

UNIGOIÁS - CENTRO UNIVERSITÁRIO DE GOIÁS
CURSO DE ENGENHARIA DA COMPUTAÇÃO



AUTOMAÇÃO RESIDENCIAL PARA AUXÍLIO À TERCEIRA IDADE

JOMAR LEITE DE SOUZA SOBRINHO

SARAH MATIAS DOS SANTOS

GOIÂNIA
Dezembro/2020

**JOMAR LEITE DE SOUZA SOBRINHO
SARAH MATIAS DOS SANTOS**

AUTOMAÇÃO RESIDENCIAL PARA AUXÍLIO À TERCEIRA IDADE

Trabalho de Conclusão de Curso apresentado ao Centro Universitário de Goiás - UNIGOIÁS, sob orientação da Professora Ma. Ludmilla Reis Pinheiro dos Santos, como requisito parcial para obtenção do título de Bacharel em Engenharia da Computação.

GOIÂNIA
Dezembro/2020

FOLHA DE APROVAÇÃO

JOMAR LEITE DE SOUZA SOBRINHO

SARAH MATIAS DOS SANTOS

AUTOMAÇÃO RESIDENCIAL PARA AUXÍLIO À TERCEIRA IDADE

Trabalho de Conclusão de Curso apresentado à banca examinadora como requisito parcial para obtenção do Bacharelado em Engenharia da Computação do Centro Universitário de Goiás - UniGoiás, defendido e aprovado em ____ de ____ de _____ pela banca examinadora constituída por:

Orientadora: Prof^a. Ma. Ludmilla Reis Pinheiro dos Santos

Prof. Me. Murilo Parreira Leal

Prof. Esp. Tiago Marcelino Reis

Dedicamos este trabalho a Deus pela vida, às nossas famílias pelo amor, carinho e suporte, aos nossos amigos pelo apoio e diversão nos momentos que precisamos, e ao nosso querido amigo Rodrigo Botelho (in memorian).

AGRADECIMENTOS

A Deus pela saúde e oportunidades, às nossas famílias que sempre nos apoiaram e nos deram total suporte, à Prof.^a Ma. Ludmilla Reis Pinheiro dos Santos pela confiança e ajuda em nós investidas como nossa orientadora acadêmica, ao Prof. Me. Murilo Parreira Leal pela contribuição em nossa formação e pela amizade.

“Eu sei que a situação pode ser intimidadora. Você olha ao redor e é tudo assustador e diferente, mas sabe... encará-las, avançar contra elas como um touro – é assim que crescemos como pessoas.”

Rick Sanches

RESUMO

A automação assistiva compreende um conjunto de produtos, serviços e sistemas, que proporcionam qualidade de vida e independência da população idosa e de indivíduos com necessidades específicas. Este trabalho apresenta o projeto de um sistema de automação residencial assistiva, com foco na população idosa, usando conceitos de *Ambient Assisted Living* (AAL), para melhorar a qualidade de vida e prevenir acidentes, através de uma pulseira assistiva *wireless* equipada com um botão para solicitar socorro, motor vibratório para receber alertas e um microcontrolador ESP8266, monitorado por um servidor num microcontrolador ESP32. Também apresenta soluções ao cuidador, disponibilizando controle e monitoramento remoto por meio de um aplicativo mobile multiplataforma, desenvolvido usando o *framework Flutter* e o banco de dados *Firebase*, para monitorar sensores no ambiente para detecção de vazamento de gás de cozinha, possíveis focos de incêndio na residência e receber alertas de emergência do idoso.

PALAVRAS-CHAVE: Automação assistiva. População idosa. ESP32. *Flutter*.

ABSTRACT

Assistive automation comprises a set of products, services and systems, that provide quality of life and independence of the elderly population and individuals with specific needs. This work presents the project of an assistive home automation system, focused on the elderly population, using Ambient Assisted Living (AAL) concepts, to improve the quality of life and prevent accidents through a wireless assistive bracelet equipped with a button to call for help, a vibrating motor to receive alerts and an ESP8266 microcontroller, monitored by a server on an ESP32 microcontroller. It also presents solutions to the caregiver, providing remote control and monitoring through a multiplatform mobile application, developed using the Flutter framework and the *Firebase* database, to monitor sensors in the environment to detect kitchen gas leaks, possible fire outbreaks in the residence and receive emergency alerts from the elderly.

KEYWORDS: Assistive automation. Elderly population. ESP32. Flutter.

LISTA DE ABREVIATURAS E SIGLAS

AAL	<i>Ambient Assisted Living</i>
ETS	<i>Engineering Tools Software</i>
GLP	Gás Liquefeito de Petróleo
IBGE	Instituto Brasileiro de Geografia e Estatística
IP	<i>Internet Protocol</i>
mW	Miliwatt
nm	Nanômetro
ONU	Organização das Nações Unidas
PL	<i>Power Line</i>
ppm	Parte por milhão
RF	Rádio Frequência
SDK	<i>Software Development Kit</i>
SoC	<i>System on Chip</i>
TCP	<i>Transmission Control Protocol</i>
TP	<i>Twisted Pair</i>
UNFPA	Fundo de População das Nações Unidas
μ W	Microwatt

LISTA DE FIGURAS

Figura 1. Exemplo de automação no ambiente residencial.	18
Figura 2. Indicação das tecnologias que atendem aos problemas e demandas identificados pela pesquisa.	21
Figura 3. Componentes de software utilizados na implementação.	23
Figura 4. Visão geral do sistema.	24
Figura 5. Diagrama de blocos do sistema de automação proposto.	29
Figura 6. Módulo ESP32 NodeMCU-32S.	30
Figura 7. Módulo ESP-01.	31
Figura 8. Módulo MQ2.	32
Figura 9. Módulo MQ2.	33
Figura 10. Motor de Vibração SP1027.	34
Figura 11. Janela Principal do Arduino IDE.	35
Figura 12. Menu Preferências indicado campo do link para adicionar placas.	36
Figura 13. Gerenciador de Placas do Arduino IDE com a placa ESP32 instalada.	36
Figura 14. IDE Android Studio com o emulador integrado aberto.	37
Figura 15. Teste mútuo dos sensores de gás.	39
Figura 16. Diagrama de ligação dos sensores MQ2 e MQ6.	40
Figura 17. Diagrama de ligação do motor de vibração.	41
Figura 18. Pulseira de silicone da Mi Band 4.	43
Figura 19. Projeto da PCB da Pulseira Assistiva.	44
Figura 20. PCB da Pulseira Assistiva finalizada.	44
Figura 21. Comparação dos componentes depois das modificações.	45
Figura 22. Visão superior da placa da Pulseira Assistiva sem microcontrolador.	46
Figura 23. Visão superior da placa da Pulseira Assistiva com microcontrolador.	46
Figura 24. Projeto da PCB do servidor	47
Figura 25. PCB do servidor após corrosão.	47
Figura 26. PCB do servidor sem o ESP32.	48
Figura 27. PCB do servidor com todos componentes.	49
Figura 28. Projeto da PCB da base dos sensores de gás.	49
Figura 29. PCB da base dos sensores após o processo de corrosão.	50
Figura 30. PCB da base dos sensores sem os sensores.	51
Figura 31. PCB da base dos sensores montada.	51

Figura 32. Tela de abertura.	53
Figura 33. Tela de log in.	54
Figura 34. <i>Pop-up</i> para recuperação de senha.	55
Figura 35. Mensagem recebida ao solicitar recuperação de senha.	56
Figura 36. Tela para criação de nova senha.	57
Figura 37. Tela de cadastro.	58
Figura 38. Tela de cadastro com <i>SnackBar</i> de confirmação.	59
Figura 39. Tela principal do aplicativo.	60
Figura 40. Tela principal após ser aberto o menu do aplicativo.	61
Figura 41. Aplicativo monitorando detecção de fumaça no ambiente.	62
Figura 42. Notificação recebida após detecção de fumaça no ambiente.	63
Figura 43. Aplicativo monitorando detecção de gás no ambiente.	64
Figura 44. Notificação recebida após detecção de gás no ambiente.	65
Figura 45. Aplicativo reconhecendo pulseira assistiva desconectada.	66
Figura 46. Notificação recebida após o reconhecimento da desconexão da pulseira assistiva.	67
Figura 47. Aplicativo reconhecendo o acionamento do botão de pânico.	68
Figura 48. Notificação recebida após o reconhecimento do acionamento do botão de pânico.	69
Figura 49. <i>Pop-up</i> para confirmar a ciência do alarme recebido.	70

LISTA DE QUADROS

Quadro 1. Endereços para adicionar as placas ESP32 e ESP8266 ao Arduino IDE.	35
Quadro 2. Código teste do motor vibratório.	41
Quadro 3. Descrição dos valores das variáveis do banco de dados compartilhadas pelo servidor e aplicativo mobile.	42
Quadro 4. Descrição dos componentes da Pulseira Assistiva e seus respectivos valores.	45
Quadro 5. Descrição dos componentes do servidor e seus respectivos valores.	48
Quadro 6. Descrição dos componentes da base dos sensores e seus respectivos valores.	50

SUMÁRIO

1.	INTRODUÇÃO	14
1.1	Objetivos	15
1.1.1	<i>Objetivos Gerais</i>	<i>15</i>
1.1.2	<i>Objetivos Específicos</i>	<i>15</i>
1.2	Justificativa	15
1.3	Metodologia	16
2.	REFERENCIAL TEÓRICO	17
2.1	Automação Residencial	17
2.2	<i>Ambient Assisted Living</i>	<i>18</i>
2.3	<i>Wearable Computing</i>	<i>19</i>
2.4	Trabalhos Relacionados	20
2.4.1	<i>A Contribuição da Automação Residencial na Solução de Problemas de Acessibilidade no Cotidiano do Idoso</i>	<i>20</i>
2.4.2	<i>Proposta de um Sistema de Assistência Personalizada para Ambientes Inteligentes</i>	<i>22</i>
2.4.3	<i>Tecnologia Assistiva para Detecção de Quedas: Desenvolvimento de Sensor Vestível Integrado ao Sistema de Casa Inteligente</i>	<i>24</i>
2.4.4	<i>Sistema de Automatização Residencial de Baixo Custo Controlado Pelo Microcontrolador ESP32 e Monitorado Via Smartphone</i>	<i>26</i>
2.4.5	<i>Considerações sobre os Trabalhos Relacionados</i>	<i>26</i>
3.	PROPOSTA DE SOLUÇÃO	28
3.1	Sistema de automação proposto	28
3.2	Componentes de Hardware	29
3.2.1	<i>Microcontroladores</i>	<i>29</i>
3.2.2	<i>ESP32 NodeMCU-32S</i>	<i>30</i>
3.2.3	<i>Módulo ESP-01</i>	<i>31</i>
3.2.4	<i>Sensores de Gases</i>	<i>32</i>
3.2.5	<i>Módulo Sensor de Gás MQ2</i>	<i>32</i>
3.2.6	<i>Módulo Sensor de Gás MQ6</i>	<i>33</i>
3.2.7	<i>Motor de Vibração SP1027</i>	<i>33</i>
3.3	Componentes de Software	34

3.3.1	<i>Arduino IDE</i>	34
3.3.2	<i>Adicionando placas de desenvolvimento compatíveis à IDE</i>	35
3.3.3	<i>Android Studio</i>	37
3.3.4	<i>Flutter</i>	37
3.3.5	<i>Firebase</i>	38
4.	TESTES DOS COMPONENTES DE HARDWARE	39
4.1	Módulo Sensor de Gás MQ2 e MQ6	39
4.2	Motor de Vibração SP1027	40
4.3	Teste de Integração	41
5.	CONSTRUÇÃO DOS DISPOSITIVOS	43
5.1	Pulseira Assistiva	43
5.1.1	<i>Desenvolvimento da PCB da Pulseira Assistiva</i>	43
5.1.2	<i>Montagem da Pulseira Assistiva</i>	44
5.2	Desenvolvimento e Confeção da PCB para o ESP32	46
5.3	Base dos Sensores de Gás	49
5.4	Resultados Obtidos	51
6.	DESENVOLVIMENTO DO APLICATIVO	53
6.1	Tela de Abertura	53
6.2	Tela de Log in	54
6.3	Tela de cadastro	57
6.4	Tela Principal	59
6.4.1	<i>Status Ambiente Ideal</i>	61
6.4.2	<i>Status Fumaça Detectada</i>	62
6.4.3	<i>Status Gás Detectado</i>	63
6.4.4	<i>Status Pulseira Assistiva Desconectada</i>	65
6.4.5	<i>Status Botão Pressionado</i>	67
6.5	Resultados Obtidos	70
7.	CONSIDERAÇÕES FINAIS	71
7.1	Trabalhos Futuros	71
	REFERÊNCIAS	73
	APÊNDICE A – CÓDIGO TESTE DOS SENSORES DE GÁS	75
	APÊNDICE B – CÓDIGO DO SERVIDOR	76
	APÊNDICE C – CÓDIGO DA PULSEIRA ASSISTIVA	80

1. INTRODUÇÃO

O termo domótica surgiu com a união da palavra domus, cuja tradução é casa, com a palavra robótica. A definição de domótica é a automação de um ambiente doméstico, ou seja, a capacidade do ambiente ser controlado por um sistema, e tomar decisões (ESTEVAM JUNIOR; FARINELLI, 2019).

A domótica vem se tornando uma tendência nas residências, em busca de conforto e comodidade, as pessoas têm investido em automação. Segundo Zion Market Research (2017), o mercado de automação residencial global chegará a US\$ 53,45 bilhões até 2022. No entanto, são poucos os sistemas automatizados voltados para as necessidades dos idosos.

A população de idosos tende a aumentar nos próximos anos, devido a diminuição da taxa de mortalidade e ao aumento da expectativa de vida, em média 5 meses por ano de 1940 a 2017. Um estudo do Fundo de População das Nações Unidas (UNFPA), indica que a taxa de fecundidade da família brasileira vem diminuindo, o que forma a projeção de um cenário social onde viverão muitos idosos e poucos cuidadores, o que justifica um investimento em automação (IBGE, 2017; NAÇÕES UNIDAS DO BRASIL, 2018).

Um dos fatores que possibilitaram a maior longevidade são os cuidados médicos. No entanto, tomar muitos medicamentos pode se tornar um problema devido à quantidade, aos horários, que devem ser controlados rigorosamente, e a natural perda de memória. Um dosador automático de comprimidos pode auxiliar neste processo, sendo um dispositivo capaz de armazenar vários tipos de comprimidos, e ser programado para liberar uma dose em intervalos de tempo, indicados na receita médica. Além disso, esse dispositivo pode ser ligado a um sistema que permite avisar a alguém, idoso ou cuidador, se o comprimido não foi retirado, e emitir alertas.

Com o avanço da tecnologia, aumentou-se a variedade de controladores, sensores e atuadores, possibilitando o desenvolvimento de dispositivos vestíveis (wearable) que o idoso pode portar e acionar, caso precise chamar socorro bem como receber alertas sobre certas situações de perigo na residência.

1.1 Objetivos

Nas próximas subseções são apresentados os objetivos deste trabalho.

1.1.1 *Objetivos Gerais*

Este trabalho objetiva desenvolver um sistema de automação residencial para auxílio a pessoas na terceira idade, que disponibiliza ao idoso uma pulseira assistiva, e ao cuidador, monitoramento remoto de sensores, e da pulseira, por meio de um aplicativo mobile, que monitora o ambiente a fim de prevenir acidentes e garantir assistência ao idoso.

1.1.2 *Objetivos Específicos*

- Projetar e construir uma pulseira assistiva *wearable*, que permite ao idoso solicitar socorro e receber alertas;
- Monitorar sensor de gás e fumaça, para detectar vazamentos de gás na cozinha e suposto foco de incêndio;
- Desenvolver um aplicativo mobile multiplataforma, usando a linguagem de programação Dart com o *framework Flutter*, bem como o banco de dados em nuvem *Firebase*, que receberá os alertas dos sensores da residência e também da pulseira assistiva.

1.2 Justificativa

Promover assistência ao idoso e evitar acidentes por descuidos, devido a limitações individuais que surgem na terceira idade, como exemplo, lapso de memória.

No quesito promover assistência, o sistema supervisionará uma pulseira assistiva, que enviará um alerta ao cuidador ao ser acionado pelo idoso, caso necessitar de socorro.

No quesito prevenir acidentes, o sistema de automação através de sensores, irá detectar vazamento de gás de cozinha e fumaça, como prevenção de acidentes e detecção de focos de incêndio.

1.3 Metodologia

- Revisão da literatura;
- Projeto e implementação de automação residencial utilizando sensores;
- Projeto e implementação de aplicativo mobile;
- Projeto e implementação de um dispositivo com a funcionalidade de botão de pânico e notificador;
- Testes unitários de componentes;
- Testes de integração do sistema.

2. REFERENCIAL TEÓRICO

Este capítulo apresenta uma breve descrição dos seguintes temas: Automação residencial, *Ambient Assisted Living* (AAL), Tecnologias *wearable* e trabalhos relacionados ao tema.

2.1 Automação Residencial

Segundo Stevan Junior e Farinelli (2019), “A ideia de controlar dispositivos de forma inteligente já não é nova. Podemos citar que, em 1894, o sérvio naturalizado estadunidense Nikola Tesla demonstrou os primeiros experimentos de comunicação sem fio”.

Segundo Camargo (2014), a automação residencial “tem por objetivo a melhoria no conforto e na segurança das residências e dos condomínios”, usando para isso acendimento automático de luzes, porteiro eletrônico, portão automático, controle de acesso por meio de biometria, circuitos de monitoramento por vídeo, controle da temperatura ambiente, dentre outros.

Porém, uma definição mais ampla é feita por Alam, Reaz e Ali (2012), em que a automação residencial consiste em um ramo da computação ubíqua, que incorpora inteligência em moradias, em busca de conforto, assistência médica, segurança e conservação de energia, oferecendo qualidade de vida ao introduzir controle automatizado, serviços de assistência, de modo a permitir controle e monitoramento remoto.

Coulouris (2013) cita computação ubíqua ou pervasiva, como o uso de dispositivos computacionais que compõe os objetos diários, ao ponto de não serem mais notados pelo usuário, ou seja, comportamento computacional transparente e intimamente vinculado à sua função física. O termo ubíquo sugere que, o acesso a serviços de computação está onipresente, disponível em qualquer lugar.

Neste cenário a Figura 1 exemplifica o que pode ser controlado em um ambiente residencial automatizado.

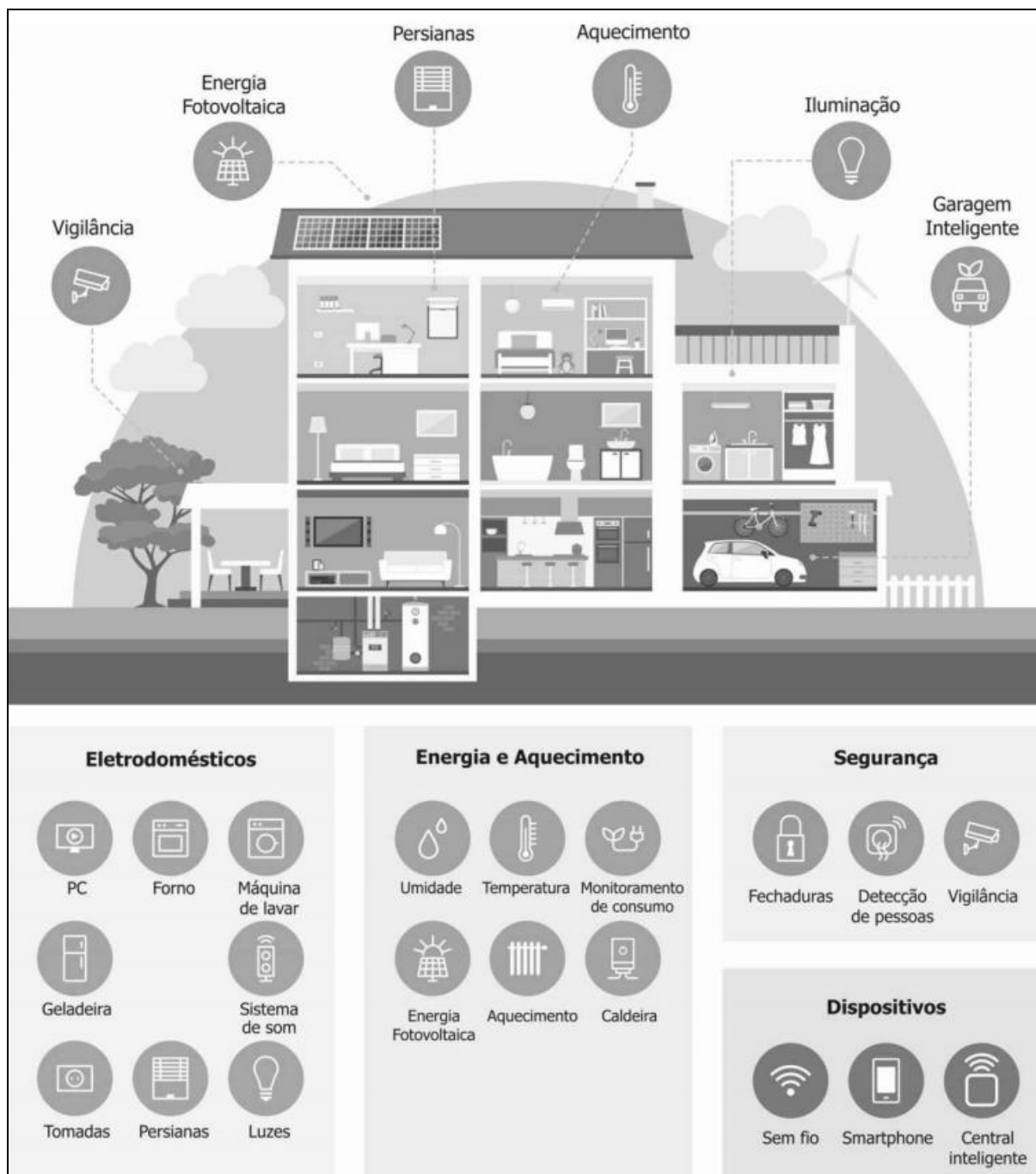


Figura 1. Exemplo de automação no ambiente residencial.

Fonte: ESTEVAM JUNIOR; FARINELLI (2019).

2.2 Ambient Assisted Living

O conceito de *Ambient Assisted Living* (AAL), também conhecido como automação assistiva, está relacionado com um conjunto de produtos, serviços e sistemas que procuram manter a qualidade de vida e independência da população idosa e de indivíduos com necessidades de cuidados específicos (GRGURIC, 2012). Sendo assim, AAL trata da

integração de tecnologias assistivas com o ambiente, caracterizada pelo uso de técnicas de domótica, no intuito de auxiliar atividades de rotina.

Há três gerações de tecnologias AAL: (a) primeira geração, surgiram os dispositivos pessoais de alarmes, por exemplo, botões de pânico; (b) segunda geração, dispositivos usando sensores eram instalados no ambiente, para detectar automaticamente potenciais casos de emergência; e (c) terceira geração, ocorreu o desenvolvimento de sistemas capazes de detectar e reportar problemas ou emergências no ambiente (BLACKMAN et al., 2015).

Como visto, a AAL é um ramo da automação, em que as inovações são diretamente relacionadas às evoluções de tecnologias das áreas de automação residencial, robótica e sensores.

2.3 *Wearable Computing*

Wearable computing, ou computação vestível, foi definido por Steve Mann na década de 1970 como: “Computação vestível permite novas formas de interação entre humanos e computadores, através de um pequeno computador programável usado junto ao corpo do usuário, que está sempre ligado, disponível e acessível” (OHTA et al., 2015).

Ohta et al. (2015) enumera dois recursos essenciais para o desenvolvimento de dispositivos vestíveis, que surgiram com o desenvolvimento da microeletrônica: (a) *System-on-Chip* (SoC), ou seja, sistemas eletrônicos em um único chip, que promove capacidade computacional em volumes pequenos; e (b) redução do consumo de energia para a ordem de mW/ μ W.

A computação vestível expandiu e se incorporou ao conceito de Internet das Coisas, principalmente pela ubiquidade de conexão com a Internet, por meio das redes sem fio. Neste contexto, Ohta et al. (2015), prevê a existência de três estágios para a computação vestível:

- a) *wearables* 1.0: estágio atual, em que se “veste” dispositivos vistos como dispositivos externos, por exemplo, relógios inteligentes, óculos de realidade virtual, sensores de batimento cardíaco, entre outros. As baterias tradicionais, são usadas por esses dispositivos como fontes de energia;
- b) *wearables* 2.0: próximo estágio em que os dispositivos reduzirão seu tamanho para milímetros, serão usados como acessórios em roupas, e usarão fontes

alternativas de energia, por exemplo, fibras geradoras de eletricidade estática, termoelétrica e vibração;

- c) *wearables* 3.0: estágio no qual, os dispositivos se integram de maneira imperceptível e ubíqua ao usuário, dificultando identificar se o usuário está ou não utilizando um dispositivo, uma vez que a tecnologia estará integrada ao corpo, por exemplo, lentes de contatos inteligentes, que medirão em tempo real a glicose e a pressão intraocular.

2.4 Trabalhos Relacionados

Nas próximas subseções serão apresentados trabalhos que abordam conceitos relacionados ao tema deste trabalho e possuem base conceitual sobre os diversos tipos de abordagens e tecnologias necessárias para a realização deste.

2.4.1 A Contribuição da Automação Residencial na Solução de Problemas de Acessibilidade no Cotidiano do Idoso

A dissertação “A Contribuição da Automação Residencial na Solução de Problemas de Acessibilidade no Cotidiano do Idoso”, aborda a contribuição da tecnologia na solução de problemas do cotidiano, vividos por idosos em suas residências, no que diz respeito a acessibilidade (SOUZA, 2015).

Souza (2015) fez uma pesquisa descritiva pela necessidade de caracterização dos idosos, bem como, das ofertas de tecnologias para automação residencial, com vistas à aplicação em possíveis soluções de problemas relacionados à acessibilidade. O foco foi a busca dos dados através de estudos de casos e questionários, tendo pessoas como fontes de coleta.

O objeto de estudo de Souza (2015) foram as tecnologias para automação residencial, voltadas às soluções de acessibilidade à pessoa idosa, sem distinção de custos, marcas ou aplicações.

Para identificar as principais doenças que afetam os idosos e quais as suas consequências em acessibilidade ou mobilidade, Souza (2015) levantou dados com base no IBGE, no Ministério das Cidades, em publicações específicas da área da saúde e em pesquisas similares. Nesse levantamento, pessoas com 60 anos ou mais declararam sofrer algum tipo de

doença, as principais foram: hipertensão; doença da coluna ou das costas e artrite ou reumatismo.

Para estudar os problemas vivenciados pelo idoso durante as realizações de tarefas cotidianas em sua residência, Souza (2015) realizou estudos de casos observando problemas com: acessibilidade, mobilidade e as atividades de vida diárias.

As tecnologias encontradas pelo autor que contribuem para a solução de problemas cotidianos dos idosos, e demandas identificados pela pesquisa são mostradas na Figura 2.

Tecnologias	Questões com os menores índices de satisfação										Problemas ou dificuldades detectados pelos estudos de casos					Tecnologias que atendem às demandas dos idosos
	03 – Sobre caminhar pela casa	04 – Sobre o uso de escadas	05 – Sobre guardar utensílios	06 – Sobre os riscos de acidentes	07 – Sobre procedimentos de segurança	09 – Sobre limpar a casa	11 – Sobre o uso do vaso sanitário	17 – Sobre fazer compras	20 – Sobre o uso de aparelhos de telefone	Dificuldades ao guardar utensílios	Inadequação do lay-out	Desníveis	Dificuldade em manusear objetos	Má qualidade da iluminação		
Leitor Biométrico																
Mensagens interativas																
Sensores de presença ou de movimento																
Simuladores de presença																
Central de vídeo																
Sensores de segurança																
Central de segurança																
Controle do consumo de energia, água e gás																
Climatização																
Controle à distância das lâmpadas, através de comando de voz																
Controle à distância das lâmpadas																
Controle à distância de equipamentos																

Figura 2. Indicação das tecnologias que atendem aos problemas e demandas identificados pela pesquisa.

Fonte: SOUZA (2015).

Entende-se ao analisar a Figura 2, que as tecnologias não atendem a todos os problemas de acessibilidade do cotidiano dos idosos e que alguns destes problemas não são relacionados à aplicação de tecnologia para sua solução. Souza (2015) identificou lacunas em tecnologias de automação residencial quanto a contribuição na solução dos problemas vividos pelos idosos, e concluiu que, os potenciais de atuação nas áreas em que existe demanda de atendimento ou problema a ser resolvido, a tecnologia ainda não atende de forma concisa. Sendo assim, a indústria de automação pode investir para melhor atender as necessidades dos idosos, nas áreas identificadas com potencial de atuação: segurança; circulação vertical quanto ao uso de escadas; atividades instrumentais da vida diária e o mobiliário.

2.4.2 *Proposta de um Sistema de Assistência Personalizada para Ambientes Inteligentes*

O principal objetivo da dissertação “***Proposta de um Sistema de Assistência Personalizada para Ambientes Inteligentes***”, foi a criação de um sistema capaz de prover assistência personalizada para usuários de Ambientes Inteligentes com foco especial em idosos e pessoas com diversidades funcionais, agindo de forma proativa de acordo com as preferências e necessidades do usuário, em situações específicas (VARGAS, 2019).

Para isso, Vargas (2016):

- desenvolveu um modelo de usuário capaz de representar corretamente as características relacionadas às limitações, condições de saúde e preferências de um indivíduo;
- criou um mecanismo capaz de inferir automaticamente um conjunto de preferências relacionadas a serviços, que tem um impacto significativo na realização das atividades de vida diárias do usuário;
- propôs e implementou uma arquitetura distribuída, flexível e escalável que definiu a interação entre o modelo de usuário, o mecanismo de inferência e a infraestrutura orientada a serviços presente no ambiente;
- integrou o sistema proposto, com a arquitetura orientada a serviços presente em um ambiente inteligente, onde essa arquitetura foi capaz de receber requisições e gerenciar a forma em que os serviços eram executados de acordo com os dispositivos presente no momento da requisição.

Vargas (2016) demonstrou o funcionamento do sistema através da aplicação em um conjunto de casos de uso padronizado definido pelo Projeto AAL Joint Programme. Esse

projeto foi criado pela *Active Assisted Living Programme* e promove ações voltadas à promoção de padrões e interoperabilidade no campo da AAL. A AAL Programme é uma associação de financiamento que visa criar melhor qualidade de vida para idosos, e fortalecer oportunidades industriais no campo do envelhecimento saudável utilizando tecnologia e inovação (AAL ASSOCIATION, 2019).

Vargas (2016) utilizou no sistema: (a) ferramentas da Web Semântica, usadas para representar as principais características do usuário e ambiente; (b) contexto do usuário e preferências em relação aos serviços disponíveis no ambiente; (c) paradigmas multiagentes para modelar a aplicação, como uma coleção de componentes chamados agentes, caracterizados pela autonomia, proatividade, capacidade de comunicação, entre outros.

A Figura 3 apresenta o diagrama que contém os principais componentes de software utilizados na implementação. Foi utilizado o software Protégé para implementar a ontologia e regras de inferência, a partir de um modelo conceitual. A linguagem de programação usada foi Java, auxiliada pelo: (a) framework *Java Agent Development Framework* (Jade), usado para implementar os agentes do sistema; (b) Pellet, provê serviços de inferência padrão; (c) OWL API, utilizada para permitir que a ontologia desenvolvida possa ser manipulada no código Java (VARGAS, 2019).

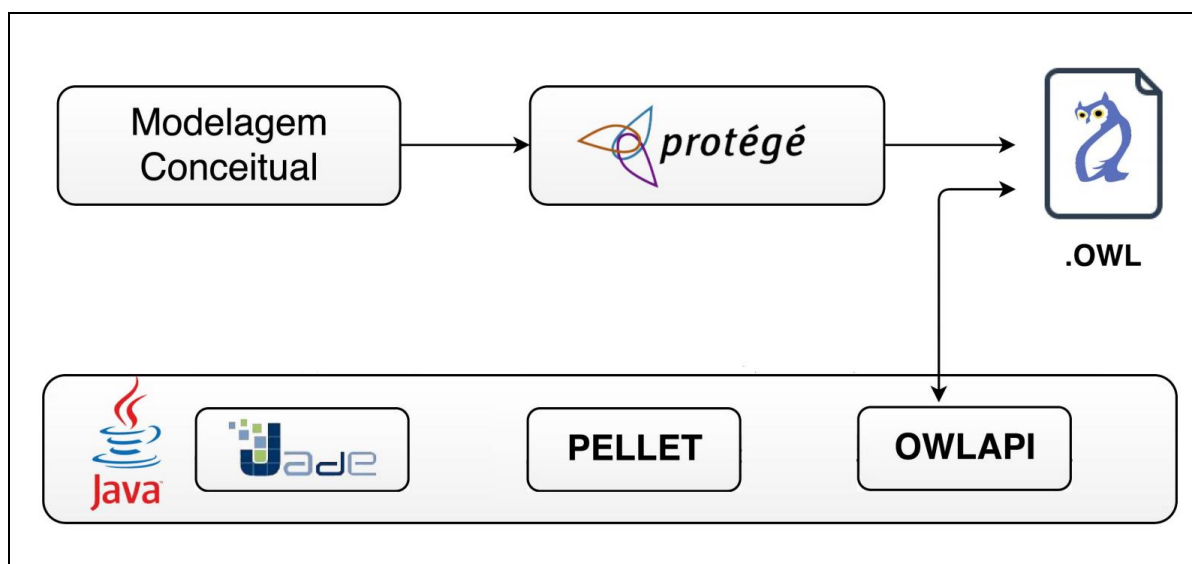


Figura 3. Componentes de software utilizados na implementação.
Fonte: VARGAS (2016).

Vargas (2016) incorporou um modelo padronizado e internacionalmente conhecido, para representar as deficiências, dificuldades e limitações dos usuários, aproveitando as vantagens do uso das tecnologias da Web Semântica, interoperabilidade e reuso do

conhecimento em diversos domínios da aplicação. Sendo assim, o sistema apresentou uma arquitetura distribuída, flexível, dinâmica e escalável, alcançando os objetivos propostos.

2.4.3 Tecnologia Assistiva para Detecção de Quedas: Desenvolvimento de Sensor Vestível Integrado ao Sistema de Casa Inteligente

Na dissertação “**Tecnologia Assistiva para Detecção de Quedas: Desenvolvimento de Sensor Vestível Integrado ao Sistema de Casa Inteligente**”, Torres (2018) baseado no aumento da população de idosos iminente, propôs o desenvolvimento e implementação de um dispositivo vestível de baixo consumo utilizando tecnologia EnOcean, capaz de detectar quedas do usuário, através de um sensor inercial contendo acelerômetro e giroscópio. Esse dispositivo foi projetado para ser parte de um sistema de automação com protocolo padrão mundialmente aberto, o KNX, integrado por uma plataforma de automação residencial open source, desenvolvida em Python: Home Assistant, instalada em um microcontrolador Raspberry PI 3.

A Figura 4 apresenta uma visão geral do sistema que resume o funcionamento e atuação de cada componente.

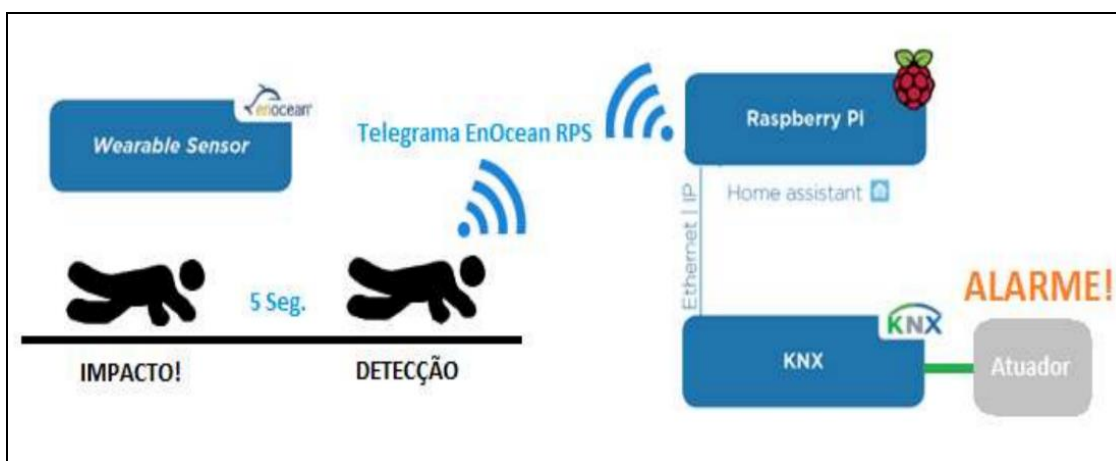


Figura 4. Visão geral do sistema.

Fonte: TORRES (2018).

O sensor vestível é composto por: (a) um módulo transmissor de rádio frequência (RF) STM300 EnOcean, que permite o desenvolvimento de sensores e atuadores com comunicação sem fio, de baixo consumo de energia, podendo ser gerada através de micro conversores como: painel solar, gerador térmico e gerador eletromecânico, dispensando o uso de bateria

em alguns casos; (b) sensor inercial MPU-9250 que possui acelerômetro, giroscópio e magnetômetro (TORRES, 2018).

Torres (2018) desenvolveu o sistema, utilizando a plataforma Home Assistant que suporta diversas tecnologias, a embarcou em um microcontrolador Raspberry PI 3, que permitiu a comunicação entre sensor vestível e o protocolo KNX. O microcontrolador possui conexão wireless, usada para comunicar com o módulo STM300, e conexão Ethernet, usada para comunicar com o KNX.

KNX é um protocolo global para automação que contém: (a) uma ferramenta de programação e concepção única e independente do fabricante, chamada Engineering Tools Software (ETS); (b) um conjunto completo de meios de ligação tais como par trançado (TP), powerline (PL), rádio frequência (RF) e protocolo de internet (IP); (c) conjunto completo de modos de configuração suportados, modo fácil e de sistema. Nesse contexto, o protocolo KNX foi utilizado para acionar o atuador, sendo que Torres (2018), não forneceu informações detalhadas, devido a um acordo de sigilo imposto pelo Instituto Fraunhofer FIT, quanto ao projeto.

Após simular quedas em vários sentidos, Torres (2018) avaliou os dados obtidos pelo sensor MPU-9250 e programou o módulo STM300 de forma que pudesse detectar a queda do usuário. Ao detectar, um sinal era enviado ao Home Assistant, que acionava o atuador, disparando um alarme. Além de validar a tecnologia EnOcean para uso em dispositivos vestíveis, o protótipo desenvolvido não indicou nenhum falso positivo através de testes realizados com dois usuários, de características corporais diferentes, onde foram reproduzidos 100 vezes cada um dos oito tipos de movimentos: quatro movimentos de quedas e quatro de não quedas.

Os testes realizados com o dispositivo revelaram: (a) 96% de sensibilidade, que é a capacidade do sensor de detectar quedas, obtida pela divisão dos positivos verdadeiros pela soma dos positivos verdadeiros e negativos falsos; e (b) 100% de especificidade, que é a capacidade de detectar somente a queda, obtida pela divisão dos negativos verdadeiros pela soma dos negativos verdadeiros e positivos falsos.

2.4.4 *Sistema de Automação Residencial de Baixo Custo Controlado Pelo Microcontrolador ESP32 e Monitorado Via Smartphone*

O trabalho “*Sistema de Automação Residencial de Baixo Custo Controlado pelo Microcontrolador ESP32 e Monitorado via Smartphone*”, teve por objetivo a implementação de um sistema de automação residencial, que possibilitasse acessibilidade e comodidade aos seus usuários. O projeto teve como base: (a) aplicação de um modelo autônomo para os principais controles elétricos de uma casa, através de uma plataforma embarcada no microcontrolador ESP32; (b) comunicação através da rede Wi-Fi, tornando possível a supervisão e controle residencial através de um servidor web; e (c) fundamentação do sistema na automação de baixo custo e de fácil aplicação (SANTOS; LARA JUNIOR, 2019).

Santos e Lara Junior (2019) utilizaram meios exploratórios para identificar os principais dados e variáveis a serem trabalhados, que tiveram como propósito, atuar sobre o controle das interrupções elétricas de uma casa, bem como reduzir o consumo energético, fornecendo um ambiente mais eficiente, confortável e seguro. Realizaram também um levantamento das variáveis presentes no ambiente, como: lâmpadas, atuadores, microcontrolador e sensores. Após a confecção e implantação do sistema, realizaram testes para comprovar o funcionamento.

Foram utilizados os seguintes componentes no projeto: módulo relé, sensor de luminosidade (LDR), sensor de presença (PIR), alarme *buzzer*, servo motor e controlador ESP32 (SANTOS; LARA JUNIOR, 2019).

Santos e Lara Junior (2019) implementaram um protótipo simulando um sistema residencial automatizado e eficiente, que englobou aspectos referentes a comodidade e racionamento energético, fundamentados no controle e supervisão remota, dos principais acionamentos elétricos presentes em um domicílio. Contudo, perceberam que para a implementação real do projeto, seria necessário implementar protocolos de segurança confiáveis, uma vez que o modelo de autenticação e acesso do Wi-Fi, podem ser violados.

2.4.5 *Considerações sobre os Trabalhos Relacionados*

Souza (2015) contribuiu com pesquisas sobre as tecnologias para automação residencial, voltadas às soluções de acessibilidade para pessoas idosas em âmbito nacional, relatando os problemas vividos por essas pessoas no contexto cultural brasileiro. Destacou

problemas ainda não solucionados pela automação residencial, porém não houve implementação.

Vargas (2016) implementou um sistema capaz de prover assistência personalizada para usuários de ambientes inteligentes com foco em idosos e pessoas com diversidades funcionais, contudo, se baseou em um programa europeu de padrão de interoperabilidade, AAL Joint Programme, que apesar de ser padronizado internacionalmente, pode não atender casos específicos devido a diferença cultural, dependendo da habilidade do programador ao usar a Web Semântica para minimizar esse impasse. O autor se destacou por desenvolver um sistema distribuído, dinâmico e escalável.

Torres (2018) desenvolveu e implementou um sensor *wearable*, capaz de detectar quedas do usuário, visando aproveitar os benefícios das tecnologias envolvidas. Apesar de ter sucesso com o alcance dos objetivos, um sistema que priorizasse a prevenção de quedas, seria mais importante no contexto abordado de tecnologia assistiva.

Santos e Lara Junior (2019) desenvolveram um sistema de automação residencial de baixo custo e monitorado via *smartphone*. Os objetivos foram alcançados, porém os componentes usados foram de prototipagem, e podem apresentar limitações na implementação *in loco*. O microcontrolador usado apresenta bom desempenho comparado aos do segmento, porém deixaram de usar os recursos que o destaca.

Após essas considerações, o próximo capítulo apresenta a proposta de solução deste trabalho.

3. PROPOSTA DE SOLUÇÃO

Este capítulo apresenta a proposta de solução deste trabalho, bem como a apresentação dos componentes que a compõe.

3.1 Sistema de automação proposto

O sistema de automação proposto tem a particularidade de não ser desenvolvido para o idoso executar comandos complexos. Dessa forma, não há necessidade do idoso dominar uma nova tecnologia, apenas usufruir dos cuidados que a automação e a computação pervasiva oferecem.

O sistema que controlará o dispositivo e sensores, será supervisionado por um aplicativo mobile multiplataforma, que poderá receber alertas do dispositivo a ele conectado e dos status dos sensores. Dessa forma, o cuidador conseguirá monitorar situações críticas na residência, através de sensores que enviarão informações ao aplicativo:

- Sensor MQ2: poderá ser instalado por toda a residência com finalidade de detectar fumaça. Caso haja detecção, o sistema emitirá um alerta no dispositivo mobile do cuidador, sobre possível foco de incêndio;
- Sensor MQ6: poderá ser instalado na cozinha da residência, com a finalidade de detectar GLP (gás de cozinha). Caso haja detecção, o sistema emitirá um alerta no dispositivo mobile do cuidador, sobre vazamento de gás;

A pulseira assistiva garantirá assistência ao portador, pois: (a) conterá um botão que ao ser acionado em caso de emergência, enviará um alerta ao dispositivo mobile do cuidador; e (b) vibrará quando for detectado vazamento de gás ou fumaça na residência.

A Figura 5 apresenta o diagrama de blocos que representa o sistema de automação proposto.

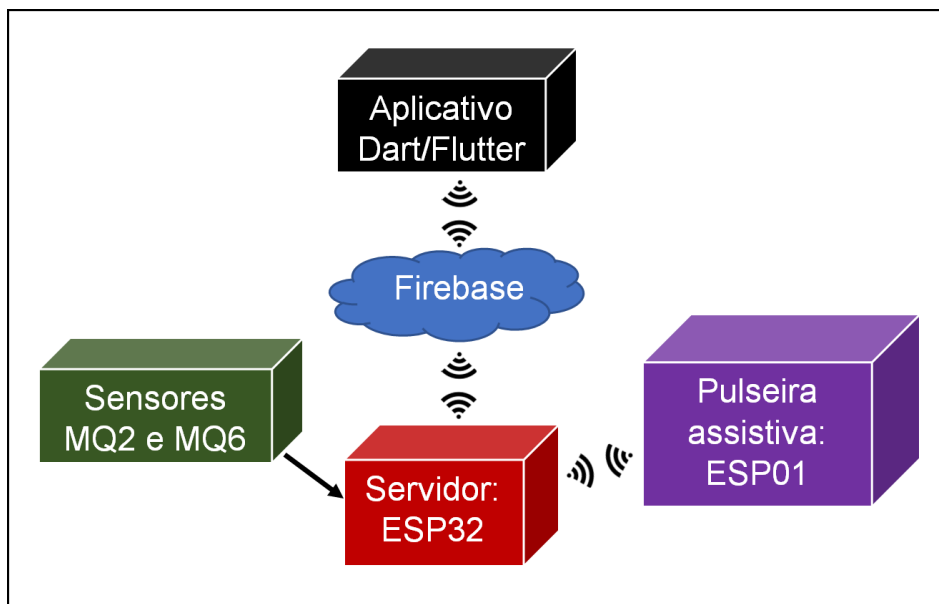


Figura 5. Diagrama de blocos do sistema de automação proposto.
Fonte: Autoria própria.

O microcontrolador ESP32 estará conectado ao banco de dados na nuvem *Firebase*, onde serão disponibilizados os dados para o aplicativo mobile, desenvolvido utilizando a linguagem de programação Dart, juntamente com o framework *Flutter*. O microcontrolador ESP32 enviará e receberá dados da pulseira assistiva e do *Firebase*, e receberá dados dos sensores MQ2 e MQ6. O aplicativo comunicará com o *Firebase* para receber e enviar dados ao sistema.

A pulseira assistiva usará a rede Wi-Fi para se comunicar, já os sensores MQ2 e MQ6, serão conectados ao sistema via fios.

3.2 Componentes de Hardware

Nos próximos subitens são apresentados os componentes de hardware utilizados neste trabalho.

3.2.1 Microcontroladores

Almeida, Moraes e Seraphim (2016) definem microcontroladores como: circuitos eletrônicos que reúnem em um único chip, a unidade de processamento, as memórias e um conjunto de periféricos de entrada e saída. Sua particularidade é poder ser usado em diversos projetos sem que seja necessário reconstrução de hardware. Para isso, ele executa uma

sequência de operações definida por um código armazenado em sua memória, onde a saída de dados depende das entradas, bem como das operações realizadas.

A seguir são apresentados os microcontroladores que compõem o projeto.

3.2.2 ESP32 NodeMCU-32S

Com tecnologia de 40 nm, o microcontrolador ESP32 oferece uma plataforma robusta e integrada, pois além de possuir dois núcleos de processamento, tem radio Wi-Fi e Bluetooth no mesmo chip. A Figura 6 apresenta o microcontrolador.

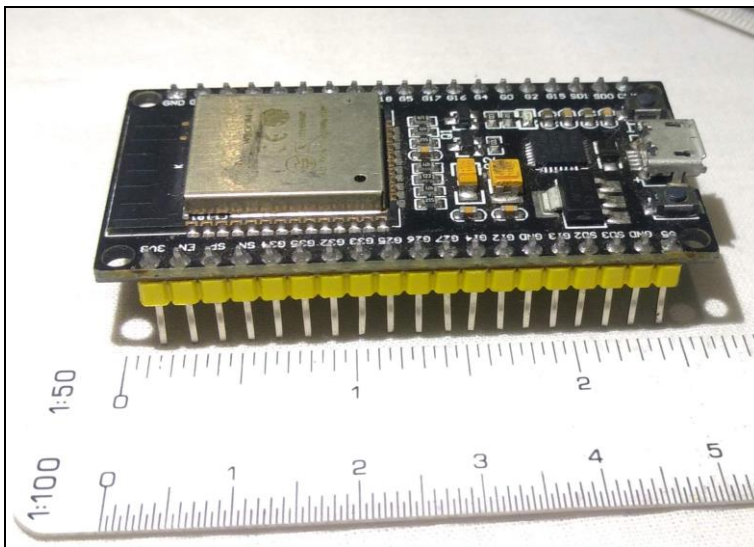


Figura 6. Módulo ESP32 NodeMCU-32S.

Fonte: Autoria própria.

Principais características:

- Modelo: NodeMCU-32S;
- Tensão de operação: 2.2V – 3.6 DC (5V via Micro USB);
- Corrente de operação (típica): 80mA;
- Nível lógico: 3.3V;
- Controlador: ESP-WROOM-32;
- Microprocessador: Tensilica Xtensa 32-bit LX6;
- Clock: até 240MHz;
- SRAM: 520KB;
- Memória flash externa de 32Mb (4 megabytes);
- Portas GPIO: 36;

- GPIO: função PWM, I2C e SPI
- Suporte a redes Wi-Fi: 802.11 b/g/n (2,4 a 2,5GHz);
- Bluetooth: v4.2 BR / EDR e BLE (Bluetooth Low Energy).

3.2.3 Módulo ESP-01

O módulo ESP-01 é composto pelo microcontrolador ESP8266, de um chip de memória flash externo, utilizado no armazenamento de programas e de arquivos, e de uma antena impressa na própria placa. Essa placa é bem pequena em relação as outras, sendo uma boa opção para desenvolvimento de dispositivos wearables. A Figura 7 apresenta o microcontrolador.

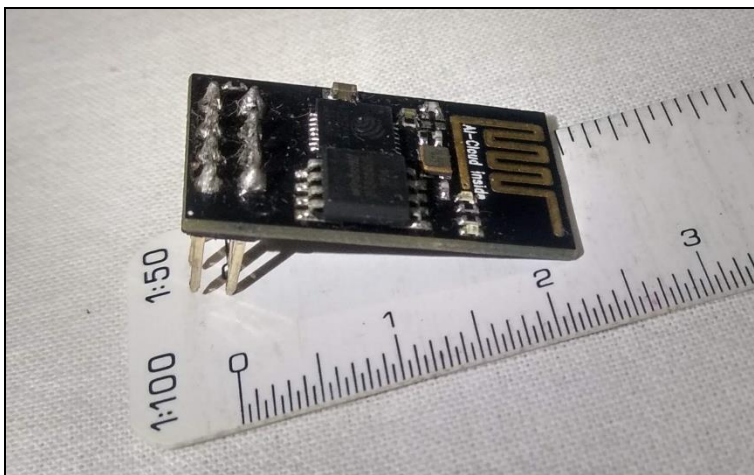


Figura 7. Módulo ESP-01.

Fonte: Autoria própria.

Principais características:

- Modelo: ESP-01;
- Tensão de operação: 3,3V;
- Nível lógico: 3.3V;
- Clock: 80 MHz, com possibilidade de operar em 160MHz;
- Arquitetura RISC de 32 bits;
- 32KB de RAM para instruções;
- 96KB de RAM para dados;
- Suporte à redes Wi-Fi: 802.11 b/g/n;
- Comunicação: Serial (TX/RX);

- Suporta comunicação TCP e UDP.

3.2.4 Sensores de Gases

Stevan Junior e Farinelli (2019) destacam os sensores de gás, devido sua importância no monitoramento de gases inflamáveis ou mesmo gases nocivos à saúde.

Esses sensores geralmente são instalados em cozinhas, áreas de serviço e próximos a eletrodomésticos como fogão e aquecedores a gás.

O material sensível dos sensores de gás é o Dióxido de Estanho, este composto químico varia sua condutividade em relação à pureza do ar.

3.2.5 Módulo Sensor de Gás MQ2

O sensor de gás MQ2 mostrado na Figura 8, é altamente sensível aos gases GLP, propano e hidrogênio, porém pode ser usado para detecção de fumaça. Os dados técnicos do sensor MQ2 são: (a) concentração de detecção de 300 a 10.000 ppm; (b) tensão de operação 5V; (c) saída analógica e digital, por meio do uso de um comparador de nível, (d) pinagem VCC de 5V, GND, D0 para saída digital e A0 para saída analógica.

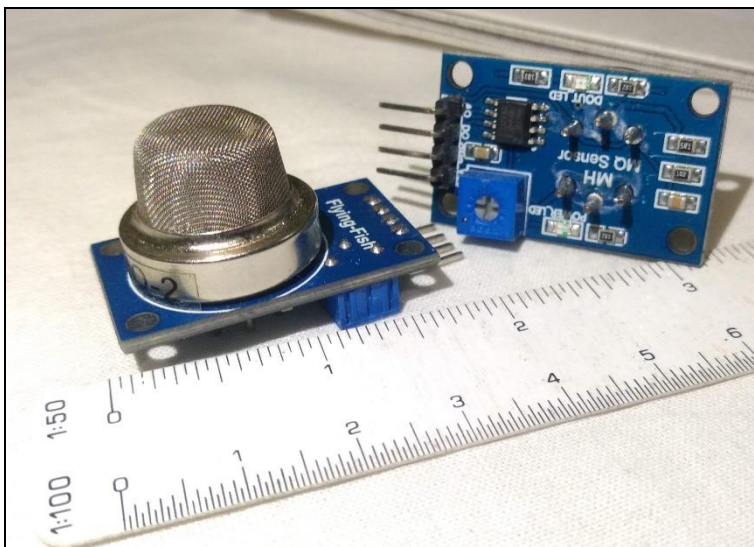


Figura 8. Módulo MQ2.
Fonte: Autoria própria.

3.2.6 Módulo Sensor de Gás MQ6

O sensor de gás MQ6 mostrado na Figura 9, possui alta sensibilidade para GLP, isobutano e propano, e baixa sensibilidade para fumaça. Os dados técnicos do sensor MQ6 são: (a) concentração de detecção de 200 a 10.000 ppm; (b) tensão de operação 5V; (c) saída analógica e digital, por meio do uso de um comparador de nível, (d) pinagem VCC de 5V, GND, D0 para saída digital e A0 para saída analógica.

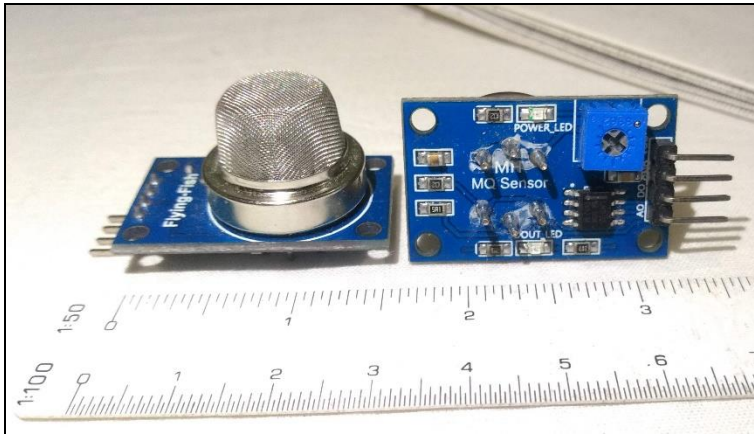


Figura 9. Módulo MQ2.

Fonte: Autoria própria.

3.2.7 Motor de Vibração SP1027

O motor de vibração SP1027 apresentado na Figura 10, é um motor elétrico, pequeno e achatado, ideal para dispositivos *wearable*. Os dados técnicos do motor de vibração SP1027 são: (a) tensão de operação 2,5V até 4V; (b) corrente máxima de operação 90mA; (c) velocidade de rotação 9000 RPM, (d) dimensão total 40x10x3mm.

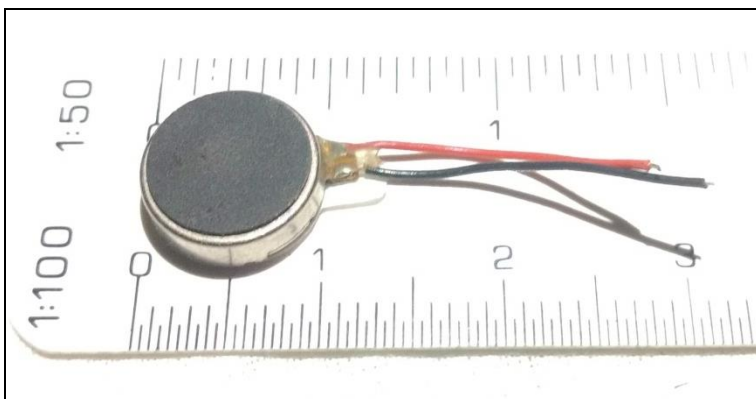


Figura 10. Motor de Vibração SP1027.

Fonte: Autoria própria.

3.3 Componentes de Software

Nos próximos subitens são apresentados os componentes de software utilizados neste trabalho.

3.3.1 *Arduino IDE*

O Arduino IDE (Integrated Development Environment) é um ambiente de desenvolvimento gratuito voltado principalmente à programação da plataforma Arduino (ESTEVAM JUNIOR; FARINELLI, 2019). A Figura 11 apresenta a janela principal do Arduino IDE indicando seus principais elementos.

Os elementos indicados na Figura 11 são: (1) barra de ferramentas, onde se encontra toda a configuração do hardware; (2) botão verificar, faz a compilação do código verificando se não há erros de sintaxe no código escrito em linguagem C ou C++; (3) botão carregar, realiza a transmissão do código após compila-lo ao hardware configurado; (4) função setup, onde é feita a configuração do hardware, é a primeira rotina a ser executada pelo microcontrolador e é executada apenas uma vez, e (5) função loop, contém toda a lógica de programação do sistema, é executada repetidamente após a função setup terminar.

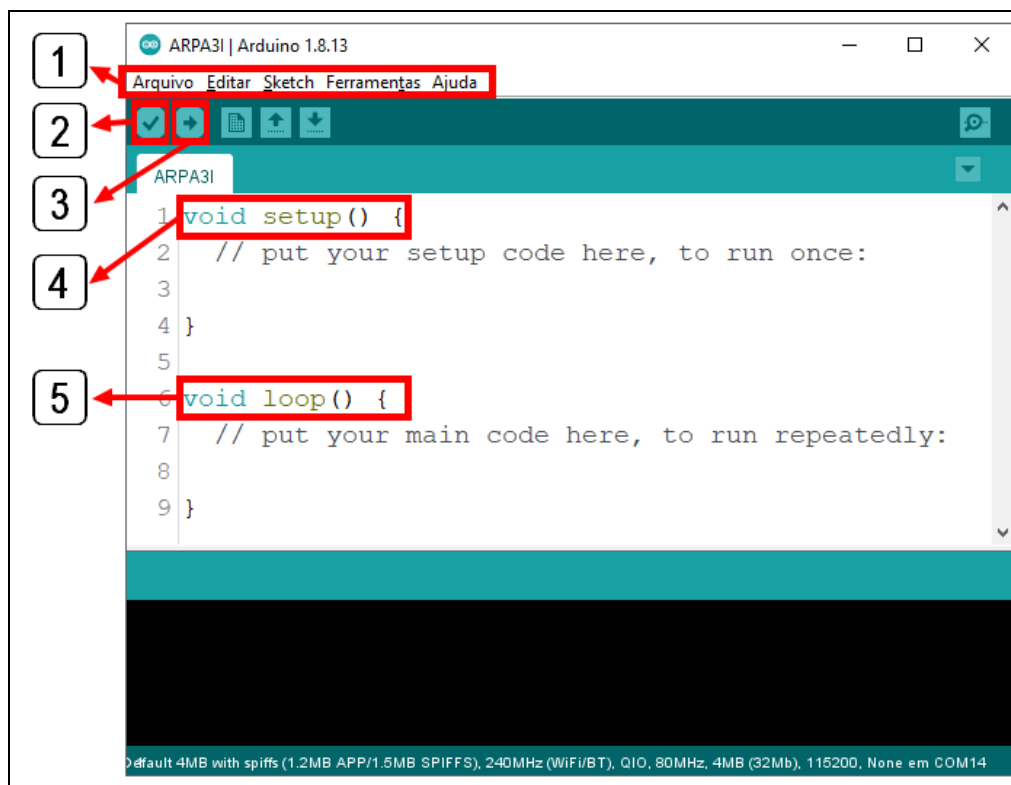


Figura 11. Janela Principal do Arduino IDE.

Fonte: Autoria própria.

Nativamente o Arduino IDE não é compatível às placas ESP32 e ESP8266, porém é possível realizar a compatibilização através de procedimentos simples, (ESTEVAM JUNIOR; FARINELLI, 2019).

3.3.2 Adicionando placas de desenvolvimento compatíveis à IDE

Para instalar as placas utilizadas neste trabalho, utilizaremos a ferramenta denominada Gerenciador de placas que pode ser alcançada através do caminho: Ferramentas → Placas → Gerenciador de placas. Geralmente, no primeiro momento as placas não são encontradas na busca, para isso acessar: Arquivo → Preferências, e adicionar os endereços indicados no Quadro 1, no local indicado na Figura 12.

<https://raw.githubusercontent.com/espressif/arduino-esp32/gh-pages/package_esp32_index.json>
 <https://arduino.esp8266.com/stable/package_esp8266com_index.json>

Quadro 1. Endereços para adicionar as placas ESP32 e ESP8266 ao Arduino IDE.

Fonte: Autoria própria.

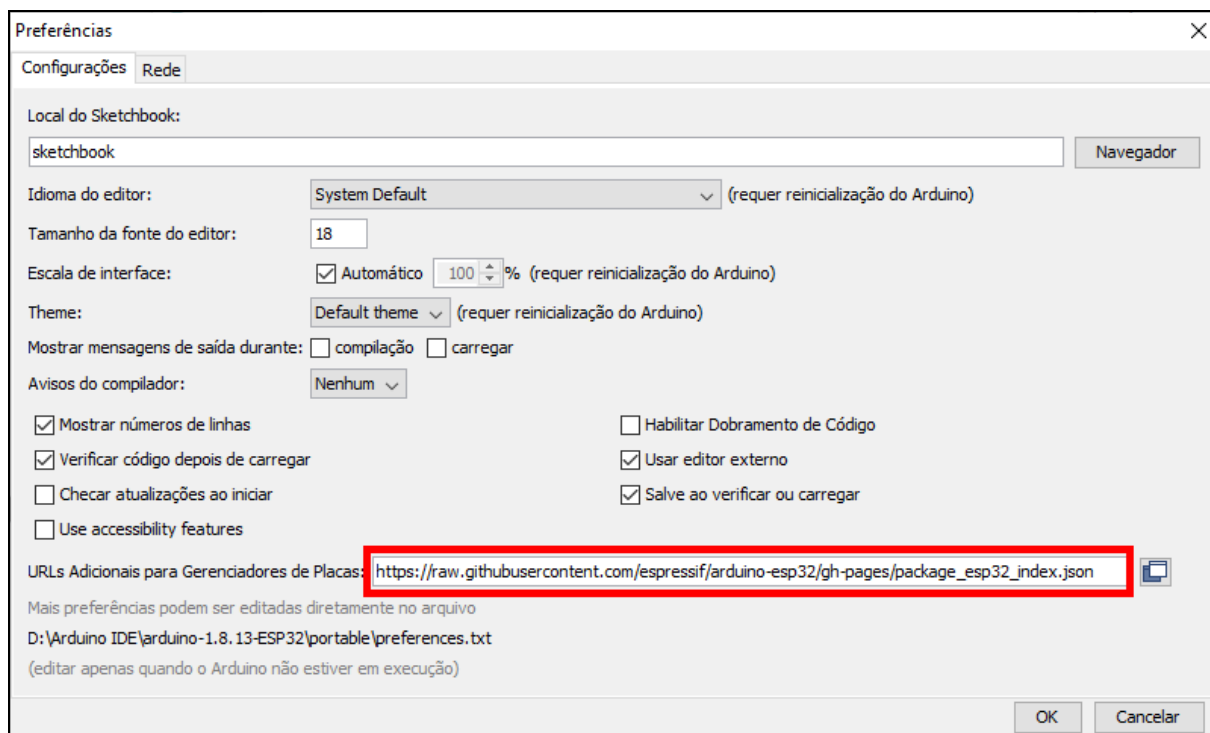


Figura 12. Menu Preferências indicado campo do link para adicionar placas.
Fonte: Autoria própria.

Após confirmar a configuração, as respectivas placas já são encontradas na busca do Gerenciador de Placas conforme a Figura 13, sendo necessário escolher a opção instalar para iniciar a instalação e finalizar o procedimento.

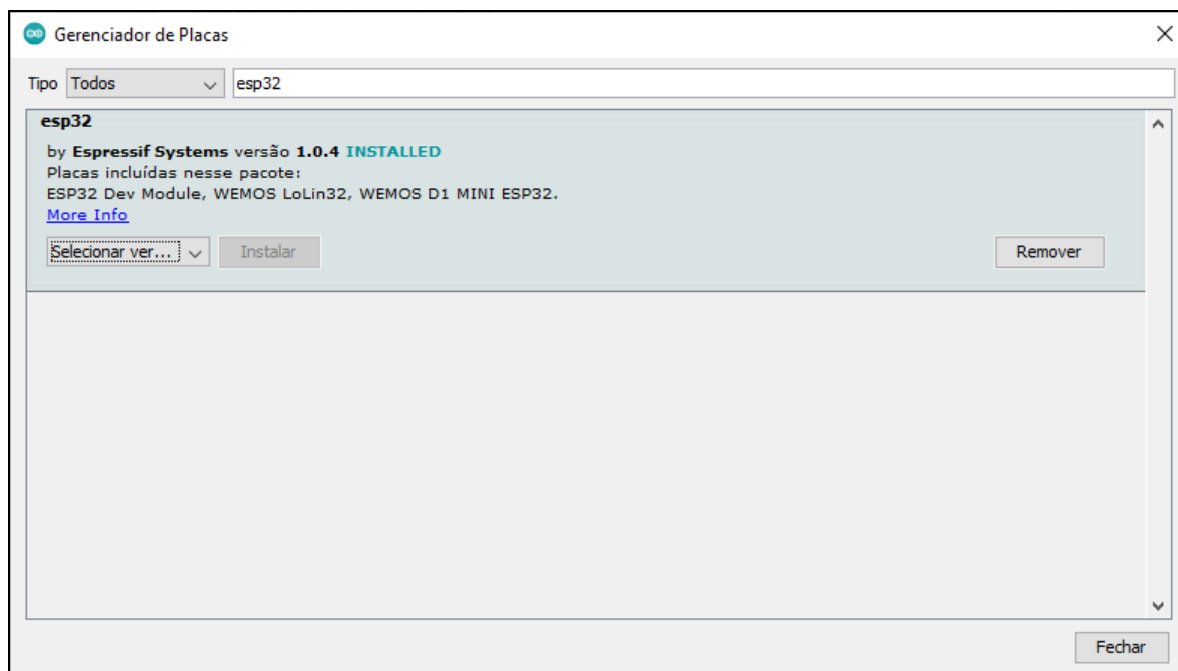


Figura 13. Gerenciador de Placas do Arduino IDE com a placa ESP32 instalada.
Fonte: Autoria própria.

3.3.3 Android Studio

O Android Studio é a IDE oficial do Google para o desenvolvimento de aplicativos Android, que com o auxílio de plug-ins, oferece uma experiência completa e integrada para o desenvolvimento com *Flutter* (GONÇALVES, 2016).

Além de proporcionar a facilidade de integração com o *framework Flutter* ele disponibiliza um emulador Android na própria IDE para executar o aplicativo sem a necessidade de um dispositivo.

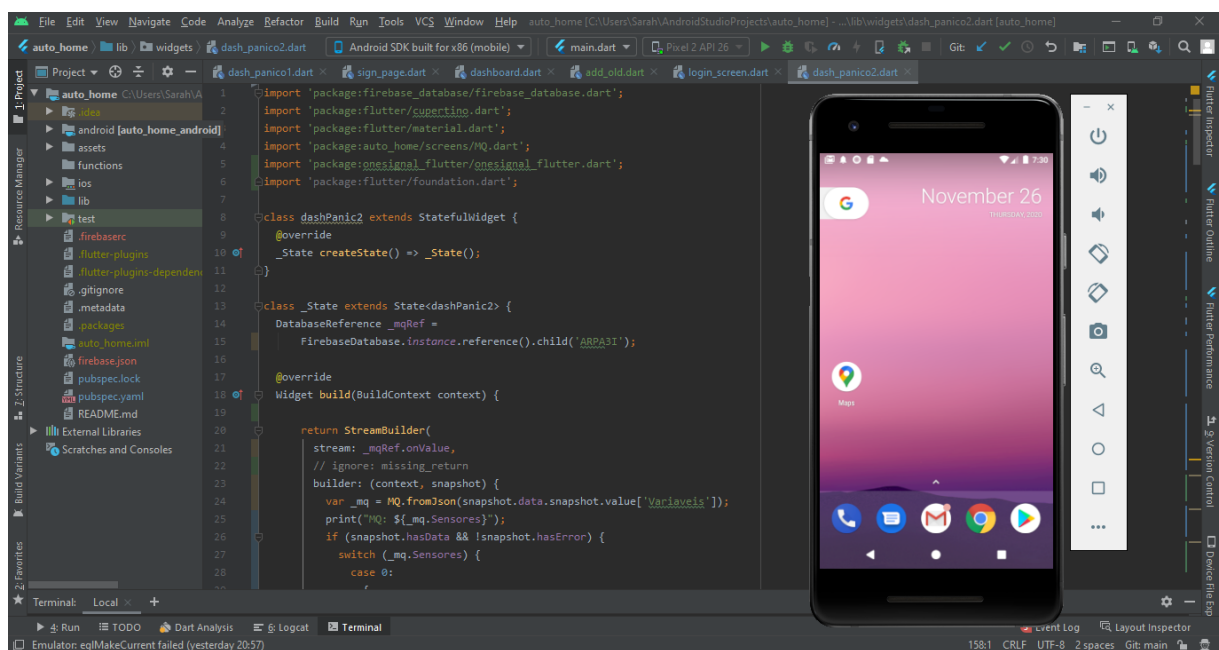


Figura 14. IDE Android Studio com o emulador integrado aberto.

Fonte: Autoria própria.

3.3.4 Flutter

Flutter é um kit de desenvolvimento de software (SDK) de código aberto, que permite desenvolver aplicações mobile na linguagem de programação Dart (CORAZZA, 2018).

As vantagens de se usar *Flutter* são:

- Alta produtividade: visto que o aplicativo desenvolvido usando *Flutter* ser multiplataforma, o código criado pode ser compilado para executar no Android e no iOS;

- b) Alta performance: Dart é compilado em código nativo, sendo assim, não há necessidade de intermediários entre o hardware e o software. Até então, é o único SDK que fornece essa comunicação sem exigir uma ponte JavaScript;
- c) Simplicidade: a programação com *Flutter* é considerada fácil, sendo possível não programadores fazerem aplicações, pois não exige conhecimentos avançados. Além disso, os *widgets* são fáceis de se manipular, eles definem elementos estruturais, estilísticos e de layout. São disponibilizados pela ferramenta já prontos e com a possibilidade de serem customizados;
- d) Robusto: é possível realizar a integração com os recursos nativos do dispositivo móvel através das bibliotecas e de plug-ins instalados durante o desenvolvimento.

Apesar de ser relativamente novo, criado em 2015, grandes empresas já o utiliza em seus aplicativos, por exemplo, Nubank, Beetech, Banco BS2 e Organizações Globo.

3.3.5 *Firebase*

O *Firebase* é uma plataforma de desenvolvimento mobile e web, que oferece serviços de fácil usabilidade, e infra estruturas que auxiliam desenvolvedores a construir aplicativos, sendo um agrupamento de diversos serviços, dentre eles o *Realtime Database* e *Cloud Firestore*. (GONÇALVES, 2016).

Neste trabalho foi usado o *Realtime Database*, que é um banco de dados em tempo real, e o *Cloud Firestore*, que também é um banco de dados em tempo real, porém mais novo, possui consultas mais avançadas e rápidas, além de melhor escalabilidade.

A maioria dos serviços disponíveis podem ser usados de forma gratuita, se tornando uma ótima opção para iniciar o desenvolvimento.

4. TESTES DOS COMPONENTES DE HARDWARE

Este capítulo apresenta os testes específicos de cada componente de hardware utilizado no projeto, além de seus respectivos códigos de programação, esquemas de montagem, e por fim, o teste de integração.

4.1 Módulo Sensor de Gás MQ2 e MQ6

Para testar estes módulos, foi usado um recipiente transparente para simular uma câmara de gás. Os sensores foram conectados ao circuito usando jumpers, e foram colocados dentro do recipiente, hora separados, hora juntos, e comparado os resultados. Além dos sensores MQ2 e MQ6, foram testados os sensores MQ4 e MQ5 para analisar os resultados das leituras e escolher os módulos que melhor atendiam a proposta do trabalho.

No intuito de testar os sensores num cenário de possível incêndio, introduziu-se os gases: fumaça resultante da queima de papelão, GLP, vapor de álcool, fumaça de cigarro, fumaça da queima de incenso e fumaça da queima de borracha.

Após analisar os dados coletados da simulação, concluiu-se que: (a) todos sensores detectaram GLP, porém não com a mesma sensibilidade, (b) GLP se assemelha ao vapor de álcool, (c) nem todos sensores são sensíveis a fumaça da queima de papelão, (d) a detecção da queima de incenso é semelhante a queima de papelão, sendo exclusivamente usado a partir de então, durante o restante do desenvolvimento, (e) fumaça de cigarro é quase imperceptível para a maioria dos sensores, (f) não foi detectado fumaça da queima de borracha.

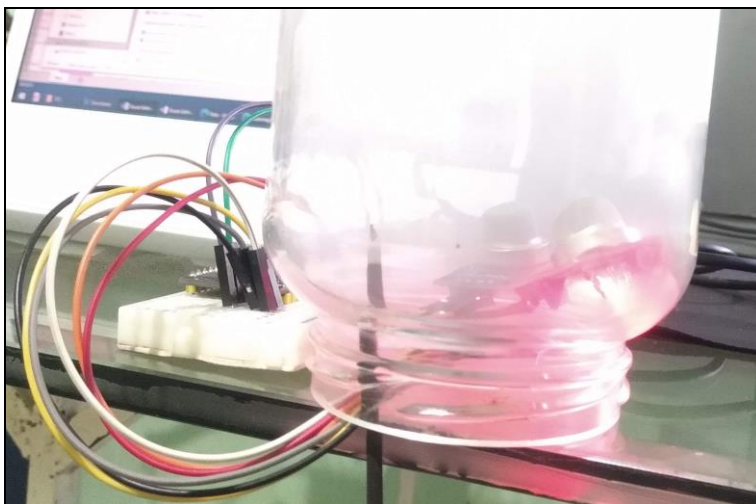


Figura 15. Teste mútuo dos sensores de gás.
Fonte: Autoria própria.

Com base nestas conclusões, o sensor MQ2 foi escolhido por ser mais sensível a fumaça que os demais, e o sensor MQ6 por ser mais sensível ao GLP e menos sensível a fumaça.

Para a realização dos testes foi usado o circuito mostrado na Figura 16, e o código de teste no Apêndice A.

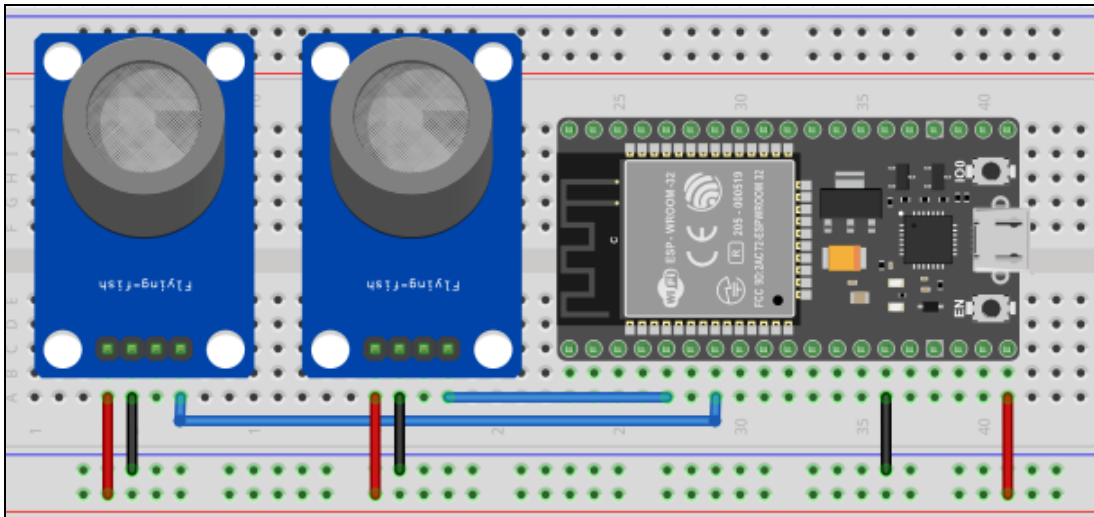


Figura 16. Diagrama de ligação dos sensores MQ2 e MQ6.

Fonte: Autoria própria.

4.2 Motor de Vibração SP1027

O objetivo do motor de vibração é notificar o idoso portador da pulseira assistiva, sobre uma eventual detecção de fumaça ou vazamento de gás, bem como qualquer alerta que o sistema necessitar enviar ao idoso.

O motor de vibração foi testado usando o ESP-01, conforme diagrama da Figura 17, sendo que o Quadro 2 apresenta o código teste do motor de vibração.

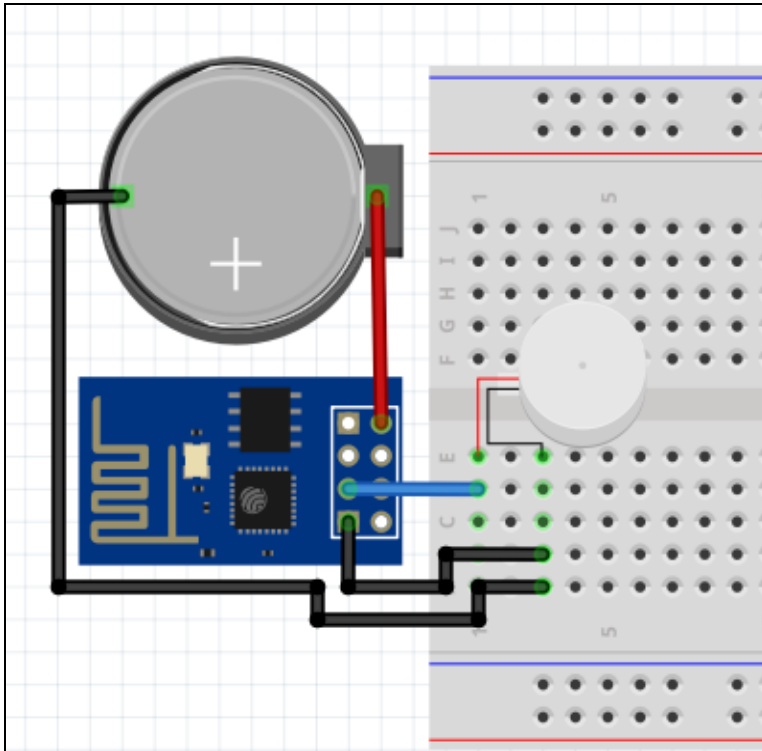


Figura 17. Diagrama de ligação do motor de vibração.
Fonte: Autoria própria.

```

1  int motorVib = 2;
2  void setup() {
3    pinMode(motorVib, OUTPUT);
4  }
5  void loop() {
6    digitalWrite(motorVib, HIGH);
7    delay(500);
8    digitalWrite(motorVib, LOW);
9    delay(1000);
10 }

```

Quadro 2. Código teste do motor vibratório.
Fonte: Autoria própria.

4.3 Teste de Integração

Testou-se os modos ponto de acesso e estação do ESP32, sendo o modo estação escolhido, pois para o contexto do projeto o gerenciamento de rede externo seria mais adequado, devido as limitações do modo ponto de acesso, sendo que possibilitaria dimensionar a área de cobertura da conexão entre os dispositivos sem manutenção no sistema.

Foram testados vários tipos de protocolos de comunicação nas placas ESP32 e ESP01, sendo o mais adequado ao contexto do trabalho o protocolo TCP, pois ele proporcionou: (a)

segurança devido a garantia de entrega de dados, (b) alto desempenho e (c) escalabilidade, pois aceitou a adição de outros dispositivos posteriormente ao servidor.

A comunicação entre o servidor e o aplicativo foi feita através do *Firebase*. No banco de dados em nuvem da plataforma, foi criada a variável chamada “Pânico” para receber atualização de estado do botão de pânico, e a variável chamada “Sensores” para as atualizações dos sensores de gás. Dessa forma, o código foi desenvolvido para enviar os códigos do Quadro 3 ao banco de dados, com base nos estados dos dispositivos:

Variável	Valor	Descrição
Pânico	0	Pulseira Assistiva conectada ao servidor.
	1	Botão de pânico da Pulseira Assistiva pressionado.
	2	Pulseira Assistiva desconectada do servidor.
Sensores	0	Não há detecção de fumaça ou gás no ar.
	1	Detecção de fumaça no ar.
	2	Detecção de GLP no ar.

Quadro 3. Descrição dos valores das variáveis do banco de dados compartilhadas pelo servidor e aplicativo mobile.

Fonte: Autoria própria.

5. CONSTRUÇÃO DOS DISPOSITIVOS

Este capítulo apresenta a construção da pulseira assistiva, do servidor com ESP32, e da base dos sensores de gás.

5.1 Pulseira Assistiva

A pulseira assistiva foi construída utilizando a pulseira de silicone da Xiaomi, modelo Mi Band 4, conforme a Figura 18.



Figura 18. Pulseira de silicone da Mi Band 4.
Fonte: Autoria própria.

O subitem a seguir apresenta a confecção da PCB (placa de circuito impresso) do dispositivo.

5.1.1 Desenvolvimento da PCB da Pulseira Assistiva

O material escolhido para a confecção da PCB foi fenolite cobreado, e a técnica usada foi o processo de transferência térmica. Essa técnica consiste em imprimir o projeto da placa usando impressão a laser em papel glossy, e esquentar o papel voltado com a impressão para o lado cobreado do fenolite até que o toner derreta e adira a placa. A Figura 19 apresenta o projeto da PCB ampliado, pronto para impressão.



Figura 19. Projeto da PCB da Pulseira Assistiva.
Fonte: Autoria própria.

Diferentes tipos de versões foram criadas, pois a placa ESP01 possui propriedades ao iniciar, que fazia com que a Pulseira Assistiva não apresentasse funcionamento adequado, como por exemplo, não iniciando a execução com o motor de vibração conectado a GPIO 2.

Depois de aplicada a técnica de transferência térmica, a placa foi submetida ao processo de corrosão utilizando percloroeto de ferro. A Figura 20 apresenta a PCB finalizada após a corrosão.



Figura 20. PCB da Pulseira Assistiva finalizada.
Fonte: Autoria própria.

5.1.2 Montagem da Pulseira Assistiva

O Quadro 4 apresenta a descrição dos componentes utilizados para montar a Pulseira Assistiva, juntamente de seus respectivos custos:

Descrição	Quantidade	Valor Unitário	Valor Total
Barra de pinos fêmea	1	R\$ 8,00	R\$ 8,00
Bateria Lipo 3.7V 70mAh	1	R\$ 26,99	R\$ 26,99
Botão tátil 180: 12x12x5mm	1	R\$ 0,40	R\$ 0,40
Conector 2 vias	1	R\$ 0,20	R\$ 0,20
Micro chave 2 posições	1	R\$ 0,29	R\$ 0,29
Módulo ESP-01	1	R\$ 20,30	R\$ 20,30
Motor de Vibração SP1027	1	R\$ 3,75	R\$ 3,75
Pulseira de Silicone	1	R\$ 15,00	R\$ 15,00
Resistor 5,6Kohm	1	R\$ 0,10	R\$ 0,10
Total			R\$ 75,03

Quadro 4. Descrição dos componentes da Pulseira Assistiva e seus respectivos valores.

Fonte: Autoria própria.

Para otimização de espaço, foi feita a redução na altura da barra de pinos fêmea, no tamanho dos pinos da placa ESP01 e na eliminação do espaçador, resultando em cinco milímetros a menos de altura. A Figura 21 apresenta a comparação dos componentes: no lado (a) antes, e no lado (b) depois dos ajustes.

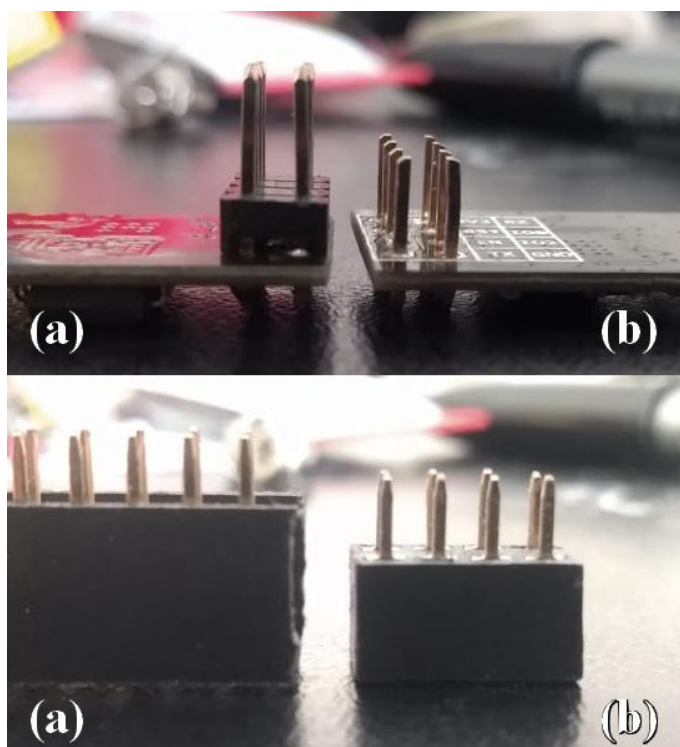


Figura 21. Comparação dos componentes depois das modificações.

Fonte: Autoria própria.

A Figura 22 apresenta a PCB com os componentes exceto o microcontrolador, para melhor visualização, já a Figura 23, apresenta a PCB com todos os componentes.

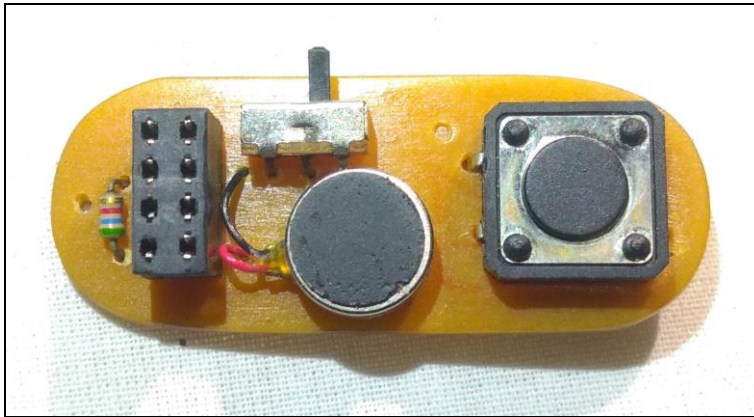


Figura 22. Visão superior da placa da Pulseira Assistiva sem microcontrolador.
Fonte: Autoria própria

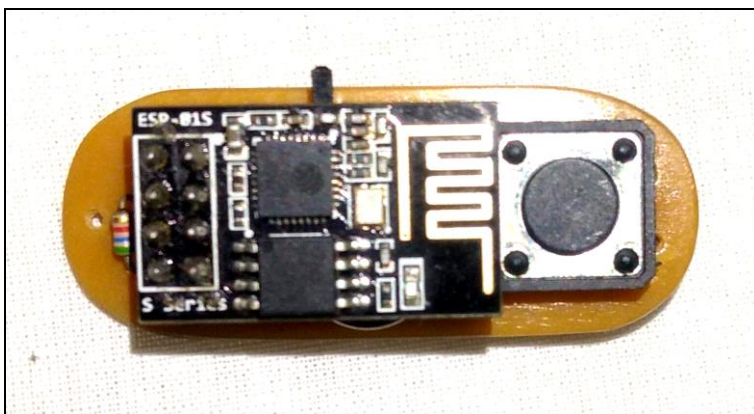


Figura 23. Visão superior da placa da Pulseira Assistiva com microcontrolador.
Fonte: Autoria própria.

5.2 Desenvolvimento e Confeção da PCB para o ESP32

Foi necessário confeccionar uma PCB para conectar a placa ESP32 e conectar a ela os sensores MQ2 e MQ6. A técnica usada foi a transferência térmica. A Figura 24 apresenta o projeto da PCB pronto para ser impresso.



Figura 24. Projeto da PCB do servidor
Fonte: Autoria própria.

A segunda versão foi feita devido a decisão de adicionar um resistor ao LED de notificações. O LED de notificação atua quando, (a) enquanto o microcontrolador está tentando se conectar a rede Wi-Fi, piscando, (b) a conexão for bem sucedida permanecendo aceso, e (c) recebe dados da Pulseira Assistiva, piscando pra indicar atividade.

A Figura 25 apresenta a PCB do servidor após a corrosão.

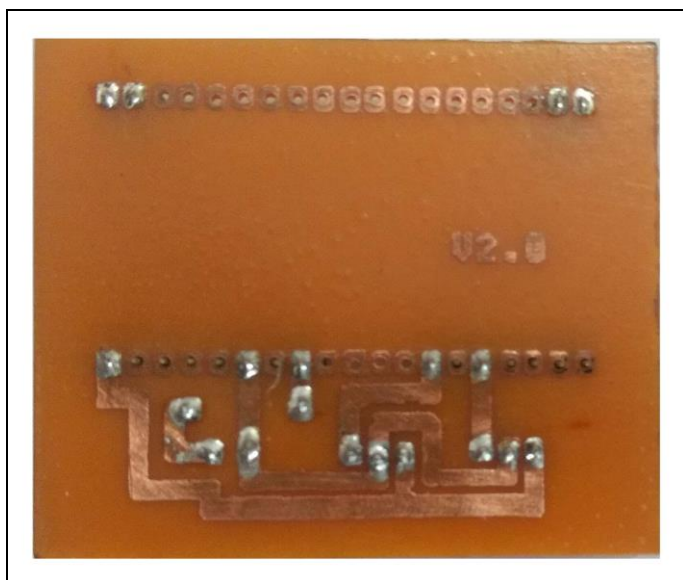


Figura 25. PCB do servidor após corrosão.
Fonte: Autoria própria.

O Quadro 5 apresenta a descrição dos componentes utilizados para montar o servidor, juntamente de seus respectivos custos:

Descrição	Quantidade	Valor Unitário	Valor Total
Barra de pinos fêmea	1	R\$ 8,00	R\$ 8,00
Conector 3 vias	2	R\$ 0,29	R\$ 0,58
LED	1	R\$ 0,99	R\$ 0,99
Módulo ESP32 NodeMCU-32S	1	R\$ 69,95	R\$ 69,95
Resistor 220 ohms	1	R\$ 0,10	R\$ 0,10
Total			R\$ 79,62

Quadro 5. Descrição dos componentes do servidor e seus respectivos valores.

Fonte: Autoria própria.

A Figura 26 apresenta a placa pós a instalação dos componentes com exceção da placa ESP32, e a Figura 27 apresenta a PCB já com a placa ESP32 colocada.

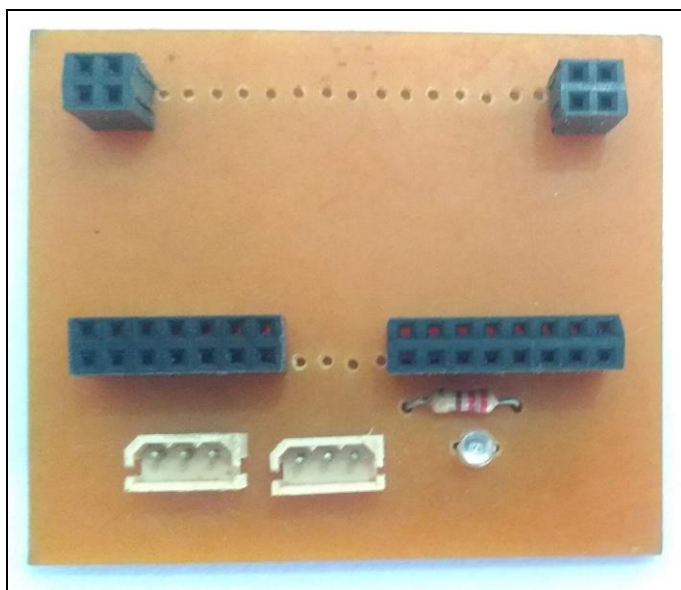


Figura 26. PCB do servidor sem o ESP32.

Fonte: Autoria própria.

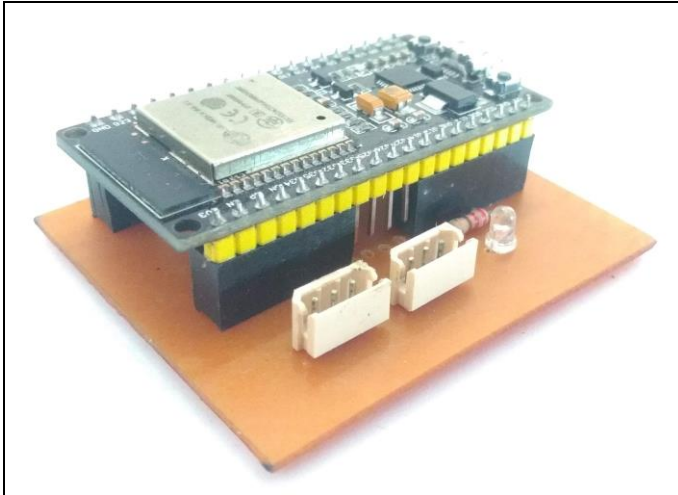


Figura 27. PCB do servidor com todos componentes.
Fonte: Autoria própria.

A barra de pinos fêmea foi instalada para permitir que o ESP32 pudesse ser removível, possibilitando troca rápida em caso de defeito, bem como facilitar a manutenção. Os dois conectores foram instalados para facilitar a instalação, manuseio e manutenção.

5.3 Base dos Sensores de Gás

A base dos sensores de gás foi desenvolvida para comportar os sensores em uma estrutura que tornasse possível sua fixação. Para isso, se produziu uma PCB usando a técnica transferência térmica. A Figura 28 apresenta o projeto da PCB da base dos sensores de gás pronto pra ser impresso.



Figura 28. Projeto da PCB da base dos sensores de gás.
Fonte: Autoria própria.

A segunda versão foi feita devido a erro de distanciamento das trilhas. O Quadro 6 apresenta a descrição dos componentes utilizados para montar a base dos sensores, juntamente de seus respectivos custos:

Descrição	Quantidade	Valor Unitário	Valor Total
Barra de pinos fêmea	1	R\$ 8,00	R\$ 8,00
Conector 3 vias	2	R\$ 0,29	R\$ 0,58
Módulo Sensor de Gás MQ2	1	R\$ 14,15	R\$ 14,15
Módulo Sensor de Gás MQ6	1	R\$ 13,00	R\$ 13,00
Total			R\$ 35,73

Quadro 6. Descrição dos componentes da base dos sensores e seus respectivos valores.
Fonte: Autoria própria.

A Figura 29 apresenta a PCB da base dos sensores de gás após corrosão.

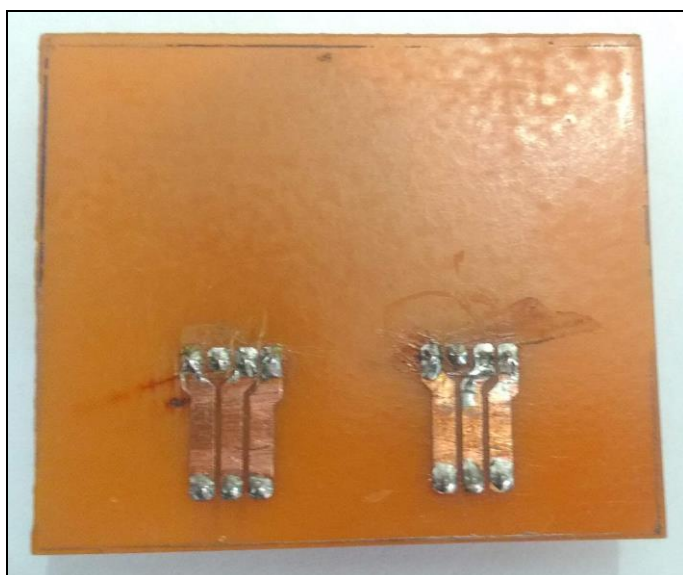


Figura 29. PCB da base dos sensores após o processo de corrosão.
Fonte: Autoria própria

A barra de pinos fêmea foi utilizada para que se pudesse remover os sensores facilmente em caso de defeito ou testes, facilitando a manutenção.

Os conectores de três vias foram usados para conectar a base dos sensores de gás ao servidor.

A Figura 30 apresenta a PCB depois de colocados os componentes com exceção dos sensores.

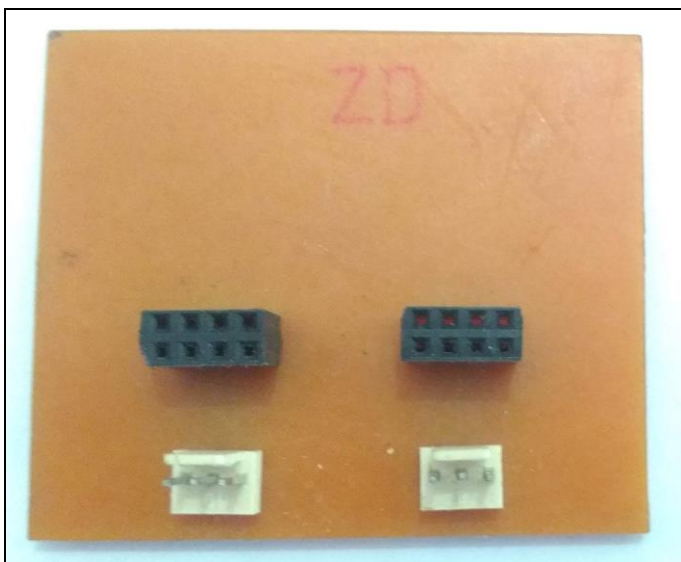


Figura 30. PCB da base dos sensores sem os sensores.
Fonte: Autoria própria.

A Figura 31 apresenta a PCB já com todos os componentes colocados.

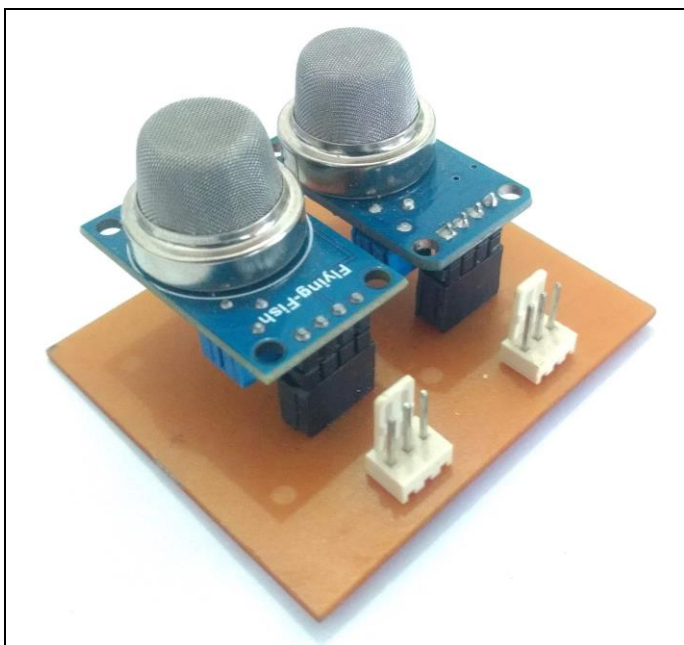


Figura 31. PCB da base dos sensores montada.
Fonte: Autoria própria.

5.4 Resultados Obtidos

A Pulseira Assistiva apresentou o funcionamento esperado. Quando conectada ao servidor, consegue receber o comando para vibrar e consegue enviar o alerta quando acionado o botão de pânico.

Os sensores utilizados neste trabalho são de baixa qualidade, sendo necessário a substituição por sensores mais confiáveis ao pensar em implementar o sistema em um ambiente. Contudo, apesar de ter sido possível utiliza-los para prototipagem, não atenderam as expectativas, além de apresentarem informações confusas em suas respectivas folhas de dados.

O servidor funcionou como esperado, a placa ESP32 supriu as expectativas mesmo deixando o gerenciamento das tarefas entre os núcleos por conta do algoritmo de seu *firmware*. Não houve perda de informações e nem perda de desempenho. A conexão com a rede Wi-Fi é quase instantânea, e o alcance foi suficiente para funcionamento em uma residência popular utilizando apenas um roteador.

6. DESENVOLVIMENTO DO APLICATIVO

Este capítulo apresenta o desenvolvimento do aplicativo ARPA3I que possibilita ao cuidador monitorar os dispositivos presentes na residência do idoso.

6.1 Tela de Abertura

A tela de abertura, Figura 32, é apresentada quando o usuário executa o comando para abrir o aplicativo. A mesma contém o logotipo que permanece estático até o software ser carregado e estar pronto pra ser utilizado pelo usuário.



Figura 32. Tela de abertura.
Fonte: Autoria própria.

6.2 Tela de Log in

A tela de log in, Figura 33, é a tela usada para autenticar o acesso do usuário, contém o campo “E-mail”, onde se digita o e-mail cadastrado, e o campo “Senha”, onde se digita a senha criada no cadastro. Além disso, contém o logotipo, botão “Entrar” para solicitar a autenticação e entrar no aplicativo, botão “Cadastre-se” para abrir a tela do formulário de cadastro, e a opção “Esqueci minha senha”, utilizada para recuperar o acesso do usuário que esqueceu a senha, que quando escolhida, abre o *pop-up* mostrado na Figura 34.

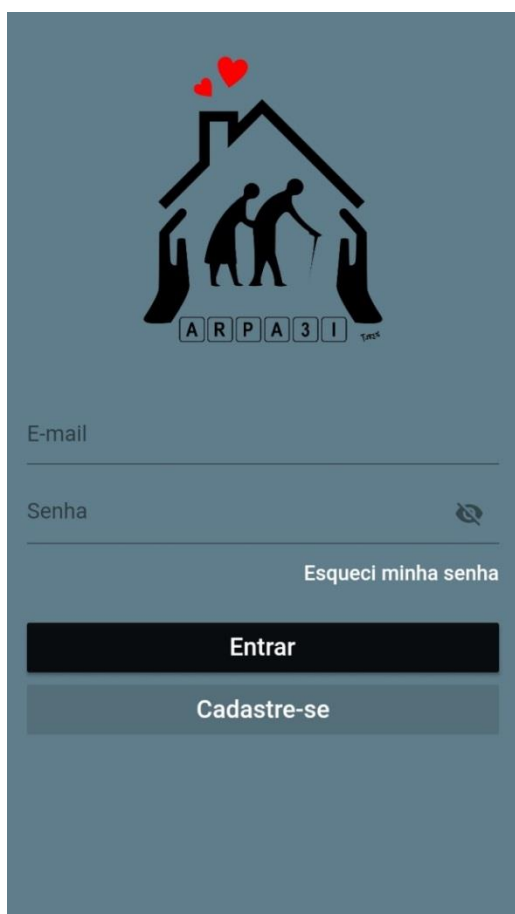


Figura 33. Tela de log in.
Fonte: Autoria própria.

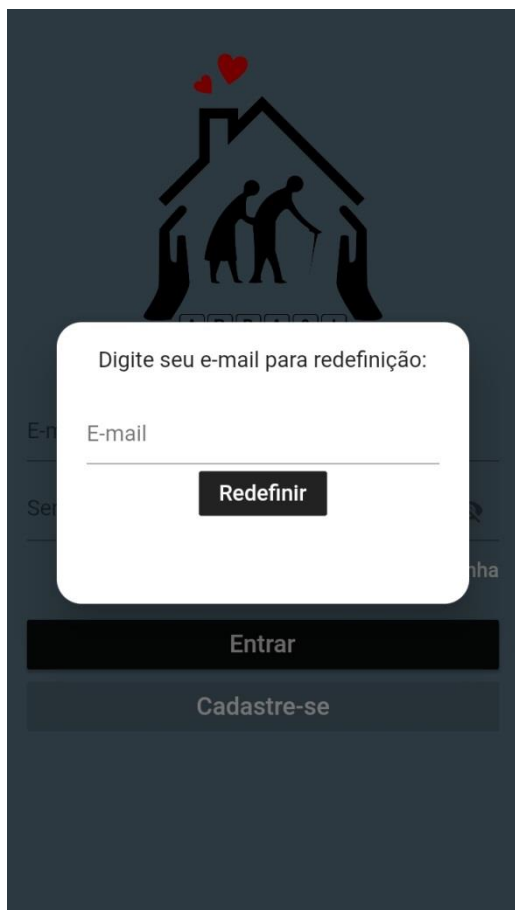


Figura 34. *Pop-up* para recuperação de senha.
Fonte: Autoria própria.

O *pop-up*, apresentado na Figura 34, contém um campo para digitar o e-mail cadastrado e o botão “Redefinir”. Ao digitar o e-mail e acionar o botão, uma mensagem é enviada ao e-mail inserido com um link de redefinição de senha, conforme exemplificado na Figura 35.

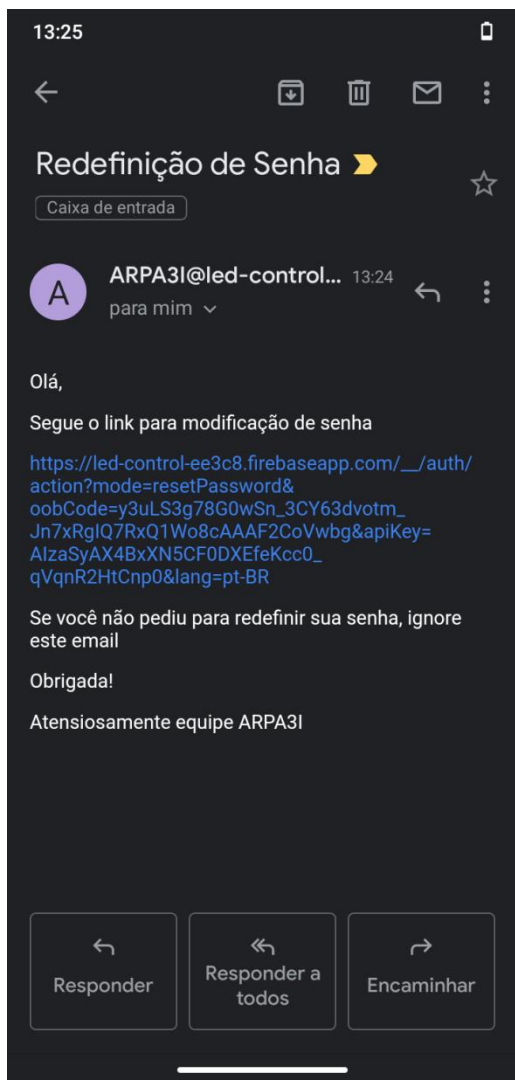


Figura 35. Mensagem recebida ao solicitar recuperação de senha.
Fonte: Autoria própria.

Ao clicar no link recebido, o usuário é encaminhado para uma tela para se criar uma nova senha como mostra a Figura 36.



Figura 36. Tela para criação de nova senha.
Fonte: Autoria própria.

6.3 Tela de cadastro

A tela de cadastro, Figura 37, surge quando o usuário, a partir da tela de log in seleciona a opção “Criar Conta”. Nessa tela contém os campos, “Nome Completo”, “E-mail”, “Senha”, “Telefone” e o botão “Criar Conta”. Após o usuário preencher todos os campos corretamente, surge uma *SnackBar* (barra de notificação inferior) com a mensagem “Usuário criado com sucesso!” como apresenta a Figura 38. A partir desse momento, todas as informações já estarão no banco de dados, o usuário é redirecionado para a tela de log in onde poderá autenticar seu acesso.

The image shows a mobile application interface for creating a new account. At the top, there is a black header bar with a white back arrow on the left and the text "Criar Conta" in white. Below the header, the background is a solid blue-grey color. There are four input fields, each with a light blue-grey background and a thin grey border. The labels for these fields are "Nome Completo", "E-mail", "Senha", and "Telefone", positioned to the left of each input area. To the right of the "Senha" input field, there is a small black icon of an eye with a diagonal slash, indicating a toggle for password visibility. Below the input fields, there is a black rectangular button with the white text "Criar Conta".

Figura 37. Tela de cadastro.
Fonte: Autoria própria.

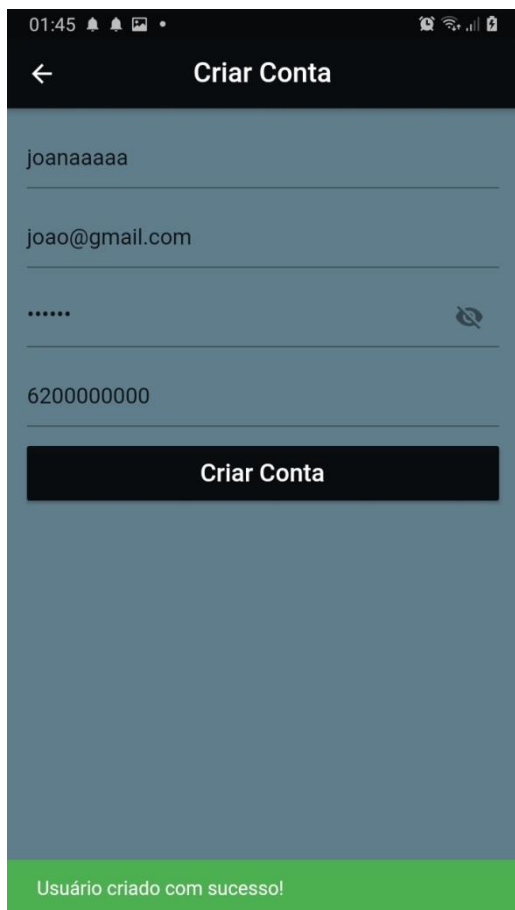


Figura 38. Tela de cadastro com Snackbar de confirmação.
Fonte: Autoria própria.

6.4 Tela Principal

Após realizar a autenticação do acesso, o usuário é redirecionado para a tela principal do aplicativo apresentada na Figura 39. No canto superior esquerdo há um botão para realizar o log out, após acionado, o usuário é direcionado para a tela de log in.

No canto superior esquerdo existe um menu onde se obtém informações e acesso as configurações, como mostra a Figura 40.

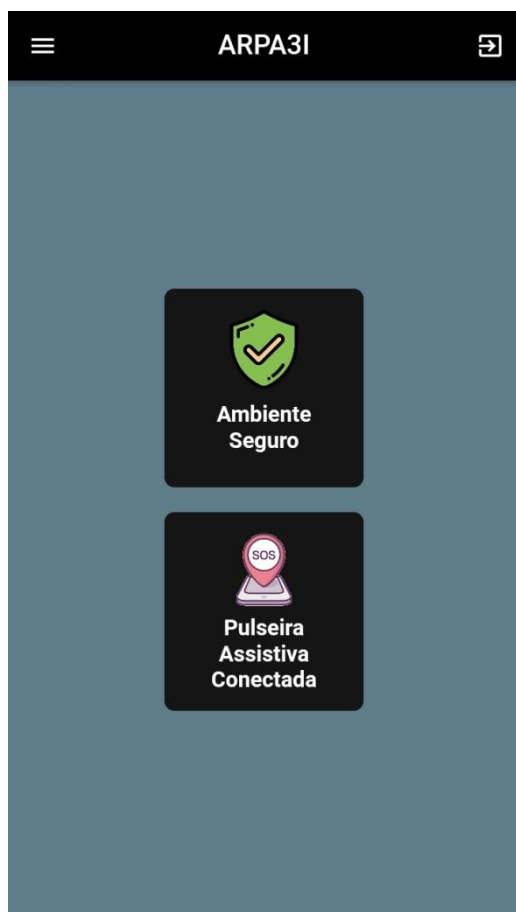


Figura 39. Tela principal do aplicativo.
Fonte: Autoria própria.

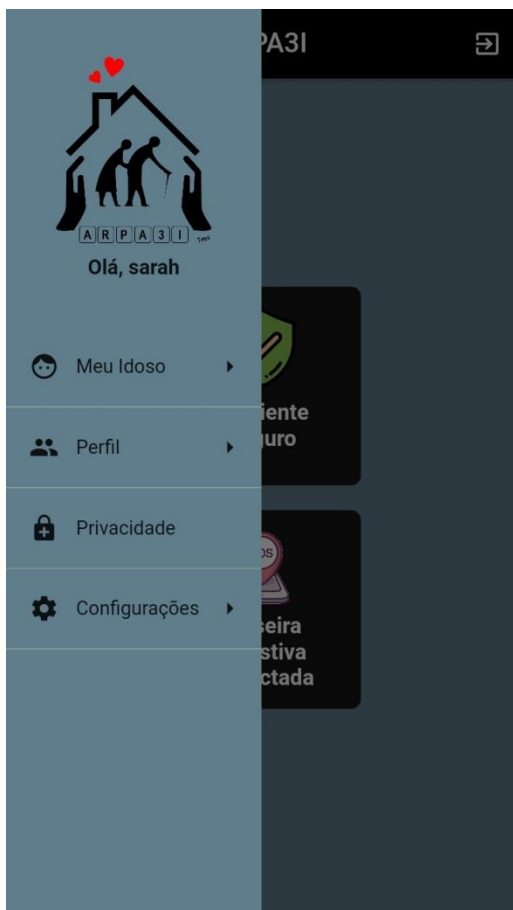


Figura 40. Tela principal após ser aberto o menu do aplicativo.
Fonte: Autoria própria.

Ao observar a Figura 39, percebe-se que na parte central da tela contém dois indicadores de estados alinhados verticalmente. O indicador superior indica o estado dos sensores de gás do ambiente, e o indicador inferior indica o estado da pulseira assistiva, e se o botão de pânico foi pressionado. Através dessas informações, o cuidador pode analisar a situação do ambiente em tempo real e tomar as decisões cabíveis para prestar assistência ao idoso.

6.4.1 *Status Ambiente Ideal*

A Figura 39 apresenta o status ambiente ideal, “Ambiente Seguro” indica que não há detecção de gás nem de fumaça no ambiente, e “Pulseira Assistiva Conectada” indica que o dispositivo está ligado e conectado, e não há solicitação de assistência vinda do idoso.

6.4.2 Status Fumaça Detectada

A Figura 41 apresenta o alerta “Fumaça”, pois os sensores de gás detectaram fumaça no ambiente, podendo ser um foco de incêndio. Nesse momento não há ações a fazer no aplicativo, pois após a saída da fumaça do ambiente, e não havendo gás no ar, o indicador apresenta “Ambiente Seguro” automaticamente.

A Figura 42 apresenta a notificação recebida pelo cuidador no momento em que ocorre a detecção.

Nesta análise, abstraiu-se o indicador da pulseira assistiva, pois as seções 6.4.4 e 6.4.5 são dedicados a ele, e seu estado não interfere no estado dos sensores de gás.

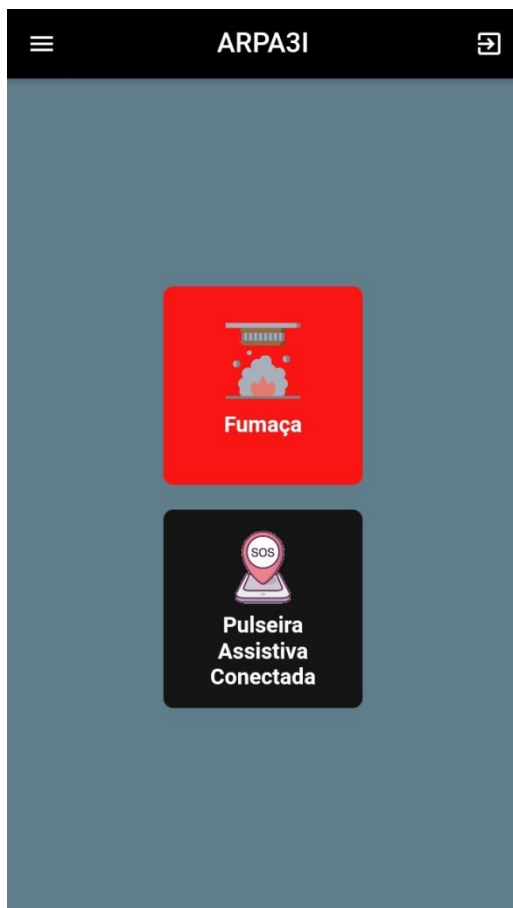


Figura 41. Aplicativo monitorando detecção de fumaça no ambiente.
Fonte: Autoria própria.

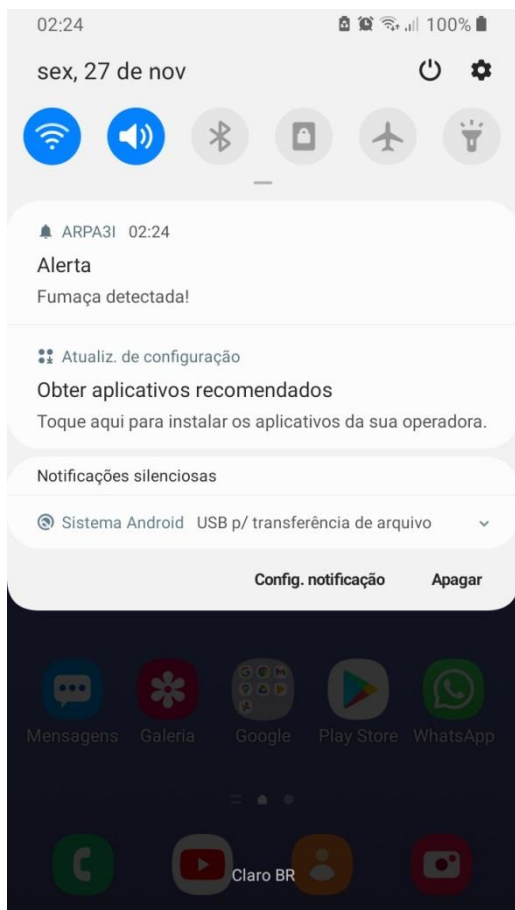


Figura 42. Notificação recebida após detecção de fumaça no ambiente.
Fonte: Autoria própria.

6.4.3 Status Gás Detectado

Análogo ao status “Fumaça Detectada”, a Figura 43 apresenta o status de alerta “Gás”, pois os sensores de gás detectaram GLP, propano, hidrogênio ou isobutano no ambiente, podendo ser um vazamento ou esquecimento do botão de controle de queimador do fogão na posição aberta, sem chama no queimador. Nesse momento não há ações a fazer no aplicativo, pois após a saída do gás do ambiente, e não havendo fumaça no ar, o indicador apresenta “Ambiente Seguro” automaticamente.

A Figura 44 apresenta a notificação recebida pelo cuidador no momento em que ocorre a detecção.



Figura 43. Aplicativo monitorando detecção de gás no ambiente.
Fonte: Autoria própria.

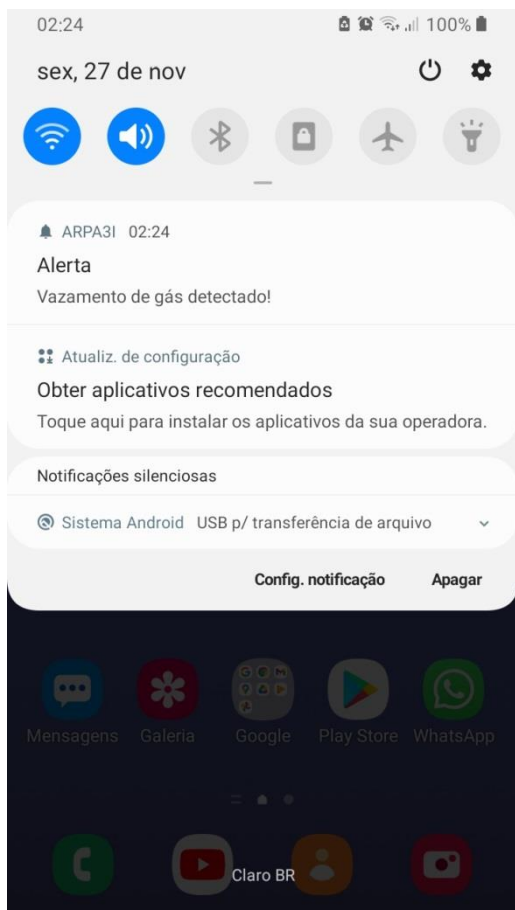


Figura 44. Notificação recebida após detecção de gás no ambiente.
Fonte: Autoria própria.

6.4.4 Status Pulseira Assistiva Desconectada

A Figura 45 apresenta o status “Pulseira Assistiva Desconectada”, isso acontece quando a pulseira assistiva está desligada ou desconectada do servidor por alguma razão. A Figura 46 apresenta a notificação que o cuidador recebe quando a pulseira se desconecta. Nesse momento não há ações a fazer no aplicativo, pois ele recebe a informação do servidor quando é conectada novamente, e atualiza o indicador para “Pulseira Assistiva Conectada”.

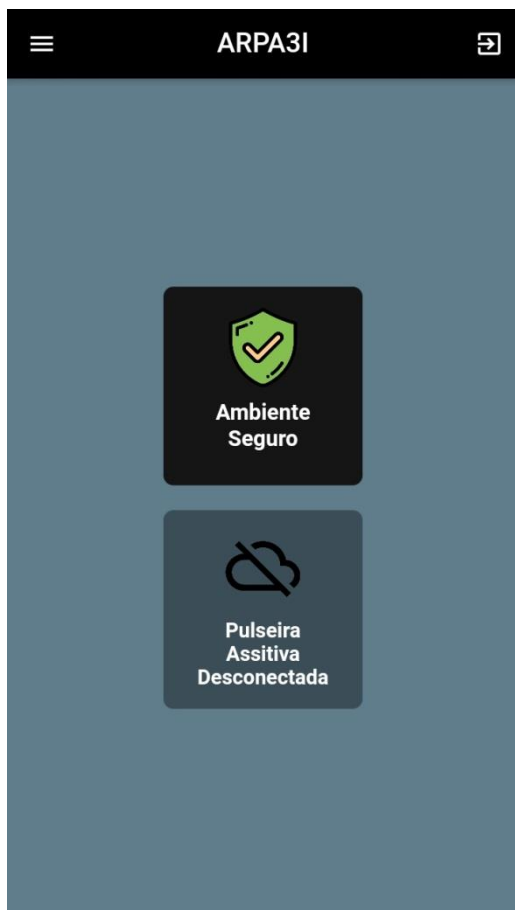


Figura 45. Aplicativo reconhecendo pulseira assistiva desconectada.
Fonte: Autoria própria.

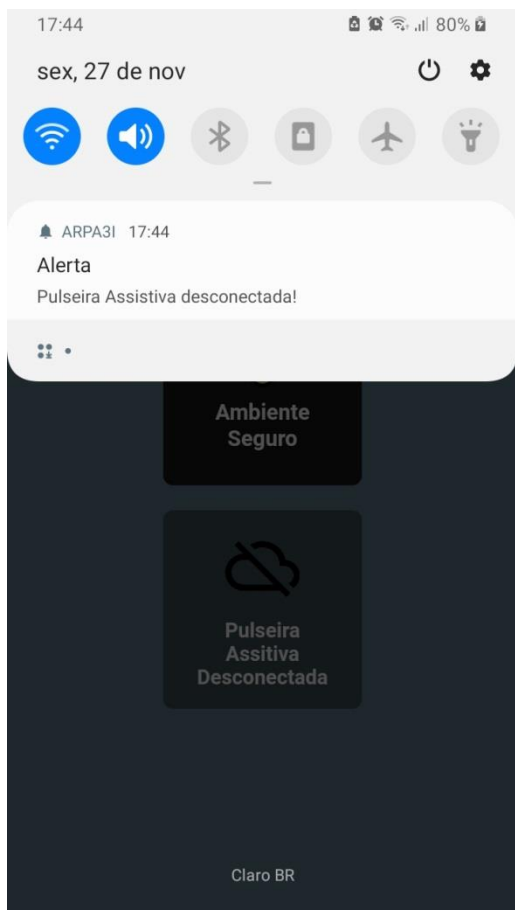


Figura 46. Notificação recebida após o reconhecimento da desconexão da pulseira assistiva.
Fonte: Autoria própria.

6.4.5 Status Botão Pressionado

A Figura 47 apresenta o status “Botão Pressionado”, que ocorre quando o portador da pulseira assistiva aperta o botão de pânico para solicitar socorro ou assistência emergencial. A Figura 48 apresenta a notificação recebida pelo cuidador no momento em que o botão de pânico é acionado.

Neste momento o indicador “Botão Pressionado” altera seu comportamento para botão, sendo necessário acioná-lo e escolher a opção “Ok, estou ciente” no *pop-up* que surge, para que o estado atualize para “Pulseira Assistiva Conectada” ou “Pulseira Assistiva Desconectada”. A Figura 49 apresenta o *pop-up* de quando o botão “Botão Pressionado” é acionado.



Figura 47. Aplicativo reconhecendo o acionamento do botão de pânico.
Fonte: Autoria própria.

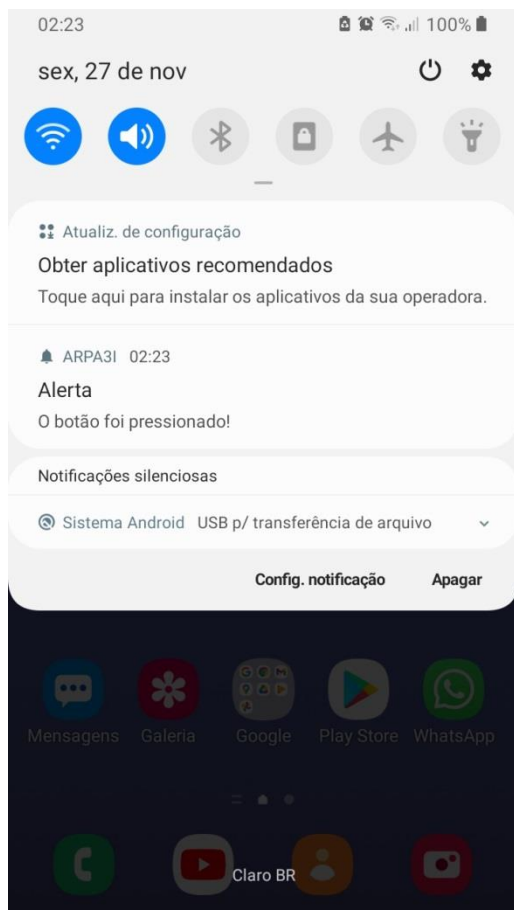


Figura 48. Notificação recebida após o reconhecimento do acionamento do botão de pânico.
Fonte: Autoria própria.

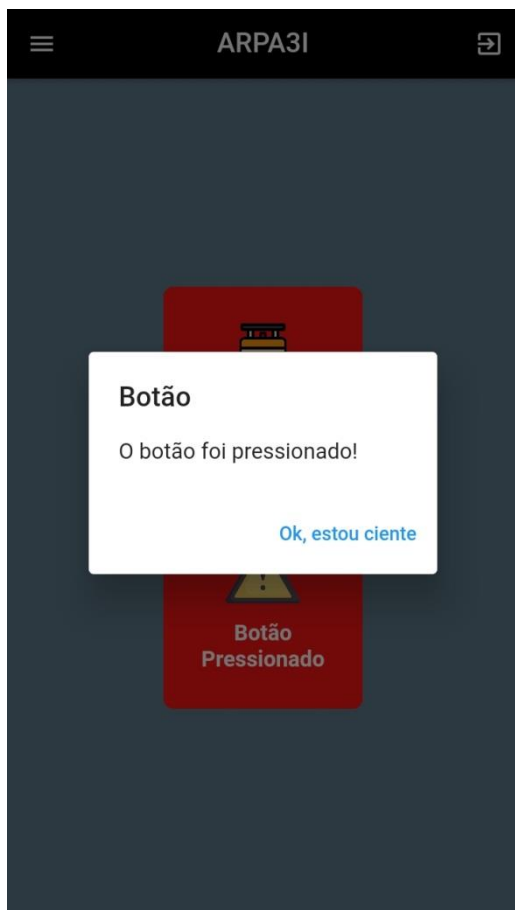


Figura 49. *Pop-up* para confirmar a ciência do alarme recebido.
Fonte: Autoria própria.

6.5 Resultados Obtidos

O aplicativo apresentou funcionamento e desempenho satisfatório. Todas as funções da tela de log in funcionaram perfeitamente, exceto ferramenta de recuperar senha, pois as vezes demora um pouco para enviar o e-mail com o link para redefinição de senha.

A tela inicial apresentou os estados dos dispositivos em tempo real. Foi simulado o acionamento do botão de pânico da pulseira assistiva, vazamento de gás de cozinha usando um isqueiro e um foco de incêndio usando fumaça, em todos os cenários o aplicativo atualizou o estado dos dispositivos no tempo programado, sendo um segundo além de receber as respectivas notificações. A opção de fazer log out presente na mesma tela também funcionou como deveria, possibilitando alternar usuários facilmente.

O capítulo 7 apresenta as considerações finais sobre o trabalho, e também as sugestões para trabalhos futuros.

7. CONSIDERAÇÕES FINAIS

A automação assistiva é um ramo da automação que tende a crescer, resultando em melhor qualidade de vida para os idosos, quanto facilidade de monitoramento dos mesmos pelos cuidadores. Apesar do público alvo ser os idosos, pessoas com necessidades especiais permanentes ou temporárias também podem se beneficiar, com o desenvolvimento de funcionalidades para necessidades específicas.

A pulseira assistiva se conecta instantaneamente ao ESP32 via rede Wi-Fi, ficando o portador assistido por toda a cobertura da rede, o dispositivo é leve e confortável. Os sensores de gás podem ser facilmente instalados por meio da base desenvolvida. O ESP32 pode ficar em qualquer local dentro da residência coberta pela rede.

O aplicativo multiplataforma ARPA3I, desenvolvido utilizando a linguagem de programação Dart e o *framework Flutter*, proporciona flexibilidade ao usuário ao utilizar os sistemas operacionais Android e iOS. Também proporciona agilidade e economia ao desenvolvimento, pois não é preciso desenvolver um aplicativo para cada sistema operacional. Além disso, o aplicativo será compatível com o sistema operacional que o Google está desenvolvendo atualmente, o Google Fuchsia.

A plataforma para criação de aplicativos móveis *Firebase*, possui serviços de banco de dados em nuvem. Porém, com o avanço do projeto o plano gratuito se torna limitado.

Este trabalho mostra que, com soluções relativamente simples, é possível prover mais segurança para os idosos, podendo preservar sua integridade física em diversas situações, até mesmo salvar vidas.

7.1 Trabalhos Futuros

Como trabalhos futuros, sugere-se a melhoria da Pulseira Assistiva, incrementação de funções ao sistema e o desenvolvimento de outros dispositivos para compor o sistema.

A Pulseira Assistiva pode ser equipada com acelerômetro para agregar a função de detector de queda, sensores de batimentos cardíacos, GPS, microfone, entre outros. Também pode se implementar uma porta micro USB ou USB-C para o carregamento da bateria sem precisar retirá-la do dispositivo.

O servidor tem capacidade de desempenhar mais funções, como por exemplo, gerenciar automação de aparelhos na residência, como por exemplo, ar condicionado, persianas, toldos, controle de acesso, entre outros.

Outros dispositivos podem ser conectados ao servidor e serem controlados e ou monitorados pelo cuidador, partindo do princípio que a parte de monitoramento e configuração seja transparente para o idoso, cabendo a ele apenas desfrutar dos benefícios, sem precisar aprender a utilizar uma nova tecnologia.

Como sugestão de dispositivos têm se, (a) um dosador de comprimidos automático para entregar ao idoso os medicamentos a serem tomados na dosagem e na hora certa e (b) um elevador de degrau para o idoso se locomover com mais facilidade entre cômodos que possuam degraus, em sua residência.

REFERÊNCIAS

AAL ASSOCIATION. **Ageing well in a digital world**. Disponível em: <<http://www.aal-europe.eu/about/>>. Acesso em: 01 nov. 2019.

ALAM, M. R.; REAZ, M. B. I.; ALI, M. A. M. **A review of smart homes: past, present, and future**. *IEEE Transactions on Systems, Man and Cybernetics Part C: Applications and Reviews*, New York, v. 42, n. 6, p. 1190-1203, 2012.

ALMEIDA, Rodrigo Maximiano Antunes de; MORAES, Carlos Henrique Valério de; SERAPHIM, Thatyana de Faria Piola. **Programação de Sistemas Embarcados: desenvolvendo software para microcontroladores em linguagem C**. Rio de Janeiro: Elsevier, 2016.

BLACKMAN, S. et al. **Ambient assisted living technologies for aging well: a scoping review**. *Journal of Intelligent Systems*, London, v. 25, n. 1, p. 55-59, 2015.

CAMARGO, Valter Luis Arlindo de. **Elementos de automação**. 1 ed. São Paulo: Érica, 2014.

CORAZZA, Paulo Victor. **Um aplicativo multiplataforma desenvolvido com Flutter e NoSQL para o cálculo da probabilidade de apendicite**. 2018. 64 f. Monografia (Especialização) - Curso de Ciência da Computação, Instituto de Informática, Universidade Federal do Rio Grande do Sul, Porto Alegre, 2018.

COULOURIS, George et al. **Sistemas distribuídos: conceitos e projetos**. 5. ed. Porto Alegre: Bookman, 2013.

ESTEVAM JUNIOR, Sergio Luiz; FARINELLI, Felipe Adalberto. **Domótica: Automação Residencial e Casas Inteligentes com Arduino e ESP8266**. 1 ed. São Paulo: Érica, 2019.

GONÇALVES, André Luz. **Desenvolvimento de um aplicativo Android utilizando banco de dados não-relacional para organização e controle de presença de um time de futebol**. 2016. 61 f. TCC (Graduação) - Curso de Ciência da Computação, Instituto de Informática, Universidade Federal do Rio Grande do Sul, Porto Alegre, 2016.

GRGURIC, A. (2012) “**ICT towards elderly independent living**”. **Research and Development Center Ericsson Nikola Tesla**, Krapinska 45, Zagreb, Croatia.

IBGE. **Tábua completa de mortalidade para o Brasil**. 2017. Disponível em: <ftp://ftp.ibge.gov.br/Tabuas_Completas_de_Mortalidade/Tabuas_Completas_de_Mortalidade_2017>. Acesso em: 14 set. 2019.

NAÇÕES UNIDAS DO BRASIL (Brasil). **Liberdade de escolha sobre ter filhos ou não afeta desenvolvimento dos países, diz relatório do UNFPA**. 2018. Disponível em: <<https://nacoesunidas.org/liberdade-de-escolha-sobre-ter-filhos-ou-nao-afeta-desenvolvimento-dos-paises-diz-relatorio-do-unfpa>>. Acesso em: 14 set. 2019.

OHTA, Ricardo Luís et al. PASSADO, PRESENTE E FUTURO: *WEARABLES* E INTERNET DAS COISAS. **Revista da Sociedade Brasileira de Computação**. Porto Alegre, v. 1, n. 29, p.36-44, abr. 2015. Disponível em: <https://www.sbc.org.br/images/flippingbook/computacaobrasil/computa_29_pdf/comp_brasil_2015_4.pdf>. Acesso em: 21 out. 2019.

SANTOS, Jean Willian; LARA JUNIOR, Renato Capelin de. **Sistema de Automação Residencial de Baixo Custo Controlado pelo Microcontrolador ESP32 e Monitorado Via Smartphone**. 2019. 46 f. TCC (Graduação) - Curso de Tecnólogo em Automação Industrial, Universidade Tecnológica Federal do Paraná, Ponta Grossa, 2019. Disponível em: <<http://repositorio.roca.utfpr.edu.br/jspui/handle/1/12133>>. Acesso em: 14 out. 2019.

SOUZA, Sandro Ferreira de. **A contribuição da automação residencial na solução de problemas de acessibilidade no cotidiano do idoso**. 2015. 134 f. Dissertação (Mestrado) - Curso de Arquitetura e Urbanismo, Universidade Federal de Viçosa, Viçosa, 2015. Disponível em: <<http://www.locus.ufv.br/handle/123456789/7654>>. Acesso em: 13 out. 2019.

TORRES, Guilherme Gerzson. **Tecnologia assistiva para detecção de quedas: desenvolvimento de sensor vestível integrado ao sistema de casa inteligente**. 2018. 71 f. Dissertação (Mestrado) - Curso de Engenharia Elétrica, Universidade Federal do Rio Grande do Sul, Porto Alegre, 2018. Disponível em: <<https://lume.ufrgs.br/handle/10183/180824>>. Acesso em: 14 out. 2019.

VARGAS, Maurício Fontana de. **Proposta de um Sistema de Assistência Personalizada para Ambientes Inteligentes**. 2016. 117 f. Dissertação (Mestrado) - Curso de Engenharia Elétrica, Universidade Federal do Rio Grande do Sul, Porto Alegre, 2016. Disponível em: <<https://lume.ufrgs.br/handle/10183/142712>>. Acesso em: 14 out. 2019.

ZION MARKET RESEARCH. **Mercado Doméstico Inteligente (Cozinha Inteligente, Segurança e Controle de Acesso, Controle de Iluminação, Assistência Médica Domiciliar, Controle HVAC e outros): Perspectiva Global da Indústria, Análise e Previsão Abrangentes, 2016-2022**. 2017. Disponível em: <<https://www.zionmarketresearch.com/report/smart-home-market>>. Acesso em: 21 set. 2019.

APÊNDICE A – CÓDIGO TESTE DOS SENSORES DE GÁS

```
1  int PinoMQ2 = 32;
2  int PinoMQ6 = 34;
3  unsigned long tempoAnterior = 0;
4  unsigned long intervalo = 500;
5  typedef bool Function();
6  void leituraMQx(int SMQ2, int SMQ6){
7  if (Temporizador(&tempoAnterior, intervalo)){
8  int leituraMQ2 = leitura(SMQ2, 50);
9  int leituraMQ6 = leitura(SMQ6, 50);
10 Serial.print("MQ2: ");
11 Serial.print(leituraMQ2);
12 Serial.print("%");
13 Serial.print(" MQ6: ");
14 Serial.print(leituraMQ6);
15 Serial.println("%");
16 delay(50);
17 }
18 }
19 bool Temporizador(unsigned long* tempoAnterior, unsigned long intervalo){
20 unsigned long tempoCorrido = millis();
21 if (tempoCorrido - *tempoAnterior > intervalo){
22 *tempoAnterior = tempoCorrido;
23 return true;
24 }
25 else{
26 return false;
27 }
28 }
29 int leitura(int pino, int nvezes){
30 int valor = 0;
31 for (int i = 0; i < nvezes; i++){
32 valor = valor + (analogRead(pino));
33 }
34 return valor = ((valor / nvezes) * 100) / 4095;
35 }
36 void setup(){
37 Serial.begin(115200);
38 pinMode(PinoMQ2, INPUT);
39 pinMode(PinoMQ6, INPUT);
40 }
41 void loop(){
42 leituraMQx(PinoMQ2, PinoMQ6);
43 }
```

APÊNDICE B – CÓDIGO DO SERVIDOR

```
1  #include <WiFi.h>
2  #include <FirebaseESP32.h>
3  #define FIREBASE_HOST "https://projeto.firebaseio.com/"
4  #define FIREBASE_AUTH "Chave de autenticação Firebase"
5  #define WIFI_SSID "Nome da rede Wi-Fi"
6  #define WIFI_PASSWORD "Senha da rede Wi-Fi"
7  #define SERVER_PORT 5000
8  #define CSERVER_PORT 6000
9  WiFiServer server(SERVER_PORT);
10 FirebaseData firebaseData;
11 FirebaseJson json;
12 IPAddress ip(192, 168, 1, 26);
13 IPAddress gateway(192, 168, 1, 1);
14 IPAddress subnet(255, 255, 255, 0);
15 IPAddress primaryDNS(192, 168, 1, 1);
16 IPAddress secondaryDNS(8, 8, 8, 8);
17 IPAddress clientAddress(192, 168, 1, 27);
18 int Valor = 0;
19 int N = 0;
20 #define Notif 14
21 enum Protocol{
22   PIN,
23   VALUE,
24   NVEZES,
25   BUFFER_SIZE
26 };
27 unsigned long tempoAnterior1 = 0;
28 unsigned long tempoAnterior2 = 0;
29 unsigned long intervalo1 = 10000;
30 unsigned long intervalo2 = 1000;
31 typedef bool Function();
32 int PinoMQ2 = 32;
33 int PinoMQ6 = 34;
34 void setup(){
35   Serial.begin(115200);
36   pinMode(PinoMQ2, OUTPUT);
37   pinMode(PinoMQ6, OUTPUT);
38   pinMode(Notif, OUTPUT);
39   WiFi.mode(WIFI_STA);
40   WiFi.config(ip, gateway, subnet, primaryDNS, secondaryDNS);
41   WiFi.begin(WIFI_SSID, WIFI_PASSWORD);
42   while (WiFi.status() != WL_CONNECTED){
43     delay(500);
44     Serial.print(".");
45     digitalWrite(Notif, HIGH);
46     delay(50);
47     digitalWrite(Notif, LOW);
```

```

48 delay(50);
49 }
50 digitalWrite(Notif, HIGH);
51 Serial.print("WiFi connected with IP: ");
52 Serial.println(WiFi.localIP());
53 Firebase.begin(FIREBASE_HOST, FIREBASE_AUTH);
54 Firebase.reconnectWiFi(true);
55 Firebase.setReadTimeout(firebaseData, 1000 * 60);
56 Firebase.setwriteSizeLimit(firebaseData, "tiny");
57 server.begin();
58 }
59 void loop(){
60 VerificaConexao();
61 delay(50);
62 LeituraMQx(PinoMQ2, PinoMQ6);
63 delay(50);
64 Leitura();
65 delay(50);
66 }
67 void Leitura(){
68 WiFiClient client = server.available();
69 if (client){
70 if (client.available()){
71 uint8_t buffer[Protocol::BUFFER_SIZE];
72 int len = client.read(buffer, Protocol::BUFFER_SIZE);
73 int pinNumber = buffer[Protocol::PIN];
74 Valor = buffer[Protocol::VALUE];
75 int XVezes = buffer[Protocol::NVEZES];
76 pinMode(pinNumber, OUTPUT);
77 digitalWrite(pinNumber, !Valor);
78 delay(150);
79 digitalWrite(pinNumber, Valor);
80 if (XVezes > 0){
81 Firebase.updateNode(firebaseData, "/ARPA3I/Variaveis", json);
82 json.set("/Panico", Valor);
83 }
84 client.stop();
85 Serial.println(Valor);
86 Serial.println(XVezes);
87 }
88 client.stop();
89 }
90 }
91 void Envia()
92 {
93 WiFiClient client;
94 if (client.connect(clientAddress, CSERVER_PORT)){
95 Serial.println("Conexão ao cliente bem sucedida!");
96 uint8_t buffer[Protocol::BUFFER_SIZE];
97 buffer[Protocol::PIN] = 3;

```

```

98  buffer[Protocol::VALUE] = 1;
99  buffer[Protocol::NVEZES] = N;
100 for (int i = 0; i < 3; i++){
101  client.write(buffer, Protocol::BUFFER_SIZE);
102 }
103 client.flush();
104 client.stop();
105 N++;
106 }
107 }
108 void VerificaConexao(){
109  if (Temporizador(&tempoAnterior1, intervalo1)){
110   WiFiClient client;
111   Firebase.getJSON(firebaseData, "/ARPA3I/Variaveis/Panico");
112   int conferente = firebaseData.intData();
113   Serial.println(conferente);
114   if (conferente == 0 || conferente == 2){
115    if (client.connect(clientAddress, CSERVER_PORT)){
116     if (conferente == 2){
117      Serial.println("Conexão ao cliente bem sucedida!");
118      Firebase.updateNode(firebaseData, "/ARPA3I/Variaveis", json);
119      json.set("/Panico", 0);
120      client.flush();
121      client.stop();
122      return;
123     }
124     else{
125      Serial.println("Conexão ao cliente bem sucedida!");
126      client.flush();
127      client.stop();
128      return;
129     }
130    }
131    else{
132     Serial.println("Azedou o pé de frango no cliente!");
133     Firebase.updateNode(firebaseData, "/ARPA3I/Variaveis", json);
134     json.set("/Panico", 2);
135     client.flush();
136     client.stop();
137     return;
138    }
139   }
140  }
141 }
142 bool Temporizador(unsigned long* tempoAnterior, unsigned long intervalo){
143  unsigned long tempoCorrido = millis();
144  if (tempoCorrido - *tempoAnterior > intervalo){
145   *tempoAnterior = tempoCorrido;
146   return true;
147  }

```

```

148 else{
149 return false;
150 }
151 }
152 void LeituraMQx(int SMQ2, int SMQ6){
153 if (Temporizador(&tempoAnterior2, intervalo2)){
154 int leituraMQ2 = LeituraAuxMQx(SMQ2, 20);
155 int leituraMQ6 = LeituraAuxMQx(SMQ6, 5);
156 if (leituraMQ2 < 5 && leituraMQ6 < 5){
157 Firebase.updateNode(firebaseData, "/ARPA3I/Variaveis", json);
158 json.set("/Sensores", 0);
159 }
160 if (leituraMQ2 > 5 && leituraMQ6 < 5){
161 Envia();
162 Firebase.updateNode(firebaseData, "/ARPA3I/Variaveis", json);
163 json.set("/Sensores", 1); // ambiente com fumaça
164 }
165 if (leituraMQ6 > 0 && leituraMQ6 > 0){
166 Envia();
167 Firebase.updateNode(firebaseData, "/ARPA3I/Variaveis", json);
168 json.set("/Sensores", 2); // vazamento de gás
169 }
170 Serial.print("MQ2: ");
171 Serial.print(leituraMQ2);
172 Serial.print("%");
173 Serial.print(" MQ6: ");
174 Serial.print(leituraMQ6);
175 Serial.println("%");
176 }
177 }
178 int LeituraAuxMQx(int pino, int nvezes){
179 int valor = 0;
180 for (int i = 0; i < nvezes; i++){
181 valor = valor + (analogRead(pino));
182 }
183 return valor = ((valor / nvezes) * 100) / 4095;
184 }

```

APÊNDICE C – CÓDIGO DA PULSEIRA ASSISTIVA

```
1  #include <ESP8266WiFi.h>
2  #define WIFI_SSID "Nome da rede Wi-Fi"
3  #define WIFI_PASSWORD "Senha da rede Wi-Fi"
4  #define SERVER_PORT 5000
5  #define CSERVER_PORT 6000
6  WiFiServer server(CSERVER_PORT);
7  IPAddress ip(192, 168, 1, 27);
8  IPAddress gateway(192, 168, 1, 1);
9  IPAddress subnet(255, 255, 255, 0);
10 IPAddress serverAddress(192, 168, 1, 26);
11 enum Protocol{
12 PIN,
13 VALUE,
14 NVEZES,
15 BUFFER_SIZE
16 };
17 #define PinoBotao 0
18 int PinoMotor = 3;
19 int Valor = 0;
20 int Estado = 0;
21 int N = 0;
22 void setup(){
23   pinMode(PinoBotao, INPUT_PULLUP);
24   pinMode(PinoMotor, OUTPUT);
25   digitalWrite(PinoMotor, LOW);
26   delay(1000);
27   Serial.begin(9600);
28   WiFi.config(ip, gateway, subnet);
29   WiFi.begin(WIFI_SSID, WIFI_PASSWORD);
30   while (WiFi.status() != WL_CONNECTED){
31     delay(500);
32     Serial.print(".");
33   }
34   Serial.print("WiFi connected with IP: ");
35   Serial.println(WiFi.localIP());
36   server.begin();
37 }
38 void loop(){
39   Envia();
40   delay(50);
41   Leitura();
42   delay(50);
43 }
44 void Envia(){
45   WiFiClient client;
46   int value = digitalRead(PinoBotao);
47   if (value == 1 && Estado == 0){
```

```

48 if (!client.connect(serverAddress, SERVER_PORT)){
49   delay(1000);
50   Serial.println("Connection to server failed");
51   return;
52 }
53 Serial.println("Connection to server successful!");
54 uint8_t buffer[Protocol::BUFFER_SIZE];
55 buffer[Protocol::PIN] = 14;
56 buffer[Protocol::VALUE] = value;
57 buffer[Protocol::NVEZES] = N;
58 for (int i = 0; i < 3; i++){
59   client.write(buffer, Protocol::BUFFER_SIZE);
60 }
61 client.flush();
62 client.stop();
63 Estado = 1;
64 N++;
65 }
66 if (value == 0 && Estado == 1){
67   Estado = 0;
68 }
69 }
70 void Leitura(){
71   WiFiClient client = server.available();
72   if (client){
73     if (client.available()){
74       Serial.println("Cliente Conectado");
75       uint8_t buffer[Protocol::BUFFER_SIZE];
76       int len = client.read(buffer, Protocol::BUFFER_SIZE);
77       PinoMotor = buffer[Protocol::PIN];
78       pinMode(PinoMotor, OUTPUT);
79       Valor = buffer[Protocol::VALUE];
80       int XVezes = buffer[Protocol::NVEZES];
81       Serial.println("Recebido ");
82       Serial.println(Valor);
83       if (XVezes > 0){
84         for (int i = 0; i < 3; i++){
85           digitalWrite(PinoMotor, Valor);
86           delay(300);
87           digitalWrite(PinoMotor, !Valor);
88           delay(300);
89         }
90       }
91     }
92     client.stop();
93   }
94 }

```