessário realizar a conexão (escolher) o MAC do EP32, fazendo assim que, sempre que a lâmpada acender, uma notificação seja apresentada na tela do aplicativo, além do celular vibrar por 1,5 segundos, avisando os mesmos que o alerta sonoro foi captado.

Figura 1. Modelo para funcionamento do projeto final

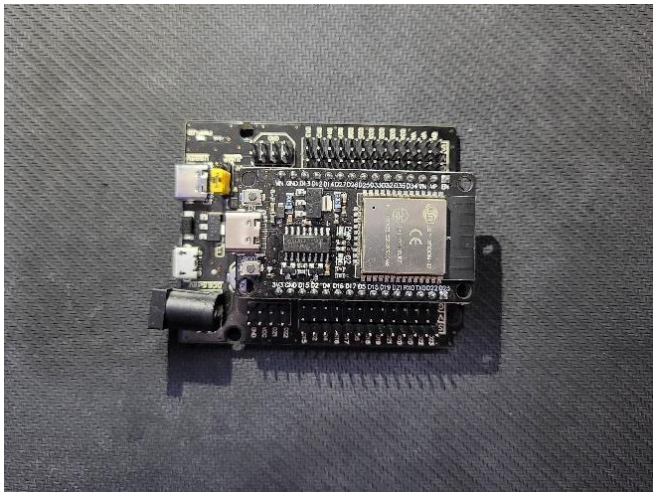


Fonte: Própria

4.1 Recursos

Para a realização do trabalho, foram utilizados os seguintes recursos para a confecção do projeto final:

Figura 2. ESP 32 juntamente do seu módulo de expansão



Fonte: Própria

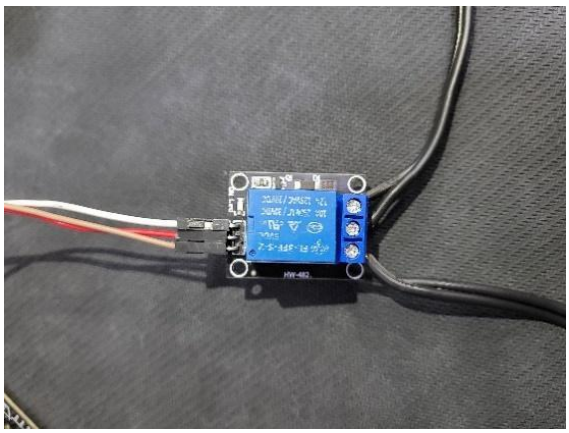
O ESP32 é um microcontrolador de baixo custo e alto desempenho que foi desenvolvido pela Espressif Systems. Geralmente, ele é utilizado em projetos de IoT por causa de sua grande versatilidade e recursos integrados, como o Wi-Fi e o Bluetooth.

Para o projeto em questão, foram utilizados dois ESP32, que serviram como o “cérebro” no projeto, em que, um deles foi programado, juntamente de um

módulo de som para captar o volume do choro do bebê. Além disso, ele envia os dados obtidos para o outro ESP32 por meio da conexão Wi-Fi, ligados por meio de um roteador. Tendo isso em vista, o segundo ESP32, localizado juntamente do Abajur, que também está interligado com uma lâmpada LED, ao receber esses dados do volume obtido por meio do primeiro ESP32, aciona a lâmpada, fazendo-a acender. Servindo assim, como um sinal visual para indiciar o choro de bebê auxiliando os pais. Ademais, o segundo ESP32 está conectado via Bluetooth ao aplicativo mobile desenvolvido pelo grupo, permitindo assim, sempre que a lâmpada fosse acessa, o aplicativo mobile recebesse um alerta por meio dessa conexão via Bluetooth com o celular.

Diante disso, o módulo de expansão do ESP32, visível na Figura 2, não necessariamente se torna obrigatório para a confecção do projeto, mas sim serve para facilitar a sua montagem. Ele capacita o ESP32 de estar conectado com uma fonte de 12V, o que auxilia na hora de ligar o ESP32 na energia. Além de possibilitar acrescentar futuros recursos extras, como outros módulos etc.

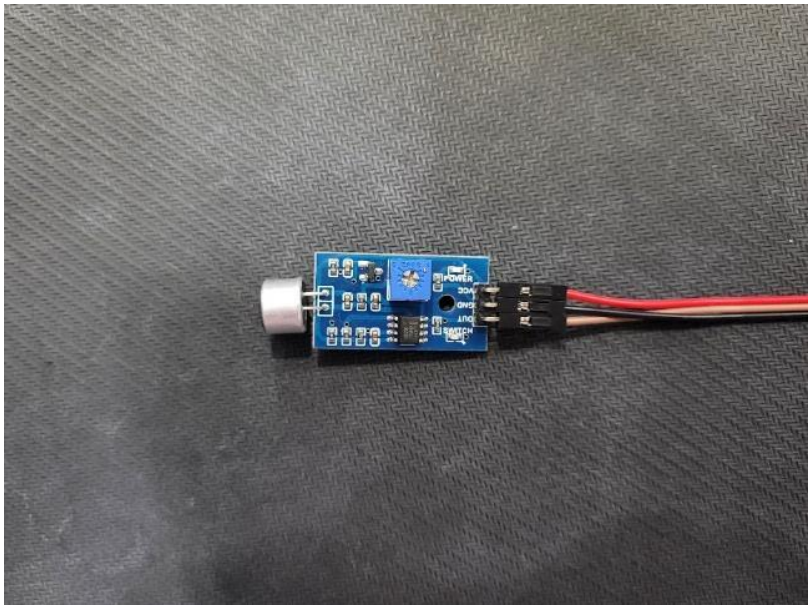
Figura 3. Módulo Relé



Fonte: Própria

O módulo Relé é um dispositivo eletromecânico que serve para controlar o fluxo de corrente elétrica em um circuito. Ele funciona como um interruptor, permitindo assim, que, um sinal de 220V que vem da rede elétrica de uma casa interaja com ele, logo, proporcionando o acionamento ou a inativação da lâmpada, oferecendo proteção e segurança, e isolando eletricamente o circuito de baixa tensão do circuito de alta tensão.

Figura 4. Módulo de som/ruído



Fonte: Própria

O módulo de som/ruído, casualmente conhecido como módulo de detecção de som, é um componente eletrônico muito utilizado para captar e converter as ondas sonoras em um sinal elétrico. No projeto em questão, ele foi usado para captar o volume do som do choro do bebê, que é processado pelo primeiro ESP32 e enviado para o segundo ESP32, assim ligando a lâmpada por meio do volume obtido naquele ambiente.

Figura 5. Recipiente de plástico para armazenar o ESP32 com o módulo de som

Fonte: Própria

Como é possível de se observar na imagem, o primeiro ESP32, no qual vai estar no quarto do bebê e poderá, ou não, captar o som por meio do módulo de som, foi armazenado em um recipiente de plástico para assim garantir a melhor

segurança do componente, visto que é algo frágil, podendo facilmente ser danificado. Além de que, como visto na figura, o módulo está acoplado à tampa do recipiente para que ele possa captar o som.

Figura 6. Abajur utilizado para armazenar o outro ESP32

Fonte: Própria

O abajur, que está com o ESP32, juntamente da lâmpada LED, ambos conectados ao relé que está com sua configuração como normalmente aberto, assim quando é detectado um alerta sonoro, o relé fecha e abre rapidamente três vezes, assim fazendo a lâmpada piscar.

Figura 7. Fonte 12V Bivolt

Fonte: Própria

A Fonte 12V é um componente que converte a corrente elétrica de uma tomada de alimentação padrão em uma voltagem de 12 volts. Para o projeto, foram utilizadas duas fontes, uma para ligar o ESP32 Transmissor, e a outra para ligar o ESP32 receptor, ambos juntos do abajur, visto na figura anterior.

Figura 8. Lâmpada LED

Fonte: Força&Luz

A lâmpada é um componente que fica conectada à rede elétrica, e quando acionada, emite luz para iluminar o ambiente. No projeto mencionado, a lâmpada foi utilizada para alertar os pais que o choro do bebê foi notificado, usada em conjunto com o módulo relé controlado pelo ESP32 para criar um sistema de alerta visual, chamando assim, a atenção dos cuidadores para a atender as necessidades da criança.

4.2 Tabela de preços

Visando garantir um sistema de monitoramento acessível e econômico, este projeto apresenta o custo tanto do primeiro protótipo que serve de referência para o desenvolvimento do projeto final, quanto do próprio projeto final. Ao considerar esta proposta, é importante observar que os pais deficientes auditivos arcarão apenas com os custos associados à compra dos componentes necessários para montar o sistema final, e não com o custo do protótipo. Desta forma, é apresentada uma análise detalhada dos custos envolvidos com o objetivo de fornecer uma visão clara e transparente do investimento necessário.

Tabela 1. Despesas e investimentos do Protótipo.

Despesas e investimentos do Protótipo				
Item	**Quantidade**	**Preço**	**Preço Total**	**Fonte do recurso**
LED	1	R$0,20	R$0,20	Doação
Módulo sensor de som (protótipo e projeto final)	1	R$9,45	R$9,45	Compra do Grupo
Resistor de 33Ω	1	R$0,05	R$0,05	Doação
Arduino UNO	1	R$ 59,76	R$ 59,76	Doação
Protoboard	1	R$ 12,90	R$ 12,90	Doação
Jumpers macho/macho	4	R$0,25	R$1,01	Doação
TOTAL: R$ 83,37				

Fonte: Própria

Tabela 2. Despesas e investimentos do Projeto Final

Despesas e investimentos do Projeto Final				
Item	**Quantidade**	**Preço**	**Preço Total**	**Fonte do recurso**
ESP32	2	R$34,20	R$68,40	Compra do Grupo
Módulo expansor para EPS32 - 30 pinos	2	R$33,40	R$66,80	Compra do Grupo
Jumpers fêmea/fêmea	6	R$0,30	R$1,70	Doação
Módulo Relé 1 canal	1	R$ 7,83	R$ 7,83	Doação
Lâmpada LED	1	R$ 12,90	R$ 12,90	Doação
Abajur	1	R$ 35,00	R$ 35,00	Compra do Grupo
Fontes 12v	2	R$ 24,00	R$ 48,00	Compra do Grupo
Conector p4 macho	1	R$0,67	R$0,67	Compra do Grupo
Recipiente plástico	1	R$8,00	R$8,00	Compra do Grupo
TOTAL: R$ 249,30				

Fonte: Própria

5 DESENVOLVIMENTO DO PROTÓTIPO

Nessa seção, serão abordados inicialmente os recursos utilizados para a realização do protótipo que serviu como modelo inicial para o desenvolvimento do projeto final, além do código utilizado para que ele funcione e, por fim, o projeto físico para demonstração.

5.1 Projeto do Circuito

Para a realização do protótipo, os recursos utilizados foram:

- 1 LED da cor vermelha
- 1 módulo sensor de som
- 1 resistor de 33 Ω
- 1 Arduino UNO
- 1 Protoboard
- 4 jumpers macho/macho

Explicando como o protótipo funciona, basicamente, quando o módulo de som capta uma certa frequência, ele enviará um sinal para o LED, que será ligado e desligado 3 vezes, piscando.

5.2 Código

A parte lógica do protótipo foi realizado por meio da ferramenta Arduino IDE, que serve como uma plataforma para desenvolvimento de programas em placas de Arduino. O código que foi utilizado para o desenvolvimento do protótipo inicial, que serve como um modelo para a realização do projeto final foi o seguinte:

Figura 9. Código para o desenvolvimento do protótipo

```
int pinoLed = 12; //PINO DIGITAL UTILIZADO PELO LED
int pinoSensor = A0; //PINO DIGITAL UTILIZADO PELO SENSOR
int estadoSensor; //VARIÁVEL QUE ARMAZENA O ESTADO DA SAÍDA DO SENSOR (HIGH/LOW)
int leitura; //VARIÁVEL QUE FAZ A LEITURA DO PINO ANALÓGICO

void setup(){
  pinMode(pinoSensor, INPUT); //DEFINE O PINO COMO ENTRADA
  pinMode(pinoLed, OUTPUT); //DEFINE O PINO COMO SAÍDA
}

void loop(){
  leitura = analogRead(pinoSensor); //FAZ A LEITURA DO VOLUME SONORO
  if (leitura > 800) {              //CASO A LEITURA SEJA MAIOR QUE 800, LIGA E
    digitalWrite(pinoLed,HIGH);     //DESLIGA O LED 3 VEZES
    delay(500);
    digitalWrite(pinoLed,LOW);
    delay(500);
    digitalWrite(pinoLed,HIGH);
    delay(500);
    digitalWrite(pinoLed,LOW);
    delay(500);
    digitalWrite(pinoLed,HIGH);
    delay(500);
    digitalWrite(pinoLed,LOW);
    delay(500);
  }
}
```

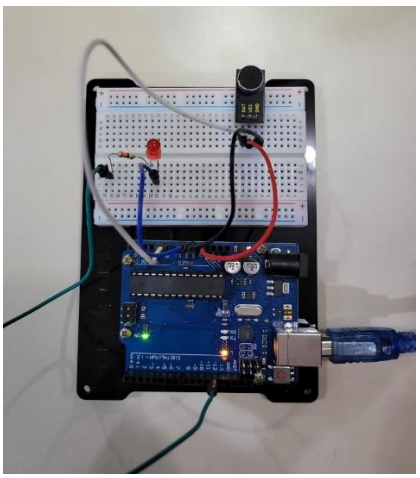
Fonte: Própria

5.3 Montagem do modelo

Para a realização da montagem, foi posto na protoboard, todos os componentes necessários para a confecção do protótipo. Pode ser destacado os o objetivo dos Jumpers, que vistos nas imagens, ambos o GND (negativo) tanto do módulo de som, quanto do LED, foram ligados diretamente no Arduino, além de que o pino digital utilizado para o LED está ligado na porta 12, e o pino digital utilizado pelo módulo está ligado na porta A0.

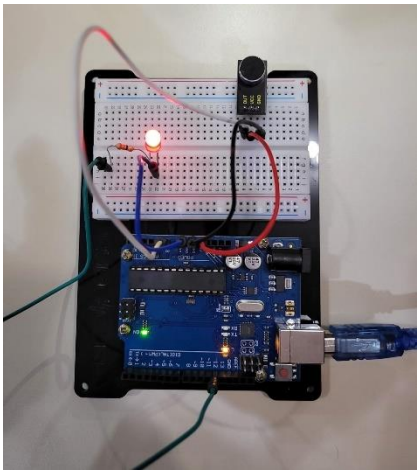
Por fim, conectado o Arduino UNO por meio de um cabo USB AB na entrada de um computador, assim, a placa foi energizada, ligando-o.

Figura 10. Protótipo LED DESLIGADO



Fonte: Própria

Figura 11. Protótipo LED LIGADO



Fonte: Própria

6 DESENVOLVIMENTO DO APLICATIVO MOBILE

Nessa seção, será abordado o desenvolvimento do aplicativo mobile, que desempenha um grande papel no projeto, pois é por meio do aplicativo, que os pais terão noção que o bebê está passando por necessidades, e serão avisados quando não estiverem perto da lâmpada, mas sim, por notificação no aplicativo desenvolvido.

6.1 Descrição do aplicativo

O aplicativo ArduNanny, desenvolvido para atender pais com deficiência auditiva, oferece uma solução prática para monitorar e alertar sobre o comportamento da criança. Ele permite um acesso e controle fácil dos registros e alertas, utilizando sensores para captar o comportamento da criança. Quando o aplicativo detecta que a criança está chorando, ele notifica os pais por meio de vibrações no celular.

6.2 Funcionalidades do aplicativo

Os aplicativos oferecem uma variedade de funcionalidades que simplificam o uso e atendem às principais necessidades dos usuários. O aplicativo possui uma guia de acesso rápido que permite ao usuário conectar-se ao ESP32. Uma vez conectado, a segunda ferramenta entra em ação, exibindo os registros de alertas sonoros. Esses registros podem ser filtrados com base na data e hora definidas pelo usuário. Enquanto estiver conectado ao ESP32, o aplicativo desempenha sua função principal, que é emitir um alerta sempre que a criança chorar. O aplicativo provoca uma resposta em que o celular vibra e exibe o ícone de alerta durante o alarme.

6.3 Código do aplicativo

Para o desenvolvimento do aplicativo, decidiu-se utilizar a plataforma “Kodular”, também conhecida como programação visual em blocos, para desenvolver o aplicativo. Essa abordagem permite uma manipulação mais fácil do código, uma vez que se baseia em blocos gráficos interconectados, em vez de escrever linhas de código tradicionais. Cada bloco representa uma instrução ou parte do código, e eles são projetados para se encaixarem logicamente uns nos outros, formando uma sequência de ações que define o comportamento do programa.

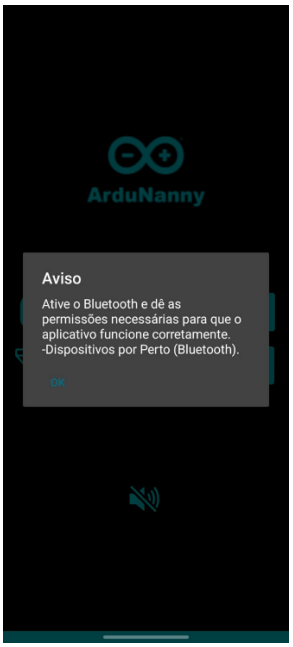
Os blocos contêm informações sobre ações específicas, como loops, condicionais, entrada de dados e saída de resultados. Ou seja, para construir o fluxo

de execução do programa, arrastam-se e conectam-se os blocos e assim os programas são criados, combinando diferentes blocos para definir a sequência de ações a serem executadas.

Após concluir a lógica de construção, basta instalar o APK gerado pela plataforma Kodular e executá-lo normalmente.

Diante disso, será primeiramente explicado a interface de usuário do aplicativo desenvolvido e como poderá ser utilizado, e após isso, o código fonte para o desenvolvimento do aplicativo.

Figura 12. Requisito ativação do Bluetooth



Fonte: Própria

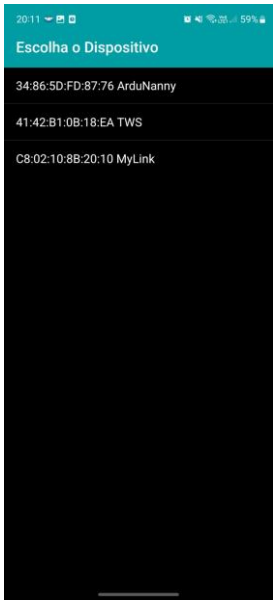
Ao abrir o aplicativo, o usuário é recebido com uma mensagem que pede que a permissão de Bluetooth seja concedida.

Figura 13. Tela de inicio

Fonte: Própria

Após a notificação a tela principal é exibida, mostrando que nenhum dispositivo está conectado.

Figura 14. Conexões Bluetooth disponíveis



Fonte: Própria

Ao clicar no botão “Conectar ao Arduino” uma listagem de dispositivos Bluetooth pareados é aberta.

Figura 15. Confirmação de conexão do Bluetooth

Fonte: Própria

Após realizar a conexão, é apresentada a mensagem “Dispositivo Conectado!”.

Figura 16. Mensagem de alerta

Fonte: Própria

Ao receber um novo alerta sonoro, o celular vibra e a mensagem “Alerta Sonoro!” é apresentada.

Figura 17. Área de filtro dos alertas

Fonte: Própria

Ao clicar no botão filtrar alertas a tela de filtragem é exibida. Nela é possível filtrar todos os alertas recebidos.

Figura 18. Botão de acesso rápido (filtro)

Fonte: Própria

Ao clicar no botão “Dia de Hoje” o dia, mês e ano é preenchido automaticamente para facilitar o uso do aplicativo pelo usuário.

Figura 19. Campo do horário

Fonte: Própria

Em seguida é necessário completar com a hora desejada.

Figura 20. Exibição dos alertas por filtragem

Fonte: Própria

Quando o botão “Filtrar” é clicado, a quantidade de alertas é exibida.

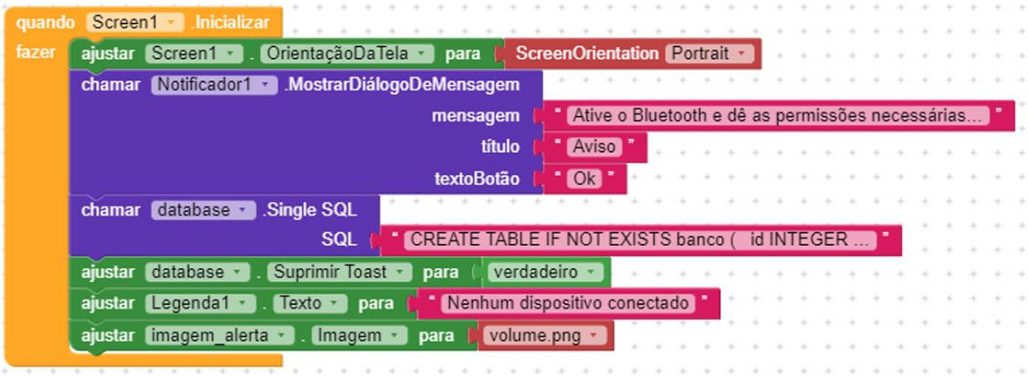
Figura 21. Botão para limpar a filtragem

Fonte: Própria

Concluindo a interface do usuário, ao clicar no botão “Limpar”, todos os dados da tela voltam ao padrão.

Agora, descrito a interface de usuário, será detalhado o código-fonte do aplicativo, feito por meio de blocos.

Figura 22. Tela de início, requisição de permissão (código)



Fonte: Própria

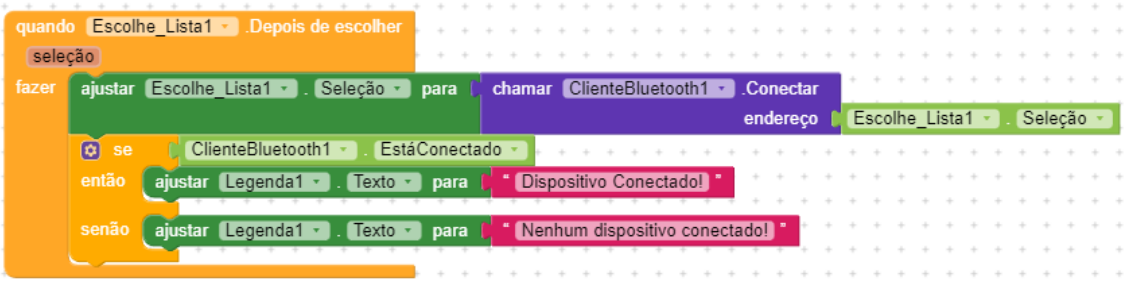
Quando o aplicativo é iniciado, a orientação da tela é ajustada para Vertical, uma mensagem é exibida pedindo para liberar as permissões necessárias para que o aplicativo funcione. O banco de dados é criado caso não exista e é definido o texto “Nenhum dispositivo conectado” além de uma imagem na parte inferior da tela.

Figura 23. Requisito ativação do Bluetooth (código)

Fonte: Própria

Já nessa parte do código, exibe na tela os dispositivos Bluetooth disponíveis para conexão.

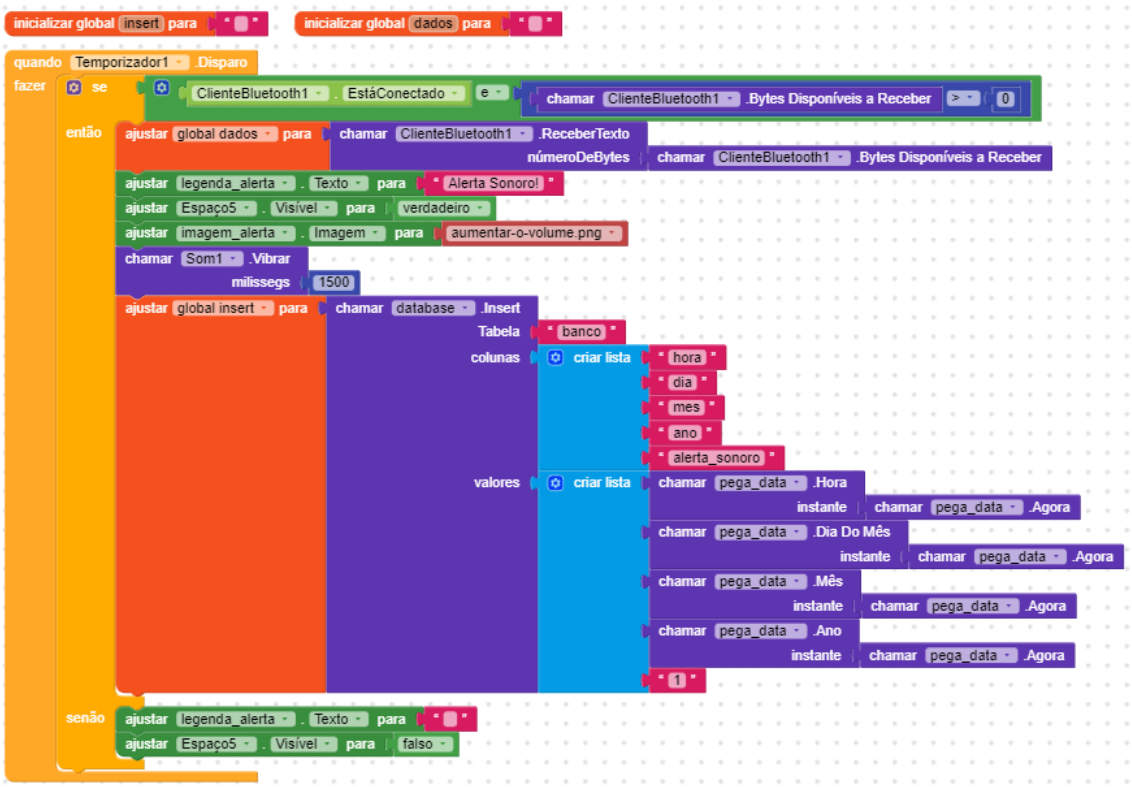
Figura 24. Confirmação ou não da conexão com o Bluetooth (código)



Fonte: Própria

Após a escolha do dispositivo, caso a conexão seja bem-sucedida, é exibido “Dispositivo Conectado!” caso contrário, é mostrado “Nenhum dispositivo conectado!”.

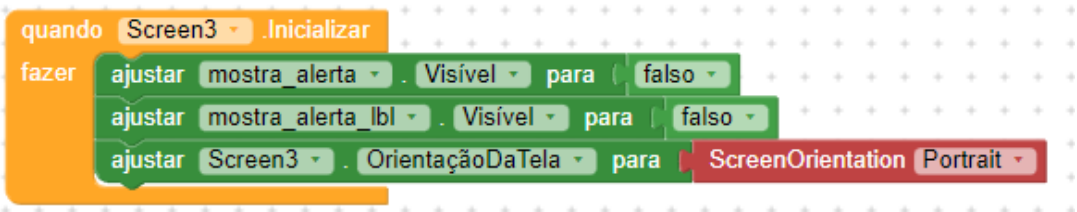
Figura 25. Aparição da mensagem de alerta sonoro (código)



Fonte: Própria

Foi declarado duas variáveis globais, "*insert*" e "dados" com valores vazios. A cada 500 milissegundos o temporizador é disparado, caso haja conexão com ESP32 e ele esteja enviando dados ao aplicativo, a variável global "dados" é ajustada para receber os valores do ESP32, assim é exibido a mensagem "Alerta Sonoro!", uma nova imagem, o celular vibra por 1,5 segundos, além de fazer uma inserção no banco de dados com o a hora, dia, mês e ano do alerta recebido. Caso nenhum dado seja recebido, o layout do aplicativo continua o mesmo.

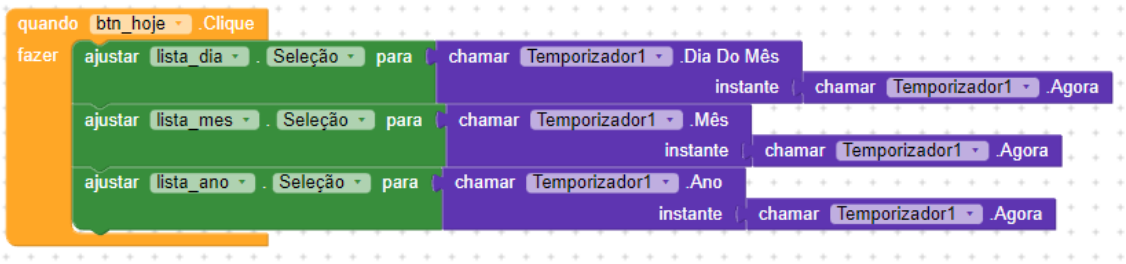
Figura 26. Aba filtragem, orientação vertical



Fonte: Própria

Quando a tela de filtragem é aberta é definido sua orientação como vertical.

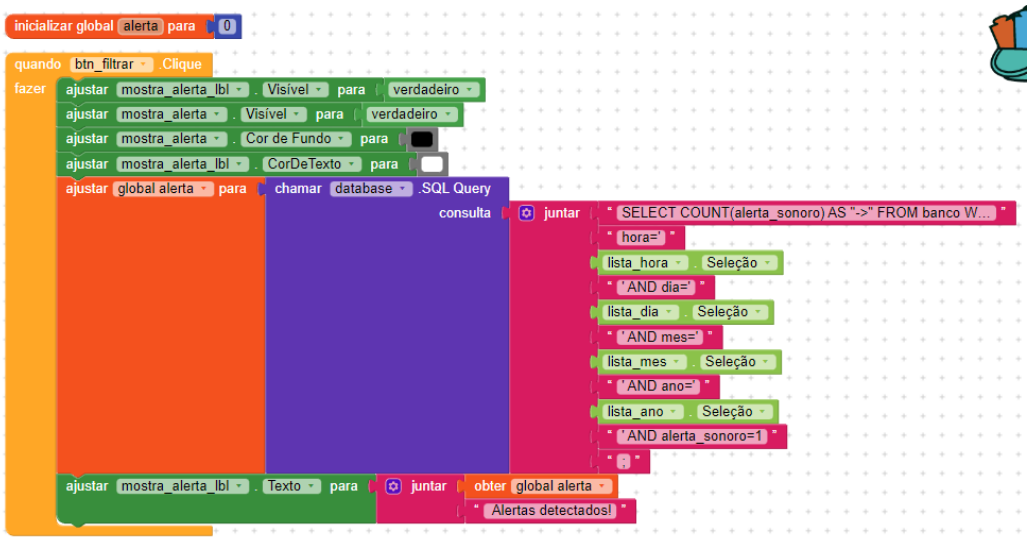
Figura 27. Botão de acesso rápido do filtro (código)



Fonte: Própria

Para facilitar o uso do aplicativo foi pensado em desenvolver um botão que completasse os campos dia, mês e ano automaticamente, com os valores da data atual no momento da utilização do aplicativo.

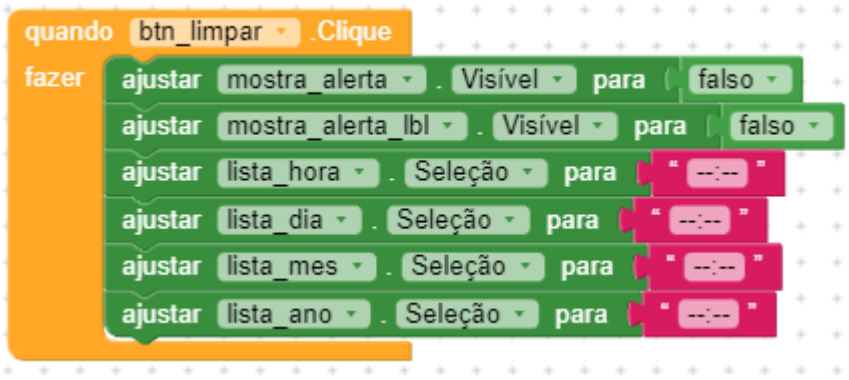
Figura 28. Variável para exibição dos alertas por filtragem (código)



Fonte: Própria

Foi criada uma variável global “alerta” com início 0. Ao clicar no botão de filtragem espera-se que o usuário já tenha selecionado a hora, dia, mês e ano do alerta. Com isso é realizada uma filtragem de quantos alertas foram encontrados em determinada data, assim sendo exibido os valores na tela.

Figura 29. Botão para limpar a filtragem (código)



Fonte: Própria

E por fim, ao clicar no botão de limpeza de seleção, todos os dados selecionados são limpos para realizar uma nova pesquisa.

7 DESENVOLVIMENTO DO PROJETO FINAL

Nessa seção, serão abordados inicialmente os recursos utilizados para a realização do projeto final, além do código utilizado para que ele funcione e, por fim, o projeto físico para demonstração.

7.1 Projeto do Circuito

Para a realização do projeto, foi utilizado os seguintes componentes:

- 2 ESP32
- 2 Módulo expansor para ESP32 30 pinos
- 1 Módulo sensor de som
- 6 Jumpers Fêmea/Fêmea
- 1 Módulo relé 1 canal
- 1 Lâmpada de led
- 1 Abajur
- 2 Fontes 12V
- 1 Conector p4 macho
- 1 Recipiente Plástico

7.2 Código

O código utilizado em ambos os ESP32 foram criados tendo como base os exemplos deixados pela biblioteca do Adafruit, utilizando a plataforma Arduino IDE para o desenvolvimento do código.

Figura 30. Arquivo de Configuração ESP32 Transmissor

TRANSMISSOR_DEFINITIVO.ino | config.h

```
/***************** Configuração io.adafruit.com *********************/
//Usuário e chave do io.adafruit.com
#define IO_USERNAME "mend0ca"
#define IO_KEY "aio_NVpV18YWiGYq4xhqcIqIJzJ2xKqj"

/****************************** WIFI ***********************************/
//Nome e senha da rede WiFI
#define WIFI_SSID "Grupo2"
#define WIFI_PASS "12345678"

//Inclusão da biblioteca do Adafruit io, configuração da biblioteca
#include "AdafruitIO_WiFi.h"

#if defined(USE_AIRLIFT) || defined(ADAFRUIT_METRO_M4_AIRLIFT_LITE) ||         \
    defined(ADAFRUIT_PYPORTAL)
//Configura os pinos usados para a conexão do ESP32
#if !defined(SPIWIFI_SS)
#define SPIWIFI SPI
#define SPIWIFI_SS 10 //Pino de seleção de chip
#define NINA_ACK 9    //a.k.a pino BUSY ou READY
#define NINA_RESETN 6 //Pino de Reset
#define NINA_GPIO0 -1 //Não conectado
#endif
AdafruitIO_WiFi io(IO_USERNAME, IO_KEY, WIFI_SSID, WIFI_PASS, SPIWIFI_SS,
                   NINA_ACK, NINA_RESETN, NINA_GPIO0, &SPIWIFI);
#else
AdafruitIO_WiFi io(IO_USERNAME, IO_KEY, WIFI_SSID, WIFI_PASS);
#endif
```

Fonte: Própria

Figura 31. Código ESP32 Transmissor - Parte 1

TRANSMISSOR_DEFINITIVO.ino | config.h

```
/************************ Configuração *********************************/
//Arquivo de configurações
#include "config.h"

/************************ Código inicia aqui ***************************/

//Contador que inicia zerado
int count = 0;

//Pino analógico do microfone
const int microphonePin = 32;

//Conecta no Feed ArduNanny
AdafruitIO_Feed *counter = io.feed("ArduNanny");

void setup() {

  //Inicia conexão serial
  Serial.begin(115200);

  //Escreve no monitor serial
  Serial.print("Connecting to Adafruit IO");

  //Conecta ao Adafruit
  io.connect();

  //Aguarda a conexão
  while(io.status() < AIO_CONNECTED) {
```

Fonte: Própria

Figura 32. Código ESP32 Transmissor - Parte 2

TRANSMISSOR_DEFINITIVO.ino | config.h

```
27    //Aguarda a conexao
28    while(io.status() < AIO_CONNECTED) {
29      Serial.print(".");
30      delay(500);
31    }
32
33    //Quando conectado exibe o status
34    Serial.println();
35    Serial.println(io.statusText());
36  }
37
38  void loop() {
39
40    //Mantém o cliente conectado ao io.adafruit.com e
41    //processa todos os dados recebidos
42    io.run();
43
44    //Lê o estado do microfone
45    int microphoneValue = analogRead(microphonePin);
46
47    //Caso o estado do microfone seja igual a 4095
48    //é enviado um sinal para o feed no io.adafruit.com
49    if (microphoneValue == 4095){
50      Serial.print("Alerta Sonoro Detectado -> ");
51      Serial.println(count);
52      counter->save(count);
```

Fonte: Própria

Figura 33. Código ESP32 Transmissor - Parte 3

```
47    //Caso o estado do microfone seja igual a 4095
48    //é enviado um sinal para o feed no io.adafruit.com
49    if (microphoneValue == 4095){
50      Serial.print("Alerta Sonoro Detectado -> ");
51      Serial.println(count);
52      counter->save(count);
53
54      //Adiciona + 1 ao contador
55      count++;
56      delay(2000);
57    }
58  }
59
```

Fonte: Própria

Figura 34. Arquivo de Configuração ESP32 Receptor

RECEPTOR_DEFINITIVO.ino | config.h

```
/****************** Configuração do Adafruit ****************/
//Usuário e chave do Adafruit
#define IO_USERNAME "mend0ca"
#define IO_KEY "aio_NVpV18YWiGYq4xhqcIqIJzJ2xKqj"

/****************************** WIFI ***********************************/

//Define o Nome e Senha do WIFI
#define WIFI_SSID "Grupo2"
#define WIFI_PASS "12345678"

//Inclui a biblioteca do Adafruit configurações da biblioteca
#include "AdafruitIO_WiFi.h"

#if defined(USE_AIRLIFT) || defined(ADAFRUIT_METRO_M4_AIRLIFT_LITE) ||         \
    defined(ADAFRUIT_PYPORTAL)
//Configura os pinos usados para a conexão do ESP32
#if !defined(SPIWIFI_SS)

#define SPIWIFI SPI
#define SPIWIFI_SS 10 //Pino de seleção de chip
#define NINA_ACK 9    //a.k.a pino BUSY ou READY
#define NINA_RESETN 6 //Pino de Reset
#define NINA_GPIO0 -1 //Não conectado
#endif
AdafruitIO_WiFi io(IO_USERNAME, IO_KEY, WIFI_SSID, WIFI_PASS, SPIWIFI_SS,
                   NINA_ACK, NINA_RESETN, NINA_GPIO0, &SPIWIFI);
#else
AdafruitIO_WiFi io(IO_USERNAME, IO_KEY, WIFI_SSID, WIFI_PASS);
#endif
```

Fonte: Própria

Figura 35. Código ESP32 Receptor- Parte 1

RECEPTOR_DEFINITIVO.ino | config.h

```
/************************ Configuração *********************************/

//Inclusão do arquivo de configuração e biblioteca Bluetooth
#include "config.h"
#include <BluetoothSerial.h>

/************************ Código inicia aqui ***************************/

//Nome do usuário do Adafruit
#define FEED_OWNER "mend0ca"

//Pino da lâmpada
const int lpa = 5;

//Último valor
int lastValue = 0;

//Variavel para chamar o bluetooth
BluetoothSerial SerialBT;

//Configura o Feed
AdafruitIO_Feed *sharedFeed = io.feed("ArduNanny", FEED_OWNER);

void setup() {

  //Inicia conexão serial
  Serial.begin(115200);

  SerialBT.begin("ESP32"); // Define o nome do dispositivo Bluetooth

```

Fonte: Própria

Figura 36. Código ESP32 Receptor- Parte 2

RECEPTOR_DEFINITIVO.ino | config.h

```
30
31    //Define a Lâmpada como saída
32    pinMode(lpa, OUTPUT);
33
34    //Conecta com o Adafruit io
35    Serial.print("Connecting to Adafruit IO");
36    io.connect();
37
38    //A função handleMessage sempre será chamada quando
39    //uma nova mensagem for recebida no Feed
40    sharedFeed->onMessage(handleMessage);
41
42    //Aguarda conexão com Adafruit
43    while(io.status() < AIO_CONNECTED) {
44      Serial.print(".");
45      delay(500);
46    }
47
48    //Quando conectado exibe o status
49    Serial.println();
50    Serial.println(io.statusText());
51    sharedFeed->get();
52
53  }
54
55  void loop() {
56    //Mantém o cliente conectado ao io.adafruit.com e
57    //processa todos os dados recebidos
58    io.run();
59
60  }
```

Fonte: Própria

Figura 37. Código ESP32 Receptor- Parte 3

RECEPTOR_DEFINITIVO.ino | config.h

```
59
60  }
61
62  //Função que verifica se uma nova mensagem foi recebida
63  //no Feed
64  void handleMessage(AdafruitIO_Data *data) {
65    //Variavel newValeue recebida em data e convertida
66    //para inteiro
67    int newValue = data->toInt();
68
69    //Caso o novo valor (newValue) seja diferente
70    //do valor antigo (lastValue) a lâmpada pisca por
71    //3 vezes e envia uma mensagem ao aplicativo via
72    //bluetooth
73    if (newValue != lastValue) {
74      digitalWrite(lpa, HIGH);
75      delay(400);
76      digitalWrite(lpa, LOW);
77      delay(400);
78      digitalWrite(lpa, HIGH);
79      delay(400);
80      digitalWrite(lpa, LOW);
81      delay(400);
82      digitalWrite(lpa, HIGH);
83      delay(400);
84      digitalWrite(lpa, LOW);
85
86      //Define a variavel message como ".",
87      String message = ".";
```

Fonte: Própria

Figura 38. Código ESP32 Receptor- Parte 4

```
86      //Define a variavel message como ".",
87      String message = ".";
88
89      //Envia a Variavel message para o aplicativo
90      //via bluetooth
91      if (SerialBT.connected()) {
92        SerialBT.println(message);
93
94      //Define o valor antigo (lastValue) com o valor
95      //do novo valor (newValue)
96      lastValue = newValue;
97      }
98    }
99  }
100
```

Fonte: Própria

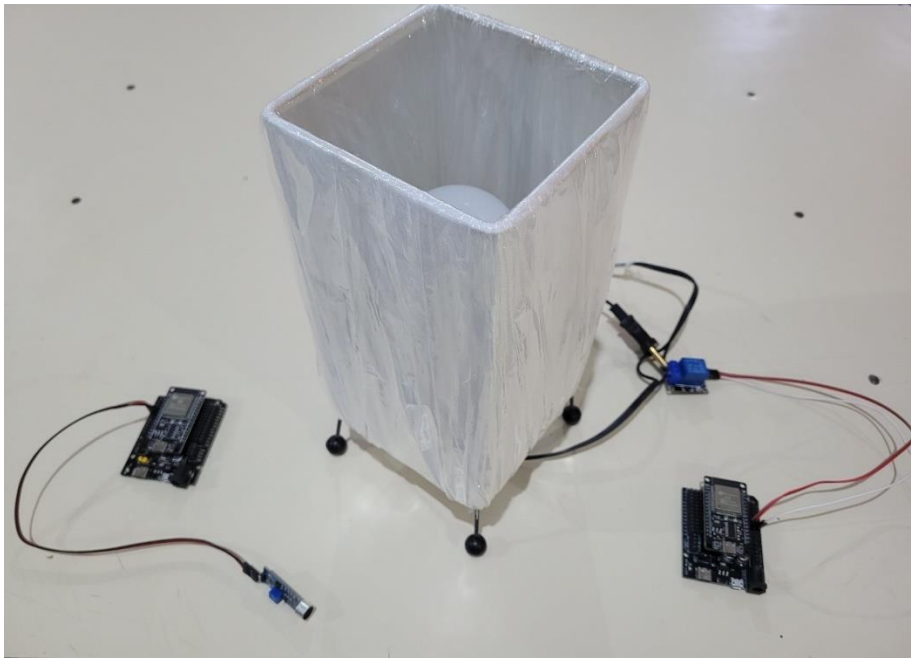
7.3 Montagem do produto final

Para realizar a montagem do projeto, foi considerado que seria necessária uma estrutura para suportar o ESP32 receptor de sinal. Como o projeto envolve uma lâmpada, optou-se por utilizar um abajur como suporte. Para o ESP32 transmissor de sinal, foi utilizado um recipiente de plástico simples, facilitando a personalização estética.

Durante a montagem do ESP32 transmissor, foram conectados três jumpers Fêmea/Fêmea ao módulo sensor de som. Um dos jumpers foi conectado à alimentação VCC 3.3V, outro ao negativo GND e o terceiro foi conectado ao pino digital/analógico D32.

Na montagem do ESP32 receptor, também foram utilizados três jumpers Fêmea/Fêmea, que foram conectados ao módulo relé. Um dos jumpers foi conectado à alimentação VCC 5V, outro ao negativo GND e o terceiro foi conectado ao pino digital D5.

Figura 39. Projeto final físico



Fonte: Própria

8 RESULTADOS E DISCUSSÃO

Como resultado, o projeto proporcionou a comunicação entre dois ESP32, sendo um o transmissor no qual por meio de um módulo que capta som e manda para o broker Adafruit, através de um roteador (conectando os dois ESP32), assim tornando-se visível para o outro ESP32 que por sua vez, recebe o sinal e liga a lâmpada conectada, assim comunicando ao cliente. Resumindo, o Arduino que recebe as informações da página web, simultaneamente com a lâmpada, transmite uma mensagem por meio do Bluetooth para o celular do cliente que está conectado, e assim recebendo uma notificação no aplicativo, além do mesmo fazer com que o celular vibre.

Partindo agora para a discussão sobre todo o projeto, o objetivo principal era suprir a necessidade dos pais que não tinham como saber quando, por exemplo, seu bebê estava chorando, em vista disso, o projeto tem um preço abaixo do que as opções atuais encontradas no mercado, que são inviáveis para a maioria dos casais.

Porém, inicialmente, o projeto seria realizado com 2 Arduinos, junto de 2 módulos ESP8266 e em um deles, um módulo Bluetooth. No entanto, após a realização de alguns testes, os resultados obtidos foram insatisfatórios, tendo em mente que os módulos de comunicação sem fio não conseguiam se comunicar. Esse problema se deu por causa da falta de compatibilidade dos módulos com o Arduino utilizado, e como solução para isso, a placa de automação foi substituída pelo ESP32, e como consequência, o preço também diminuiu, visto que o Arduino é mais custoso que o ESP32. Além do exposto, a compra dos módulos ESP8266 e módulo Bluetooth não seriam mais necessárias, visto que o microcontrolador ESP32 já apresenta em sua composição ambas as funções, tanto de Wi-Fi, quanto de Bluetooth.

8.1 Instruções técnicas

Neste tópico, será abordado as instruções técnicas que serão necessárias para a utilização do projeto ao todo de forma eficiente, que deverá ser efetuado pelos responsáveis do bebê.

Inicialmente, é necessário realizar alguns pré-requisitos para utilizar o ArduNanny. Isso inclui conectar o dispositivo à tomada e instalar o aplicativo no celular do usuário. Além disso, o hardware com o microfone deve ser posicionado a uma

distância adequada da criança, enquanto o aplicativo é disponibilizado pelos desenvolvedores. Após a conclusão dessa etapa inicial, o usuário pode abrir o aplicativo, onde encontrará uma aba específica para a conexão via Bluetooth. Basta garantir que o Bluetooth do celular esteja ativado para estabelecer a conexão. No que diz respeito à conexão do hardware com a rede Wi-Fi, é necessário que um técnico configure as informações de acesso, como nome de usuário e senha do roteador, para o usuário.

Finalmente, o usuário pode utilizar o aplicativo normalmente e receber alertas, que podem ser filtrados por horário. É importante certificar-se de que o telefone não esteja no modo silencioso e de que o ArduNanny tenha acesso à internet através do sinal Wi-Fi.

9 CONSIDERAÇÕES FINAIS

A partir da realização deste projeto, os objetivos principais foram atingidos, os quais consistem no atendimento das necessidades dos casais deficientes auditivos, e ao mesmo tempo, um custo-benefício enorme, ou seja, o resultado do projeto foi extremamente satisfatório. Como forma de estudo, para o grupo, teve uma enorme relevância para o aprendizado, pois possibilitou adquirir experiência com Arduino, ESP32, módulos para a utilização, além de criação de um projeto mais viável para os pais deficientes auditivos, contribuindo assim, não somente para a conclusão do curso como um todo, mas também como uma forma de ajudar um grupo que por muitas vezes é tratada como invisível aos olhos do sistema capitalista, que visa atingir, na maioria das vezes, a maior parcela das pessoas.

9.1 Implementações futuras

No contexto deste trabalho, é importante considerar implementações futuras que não puderam ser realizadas devido a restrições de tempo. Uma delas é a implementação da execução do aplicativo móvel em segundo plano. Isso permitiria que os pais recebam notificações mesmo quando o aplicativo não está sendo executado em primeiro plano. Outra área de melhoria é seria a implementação do flash móvel como uma notificação visual adicional quando o aplicativo recebesse um sinal por Bluetooth. Essas implementações seriam interessantes, pois melhorariam a usabilidade do sistema e permitiria que pais surdos reconhecessem rapidamente a necessidade de atenção de seu bebê. Ao considerar essas melhorias, este estudo contribui para o avanço do conhecimento na área de tecnologia assistiva e incentiva futuros pesquisadores a explorar e aprimorar as soluções propostas.

REFERÊNCIAS

ADAFRUIT. **Adafruit IO.** Disponível em: https://io.adafruit.com. Acesso em: 10 jun. 2023.

BANKS, A. **MQTT Version 3.1. 1.** Edited by Andrew Banks and Rahul Gupta. OASIS Standard, 2014. Tradução: Própria. Disponível em: http://docs.oasis-open.org/mqtt/mqtt/v3.1.1/os/mqtt-v3.1.1-os.pdf. Acesso em 14 jun.2023.

CRAVO, Edilson. **Como o Protocolo MQTT funciona e quais são as suas vantagens?** Disponível em: https://blog.kalatec.com.br/protocolo-mqtt/#:~:text=O%20que%20%C3%A9%20o%20protocolo,IoT%20(Internet%20of%20Things). Acesso em 14 jun.2023.

DA SILVA, Sani de Carvalho Rutz; JUNIOR, Albino Szesz. **Internet das coisas na educação:** uma visão geral. Revista ENCITEC, v. 8, n. 2, p. 57-69, 2018.

ERLANDSON, Robert F. **Universal and Accessible Design for Products, Services, and Processes**. Tradução: Própria. Boca Raton, FL, USA: CRC, 2008.

FRANÇA, A.C.C.V de. **Interação Social de Pessoas Surdas no Cotidiano, Mediada por Sistemas de Produtos e Serviços de Comunicação**. 2011. Dissertação (Mestrado em (Tecnologia) - Programa de Pós-Graduação em Tecnologia, UTFPR, Curitiba.

IBGE. **Censo demográfico 2010:** características gerais da população, religião e pessoas com deficiência. Ministério do Planejamento, Orçamento e Gestão. Rio de Janeiro: Instituto Brasileiro de Geografia e Estatística, 2012. 215 p.

LOCATELLI, Caroline. **Desenvolvimento de Dashboard MQTT com Adafruit.IO**. Disponível em: https://curtocircuito.com.br/blog/Categoria%20IoT/desenvolvimento-de-dashboard-mqtt-com-adafruitio. Acesso em 14 jun.2023.

LOPEZ, M., HAMMES, G., VERGARA, L. **Desenvolvimento de Produtos Para A Diversidade Humana:** Ouvindo A Pessoa Surda (AC). Blucher Design Proceedings, v. 2, n. 4, p. 262-278, 2016.

NAIK, Nitin**. Choice of effective messaging protocols for IoT systems**: MQTT, CoAP, AMQP and HTTP. Tradução: Própria. In: 2017 IEEE international systems engineering symposium (ISSE). IEEE, 2017.

SANTOS, Bruno P. et al. **Internet das coisas:** da teoria à prática. Minicursos SBRC-Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuıdos, v. 31, p. 16, 2016. Disponível em: https://homepages.dcc.ufmg.br/~mmvieira/cc/papers/internet-das-coisas.pdf . Acesso em 14 jun.2023.

ZUQUETO, Douglas. **Adafruit IO** – Uma nova plataforma de IoT. Disponível em: https://www.makerhero.com/blog/adafruit-io-plataforma-iot/. Acesso em 14 jun.2023.