Centro Federal de Educaçào Tecnológica de Minas Gerais
Campus Divinópolis
Graduação em Engenharia Mecatrônica

Lanna Gontijo Ferreira

InCasa - Projeto e protótipo de sistema de automação multiplataforma com funções de comando via reconhecimento de voz

Divinópolis
2021

Lanna Gontijo Ferreira

InCasa - Projeto e protótipo de sistema de automação multiplataforma com funções de comando via reconhecimento de voz

Monografia de Trabalho de Conclusão de Curso apresentada ao Colegiado de Graduação em Engenharia Mecatrônica como parte dos requisitos exigidos para a obtenção do título de Engenheira Mecatrônica.
Eixo de Formação: Computação e Automação Residencial.

Orientador: Prof. Me. Tiago Alves De Oliveira
Coorientador: Me. Álan Crístoffer e Sousa

Divinópolis
2021

Centro Federal de Educação Tecnológica de Minas Gerais
CEFET-MG / Divinópolis
Curso de Engenharia Mecatrônica

Monografia intitulada “InCasa - Projeto e protótipo de sistema de automação multiplataforma com funções de comando via reconhecimento de voz”, de autoria da graduanda Lanna Gontijo Ferreira, aprovada pela banca examinadora constituída pelos seguintes professores:

Prof. Me. Tiago Alves De Oliveira

Prof. Me. Amanda Fernandes Vilaça Martins

Prof. Me. Michel Pires da Silva

Coordenador do Curso de Engenharia Mecatrônica

Prof. Dr. Marlon Antônio Pinheiro

Divinópolis

Abril de 2021

Aos meus pais.

Agradecimentos

Agradeço,

- aos meus pais, Elaine e Agnaldo, pela paciência e pelo apoio incondicional;
- ao Prof. Tiago Alves pela orientação, pela disposição e pela confiança;
- ao Álan Crístoffer, pelas conversas e lições durante o curso, ainda mais - no final;
- ao Guilherme Gazzinelli, pelos suporte e indicações necessárias;
- ao Guilherme Oliveira, meu primo, mais um guia pontual;
- às companheiras de engenharia Cláudya Martins e Daniele Araújo, pelos momentos compartilhados;
- às companheiras de Nova Serrana, Bruna Araújo e Larissa Lopes, amizades de longa data;
- à Bárbara Fonseca e Julie Correia, pela boa convivência em Divinópolis;
- ao meu irmão André e familiares próximos, pelo apoio;
- aos professores e funcionários do CEFET-MG, pelo empenho no *campus* V;
- e a todos aqueles que de alguma forma colaboraram para que eu chegasse até aqui.

De tudo, ficaram três coisas: a certeza de que ele estava sempre começando, a certeza de que era preciso continuar e a certeza de que seria interrompido antes de terminar. Fazer da interrupção um caminho novo. Fazer da queda um passo de dança, do medo uma escada, do sono uma ponte, da procura um encontro.

Fernando Sabino

Resumo

Baseado no conceito de Internet das Coisas, foi desenvolvido um sistema de automação residencial multiplataforma, com foco na elaboração de uma aplicação *mobile*. Neste sistema, o usuário é capaz de realizar as configurações de rede necessárias, cadastrar os dispositivos e realizar acionamentos pelo aplicativo InCasa, desde que conectado a uma rede Wifi. Um protótipo de sistema de iluminação atua como dispositivo real. Este protótipo é composto por um ESP-32, capaz de se comunicar com o banco de dados remoto da aplicação, e por outro ESP-32, que atua implementando os comandos do usuário no dispositivo real. Como forma de aplicar tendências para a área de automação residencial, que levam em conta o cenário dos mercados atuais e a crescente demanda por produtos interativos, foi investigada e realizada uma forma de adicionar a funcionalidade de comando por voz no aplicativo desenvolvido, utilizando pacote para o *flutter Speech To Text*.

Palavras-chave: Internet das coisas, automação residencial, comando por voz;

Abstract

Inspired on The Internet of Things concept, a multi platform system was designed for an automated home project, with focus on the settings of a mobile application. In this system, the user is able to perform the necessary network configurations, register the devices and perform operations via the InCasa app, needing only to be connect to a WiFi network to be functional. This prototype is composed of an ESP-32, capable of communicating with the remote database of the application and another ESP-32 that works by implementing the user's commands on the real device; a lighting system prototype acts as a real device. Also, in order to follow the current trends in the home automation area, mainly perceived by the growing demand for interactive products, there was also an attempt to implement the voice command functionality using a Speech to text package on flutter in the project.

Keywords: Internet of things, home automation, voice control;

Sumário

Lista de figuras

Lista de tabelas

Lista de códigos-fonte

Lista de acrônimos e notações

ALIVE	Ambiente de video interativo com vida artificial, do inglês *Artificial Life Interactive Video Environment*
BaaS	*Backend* como serviço, do inglês *Backend as a Service*
BD	Banco de Dados
BLE	Bluetooth de baixa energia, do inglês *Bluetooth Low Energy*
CRUD	do inglês, *Create, Read, Update and Delete*
EPC	Código Eletrônico de Produto, do inglês *Electronic Product Code*
EPG	Guia de Programação Eletrônica, do inglês *Electronic Programming Guide*
GPRS	Serviços Gerais de Pacotes por Rádio, do inglês *General Packet Radio Service*
GSM	Sistema Global para Comunicações Móveis, do inglês *Global System for Mobile Communications*
HTTP	Protocolo de Transferência de Hipertexto, do inglês *Hypertext Transfer Protocol*
IdC	Internet das Coisas
IDE	Ambiente de Desenvolvimento Integrado, do inglês *Integrated Development Environment*
IoT	Internet das Coisas, do inglês *Internet of Things*

LAN	Rede de Área Local, do inglês *Local Area Network*
LTE	Evolução de Longo Prazo, do inglês *Long Term Evolution*
MAC	Media Access Control
MAN	Rede de Área Metropolitana, do inglês *Metropolitan Area Network*
MCU	Unidade de Microcontrolador, do inglês *Microcontroller Unit*
NoSQL	Banco de Dados Não Relacional, do inglês *Non-Relational Database*
OMS	Organização Mundial da Saúde
PC	Computador Pessoal, do inglês *Personal Computer*
PLC	Comunicação Via Rede Elétrica, do inglês *Power Line Carrier*
RFID	Identificação por Radiofrequência, do inglês *Radio-Frequency Identification*
SDK	Kit de Desenvolvimento de Software, do inglês *Software Development Kit*
SMS	Serviço de mensagens curtas, do inglês *Short Message Service*
SNMP	Protocolo Simples de Gerência de Rede, do inglês *Simple-Network Management Protocol*
SO	Sistema Opreracional
SSID	Identificador do Conjunto de Serviço, do inglês *Service Set Identifier*
TCP/IP	Protocolo de Controle de Transmissão/Protocolo de Internet, do inglês *Transmission Control Protocol/Internet Protocol*
UI	Interface do Usuário, do inglês *User Interface*
uid	Identificação de usuário

URI	Identificador Universal de Recurso, do inglês *Universal Resource Identifier*
WAN	Rede de Área de Longa Distância, do inglês *Wide Area Network*
WLAN	Rede Local Sem Fios, do inglês *Wireless Local Area Network*

Capítulo 1

Introdução

Um dos aspectos inerentes às sociedades é a forma como se utiliza a tecnologia da época. A divulgação cultural de objetos “super tecnológicos”, cidades inteligentes e meios de transporte inovadores remonta a década de 1960, com o lançamento do desenho animado *Os Jetsons*. Ambientado em uma sociedade futurista, que já idealizava o uso de tecnologias contemporâneas e recentes como chamadas por vídeo pelo *tablet*, assistente pessoal, comida impressa, robô para limpeza da casa, dentre outras (MEHBOOB; ZAIB; USAMA, 2016).

Vinte anos depois, a ideia de conectar diferentes redes computadores se tornou algo possível, com a padronização do Protocolo de Controle de Transmissão/Protocolo de Internet, do inglês *Transmission Control Protocol/Internet Protocol* (TCP/IP) (FALL; STEVENS, 2012). Posteriormente adveio uma extensão da Internet atual. Aliada a microeletrônica e sistemas embarcados, a comunicação e o processamento de dados entre objetos e a atuação são a inovação no controle remoto de objetos. Elaborou-se assim o conceito de Internet das Coisas (IdC), do inglês, *Internet of Things* (IoT) (SANTOS; SILVA et al., 2016).

O termo “Internet of Things” foi primeiro citado em 1999, por Kevin Ashton como título de uma apresentação para a *Procter & Gamble* (P&G), associando o uso de sensores RFID/EPC na cadeia de suprimentos da companhia. Ao aliar Identificação por Radiofrequência (RFID) e o Código Eletrônico de Produto (EPC) foi desenvolvido um sistema global de identificação de itens. Mas seu uso se popularizou na premissa de que seria uma das maiores inovações em tecnologia para os próximos anos, usando os benefícios de mobilidade, *cloud computing* e *big-data* (ASTHON, 2010).

As possibilidades de aplicação de IdC são inúmeras: indústria 4.0, monitoramento e segurança, agricultura e pecuária, meios de transporte e logística, medicina e bem estar, automação em habitações, gerenciamento de recursos energéticos, entretenimento, dentre outras. Segundo Brody e Pureswaran (2015), a estimativa é de que existam cerca de 30 bilhões de dispositivos conectados em 2020. E a projeção é de que em 2050 esse montante exceda 100 bilhões (Figura 1.1). Tamanha heterogeneidade de dispositivos e meios de comunicação que envolvem as implementações de IdC apresentam desafios cruciais para seu sucesso: a padronização, a regulamentação e a segurança (SANTOS; SILVA et al., 2016).

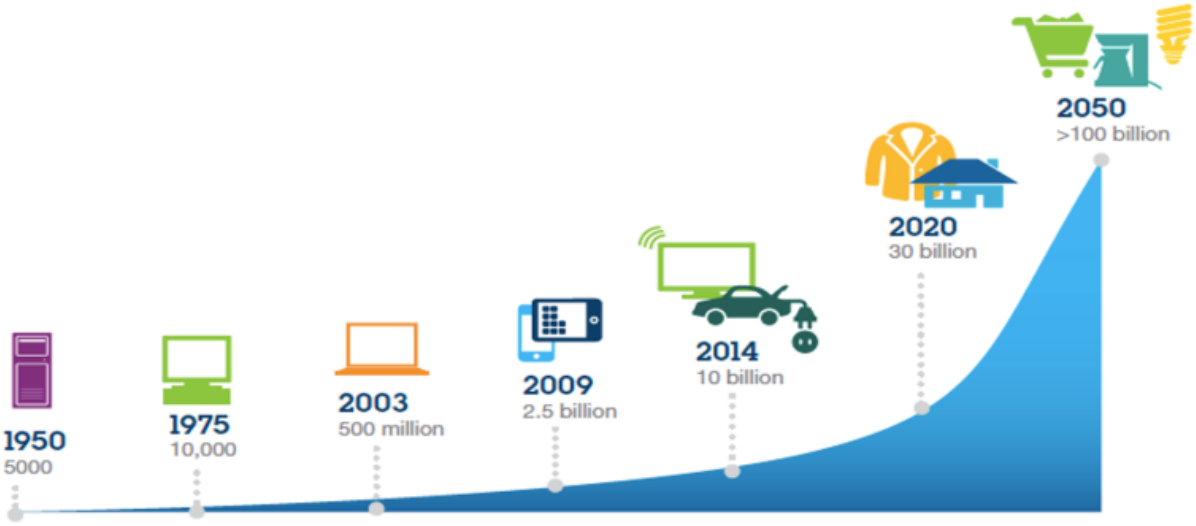


Figura 1.1 – Salto no número de dispositivos, desencadeado por melhoramentos significativos na computação (BRODY; PURESWARAN, 2015).

Em especial, os *smartphones* podem ser considerados como dispositivos fundamentais no paradigma IdC Oportunista, onde ocorre a interação harmoniosa entre humanos, sociedades e objetos inteligentes (ALOI et al., 2017).

No Brasil, segundo o IBGE (2018), a Internet era utilizada em 79,1% dos domicílios (cerca de 192,6 milhões de pessoa com acesso). Destes domicílios conectados à Internet, em 99,2% pelo menos uma forma de utilização se dava por meio de telefone móvel celular. O mesmo relatório aponta que o percentual de pessoas que acessaram a Internet com a finalidade de conversar por chamadas de voz ou vídeo e assistir a vídeos (de qualquer tipo) apresentou nítida tendência de crescimento. Isso evidencia a preferência dos brasileiros pelo uso de dispositivos *mobile*, ainda que para fins sociais e de entretenimento.

A Automação Residencial (conceito surgido nos anos 1970 com comandos via PLC (do inglês, *Power Line Carrier*)), visa alterar como indivíduos interagem em ou com um ambiente domiciliar. Segundo Muratori e Dal Bó (2011), ela atua melhorando as necessidades básicas de segurança, comunicação, gestão energética e conforto de uma habitação. Aliada ao nível atual de IdC, torna-se sinônimo de “casa inteligente” e domótica.

A Figura 1.2, da pesquisa “Comportamento do Consumidor Imobiliário para 2040”, mostra que em 2019, 56% dos entrevistados tem ou tiveram acesso algum tipo de assistente de voz. Sobre sistema de iluminação inteligente, cerca de 42% diz ter vontade de ter acesso a esta tecnologia. A pesquisa também aponta que a tecnologia será aplicada de modo a priorizar a customização pelo usuário, o ganho de tempo em sua rotina, monitoramento virtual em tempo real dos recursos habitacionais, segurança e mais acessibilidade. (CORDEIRO; BAGGIO; FRANÇA, 2019).

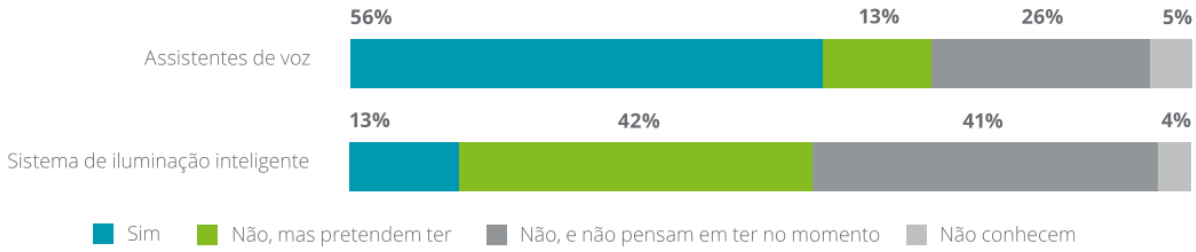


Figura 1.2 – Tecnologias para a casa. Adaptado (CORDEIRO; BAGGIO; FRANÇA, 2019).

Tecnologia Assistiva é um termo, anteriormente designado “Ajudas Técnicas”, que se refere a produtos, recursos, metodologias, estratégias, práticas e serviços que buscam oferecer funcionalidade, relacionada à atividade e participação de pessoas com deficiência, incapacidades ou mobilidade reduzida, visando sua autonomia, independência, qualidade de vida e inclusão social.

A automação residencial tem também em seu aspecto de Tecnologia Assistiva, um potencial de acessibilidade, ao viabilizar muitas vezes a ampliação de recursos funcionais de idosos, pessoas com deficiência ou mobilidade reduzida, promovendo maior grau de independência e inclusão (BERSCH, 2017).

1.1 Definição do Problema

O problema no qual este trabalho atua é no desenvolvimento de um aplicativo *mobile* multiplataforma (Android e IOS), com função de acionamento por voz e de um protótipo para acionamento via aplicativo.

1.2 Motivação

As motivações para o trabalho incluem as tendências de mercado para IdC e para o ambiente residencial, via componente essencial para aplicações IdC (*smartphones* utilizados em larga escala) e a possibilidade de avaliar o uso das tecnologias do projeto aplicado à tecnologia assistiva, por pessoas com mobilidade reduzida.

1.3 Objetivos

1.3.1 **Objetivo geral**

O objetivo principal deste projeto é o desenvolvimento de uma aplicação de automação residencial multiplataforma, com funções de comando via reconhecimento de voz, aplicado a um protótipo.

1.3.2 **Objetivos específicos**

Para alcançar esse objetivo, foi necessária a realização das seguintes ações:

- Desenvolver aplicação multiplataforma: revisar a linguagem de programação Dart com o *framework* Flutter. Definir a arquitetura da aplicação, vinculada a banco de dados *mobile*. Desenvolver a aplicação.
- Definir os protocolos de rede para comunicação do sistema: adequar a aplicação desenvolvida aos protocolos pretendidos. Determinar e implementar a utilização de *hardwares* de processamento para viabilização do sistema.
- Adicionar função de reconhecimento por voz: viabilizar o uso do pacote *open source Text To Speech* na aplicação desenvolvida. Realizar procedimentos de

calibração e testes em dispositivo móvel afim de tornar o uso do sistema comandado por voz algo direto para o usuário.

- Aplicar o sistema a uma rede elétrica residencial: definir a totalidade dos componentes eletrônicos e as especificações de rede que possibilitam o acionamento remoto dos dispositivos a serem automatizados. Realizar a integração das partes do projeto.

1.4 Organização do documento

Este documento é dividido em quatro capítulos. Neste primeiro capítulo, foi feita uma Introdução geral sobre sistemas de automação residencial. Também contém a definição do problema, a motivação e os objetivos deste trabalho de conclusão de curso.

O segundo capítulo, Fundamentos, é dividido em revisão bibliográfica, estado da arte e fundamentação teórica, que contém uma breve descrição de algumas ferramentas empregadas no projeto.

O terceiro capítulo, Metodologia, contém a relação de materiais e métodos utilizados, detalhamento do aplicativo, redes e custos estimados do projeto.

O último capítulo, Considerações Finais, levanta considerações acerca das atividades a desenvolvidas no Trabalho de Conclusão de Curso II.

Capítulo 2

Fundamentos

Neste capítulo é apresentado um breve histórico de tecnologias que possibilitaram o desenvolvimento do sistema de automação residencial proposto. Também são apresentados conceitos teóricos necessários para o desenvolvimento deste trabalho.

2.1 Revisão da literatura

Uma torradeira comandada pelo computador. Eis o primeiro dispositivo doméstico comandado via Internet, apresentado na conferência anual de tecnologia da informação *Interop*, em 1990. Foi utilizado um *laptop*, um *hardware*, com relé, controle de corrente, conexão de porta paralela e Protocolo Simples de Gerência de Rede, do inlgês *Simple Network Management Protocol* (SNMP). A ideia era mostrar uma aplicação pouco convencional e demonstrar que o protocolo SNMP poderia ser utilizado para controle físico, além do usual gerenciamento de dispositivos (roteadores, contadores) em redes IP. Neste ano, cerca de 3 milhões de pessoas tinham acesso à Internet (ROMKEY, 2017).

Os custos relacionados à computação, memória e hardware atingiram patamar comercializável quando as tecnologias relacionadas deixaram de ser de uso exclusivo militar no anos 1960. Ao alcançarem a indústria e encontrarem novos mercados foi possível contemplar mudanças radicais na maneira como humanos interagem com as máquinas, adaptando o que se via no ambiente de trabalho para o ambiente doméstico. A transformação de objetos inanimados em sistemas responsivos despertou a investigação pioneira deste tópico, no *MIT Media Lab*, com o protótipo *Smart Room*,

em 1991. Neste projeto, um ambiente foi instrumentado com câmeras e microfones. A ideia era que o ambiente estivesse a disposição das pessoas, como um "mordomo em *stand-by*". Foram realizados estudos sobre visão computacional, detecção de gestos, rastreamento da cabeça, recuperação de forma, interpretação de áudio, reconhecimento facial e de voz. Logo, projetos experimentais semelhantes se espalharam por centros de pesquisa pelo mundo (PENTLAND, 1998).

A Figura 2.1 abaixo registra um dos projetos, voltado para "ambiente de vídeo interativo com vida artificial" (ALIVE). O sistema de monitoramento determina onde o humano está e simulações determinam onde as "criaturas virtuais" podem estar. Em seguida a cena é renderizada e exibida ao usuário. Se a criatura se movimenta para trás do usuário, ela tem partes do "corpo" ocultadas, de acordo com a posição (CASEY; GARDNER; BASU, 1995; ALIVE, 1995).

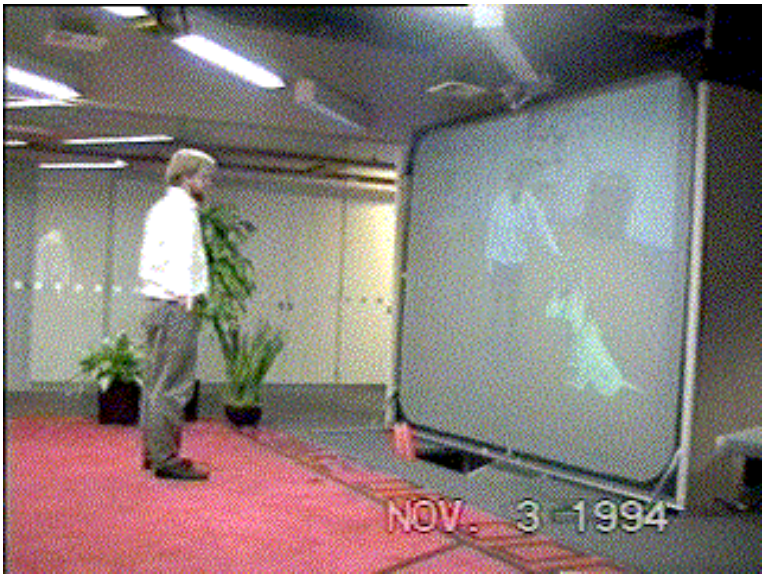


Figura 2.1 – Projeto "ALIVE", derivado de *Smart Room* (ALIVE, 1995).

Em meados dos anos 1990, estudos apontavam as dificuldades para implementações de ambientes domésticos inteligentes: alto investimento do consumidor para aquisição, necessidade de modificar ambientes (*retrofit*) para instalação, falta de protocolo comum entre as aplicações e "impulso pela tecnologia", onde se presta pouca atenção às reais necessidades dos usuários. A pesquisa também destaca que nesta época, o *design* de casas inteligentes falha ao idealizar uma parte importante da vida diária sem verificar os impactos no trabalho diário de uma casa (MEADOWS-KLUE, 2004).

Segundo Bowden e Offer (1994), as intervenções em domicílios poderiam ser consideradas economizadoras de tempo (como uma máquina de lavar louças) ou consumidoras de tempo (como televisão, rádio).

Estudos analisaram o comportamento de telespectadores diante do uso de um dos eletrodomésticos mais difundidos, a TV. Com a inserção de um Guia de Programação Eletrônica (EPG) ainda em 1981, a listagem da programação dos canais era disponibilizada vinte e quatro horas por dia. Mesmo com este recurso disponível nos primeiros EPGs, os usuários tendiam a seguir um curso, mostrado na Figura 2.2, onde o EPG era a última opção para escolha do que assistir (LOGAN et al., 1995; CASEY; GARDNER; BASU, 1995).

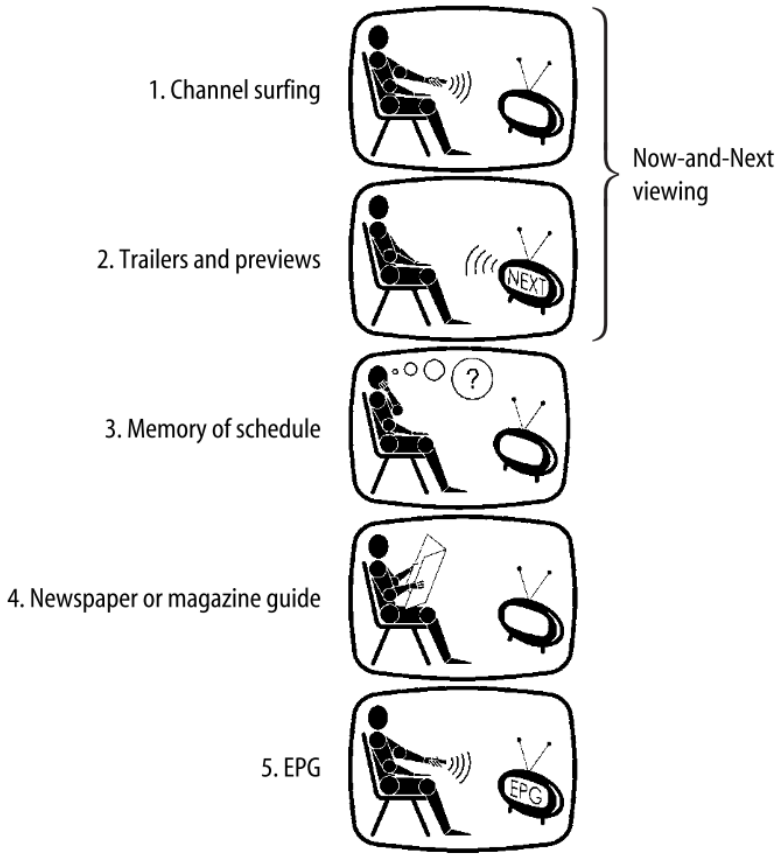


Figura 2.2 – Sequência de passos para escolher um programa na TV (MEADOWS-KLUE, 2004).

Jones e Carey (2000) concluíram que os telespectadores não estavam ainda habituados e achavam o EPG, difícil de operar. Ao inserir a função de exibição do canal selecionado na tela, ao lado do EPG, a adesão ao recuso teve aumento. Isso evidencia como o *design* de uma aplicação pode influenciar no sucesso da mesma.

A aquisição de Computadores Pessoais (PCs) também foi estudada nos anos 1990. No início da década, o PC era usado para trabalho e para jogos, por alguns membros da casa. Com a incorporação de PCs nos domicílios, além do surgimento de mais softwares e maior acesso à internet, o mesmo passou a ser usado para outras funções: educação, comunicação familiar, recreação, agendamento de viagens, compras e controle de finanças.

Mais membros passaram a utilizar esta ferramenta com frequência. Com o PC compartilhado por várias pessoas na casa, algumas questões começaram a ser levantadas como o melhor local para sua instalação e a supervisão parental do uso do computador, de suas configurações e da internet. Em algum ponto, começou-se a notar conflitos causados pelo uso comum de um computador na casa, levantando-se a possibilidade de cada membro da família ter um próprio PC (MEADOWS-KLUE, 2004).

A Figura 2.3 a seguir ilustra o computador industrial, o computador pessoal e computadores portáteis. A diminuição do tamanho dos componentes eletrônicos também alterou a comunicação, num processo que uniu computadores portáteis a telefones móveis.

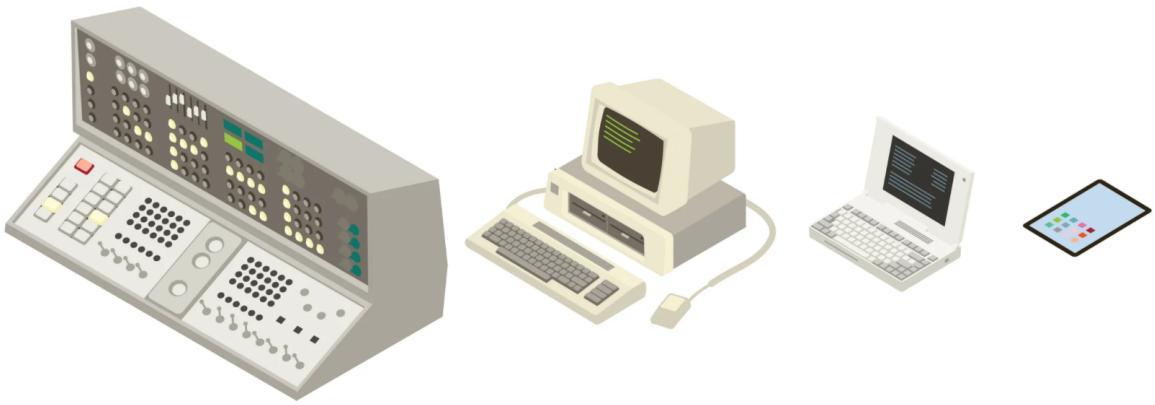


Figura 2.3 – História dos computadores (CALVELLO, 2019).

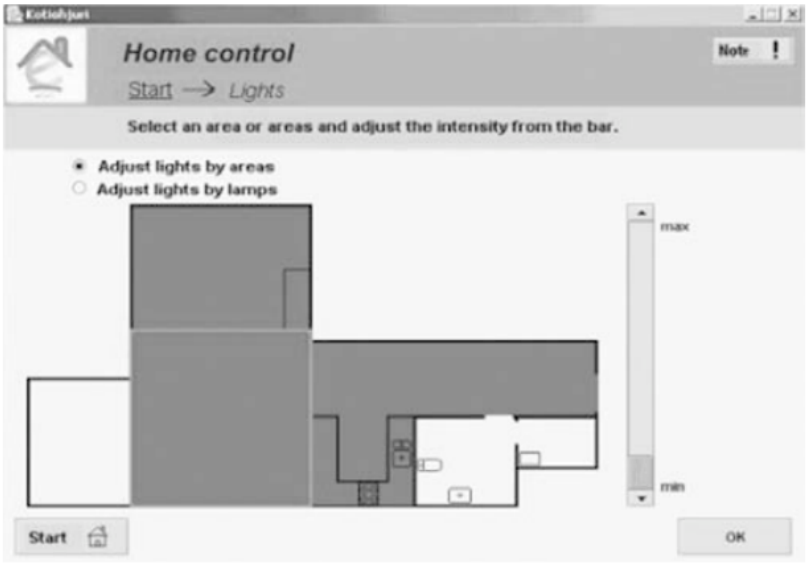
Os primeiros protótipos multifuncionais de smartphones datam do final da década de 1990. O *Simon* da IMB, *Communicator* da Nokia, *PdQEm* da Qualcomm. Pequenas telas, poucas aplicações, alfabeto integrado ao teclado numérico e taxas de dados próximas a 14,4 kbps com GSM. Em 1999 a companhia telefônica japonesa NTT DoCoMo lançou o i-mode, sistema que viabilizava visualização de dados, websites, email em um navegador no celular, com tela colorida, usando C-HTML. Nesse mesmo con-

texto, a Blackberry lançava seu primeiro *smartphone* voltado para troca de emails, com teclas dedicadas ao alfabeto no teclado (ISLAM; WANT, 2014).

Em desenvolvimento, buscando mais funcionalidades e velocidade de dados, o uso de dispositivos móveis para automação residencial já foi idealizado para esta aplicação nos anos 2000.

Koskela e Väänänen-Vainio-Mattila (2004) conduziram um estudo empírico, comparando o uso de três interfaces funcionais (um PC, um terminal de mídia para TV e um telefone móvel), em um apartamento com dois moradores, por seis meses. Os dispositivos com as interfaces gráficas foram conectados via redes WLAN, Ethernet LAN e GPRS com o computador principal, que funcionava como uma ponte, enquanto os "objetos inteligentes" estavam conectados ao *hub* serial.

Na Figura 2.4 a seguir são mostradas as interfaces, para PC e telefone móvel, desenvolvidas por Koskela e Väänänen-Vainio-Mattila (2004).



(a) GUI para PC.

(b) Interface para telefone móvel.

Figura 2.4 – Interfaces gráficas (KOSKELA; VÄÄNÄNEN-VAINIO-MATTILA, 2004).

No estudo anterior, parâmetros relacionados à mobilidade, como a facilidade de configurar as lâmpadas e outros componentes de automação residencial de qualquer lugar do apartamento, mesmo fora dele, foram destacados pelos usuários como algo desejável, por adicionar praticidade à rotina diária.

Em 2007 a Apple lançou o *iPhone* com iOS. No ano seguinte foi lançado pelo Google o sistema operacional (SO) *Android*. Juntamente com as redes de dados de alta velocidade, 3G e 4G (LTE), telas capacitivas, e interfaces mais complexas, grandes transformações aconteceram no uso de telefones móveis.

A Figura 2.5 ilustra a evolução do celular através de alguns modelos. As tendências seguem indicadores de consumo da época. A partir do Motorola 8900X-2 de 1994, com 14cm de antena e 1,6kg percebe-se diminuição dos dispositivos. A tela começa a ganhar destaque nas preferências de consumo, ao dispor de cores e mais itens no menu. Hoje em dia, boa parte dos requisitos dos consumidores envolvem a responsividade da tela, sua durabilidade, adaptabilidade e tamanho (LIU; YU, 2017).

Figura 2.5 – Evolução do celular/*smartphone* (VELOCITY WERX, 2018).

2.2 Estado da Arte

Nesta seção são apresentadas algumas abordagens recentes que contextualizam sistemas *mobile* para automação residencial.

2.2.1 **Tecnologias e Mercado**

A Figura 2.6 abaixo mostra percentuais do uso de algumas tecnologias por estadunidenses, partindo do uso de Internet em 1994 até 2018. De 2016 para 2018, houve pouca variação nos dados, sendo que ocorreu leve diminuição no uso de *desktop*. Uma vez que a pesquisa leva em conta larga escala de idade, os dados apontam para uma saturação em alto percentual do uso destas tecnologias, especialmente de celulares (PEW RESEARCH CENTER DATA, 2018).

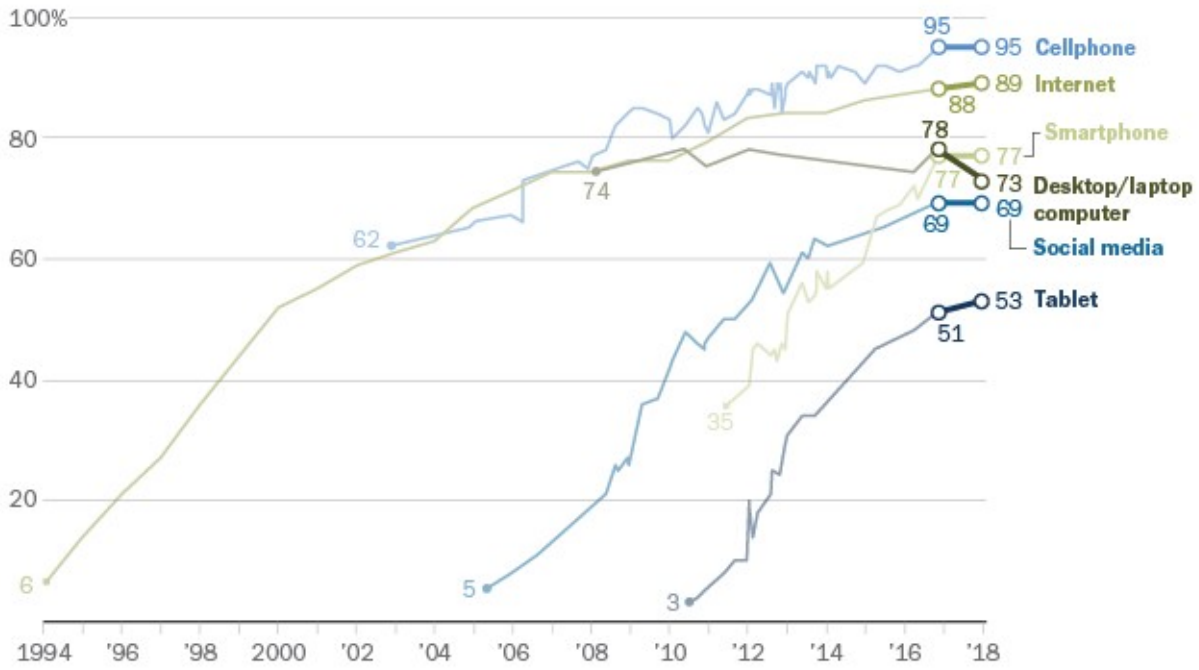


Figura 2.6 – Uso de tecnologias nos EUA de 1994 a 2018 (PEW RESEARCH CENTER DATA, 2018).

De acordo a agência de dados de mercado e consumidores Statisa (2020), no que se refere ao mercado mundial de sistemas operacionais, desde 2012, o uso de Android teve crescimento expressivo. Em maio de 2020, destaca-se como o SO líder mundial, controlando o mercado com 74,6% por cento de participação. O Google Android e o Apple iOS, juntos, possuem atualmente quase 99% da participação no mercado global.

A Figura 2.7 mostra as participações mundiais de alguns SOs. Ao mesmo tempo que se vê um domínio de dois SOs em relação aos outros, em relação a IdC, pode significar um ponto favorável em meio a falta de padronização (no que se refere suporte de aplicações). Claro que a falta de padronização se estende a outros dispositivos e protocolos, além de *smartphones*.

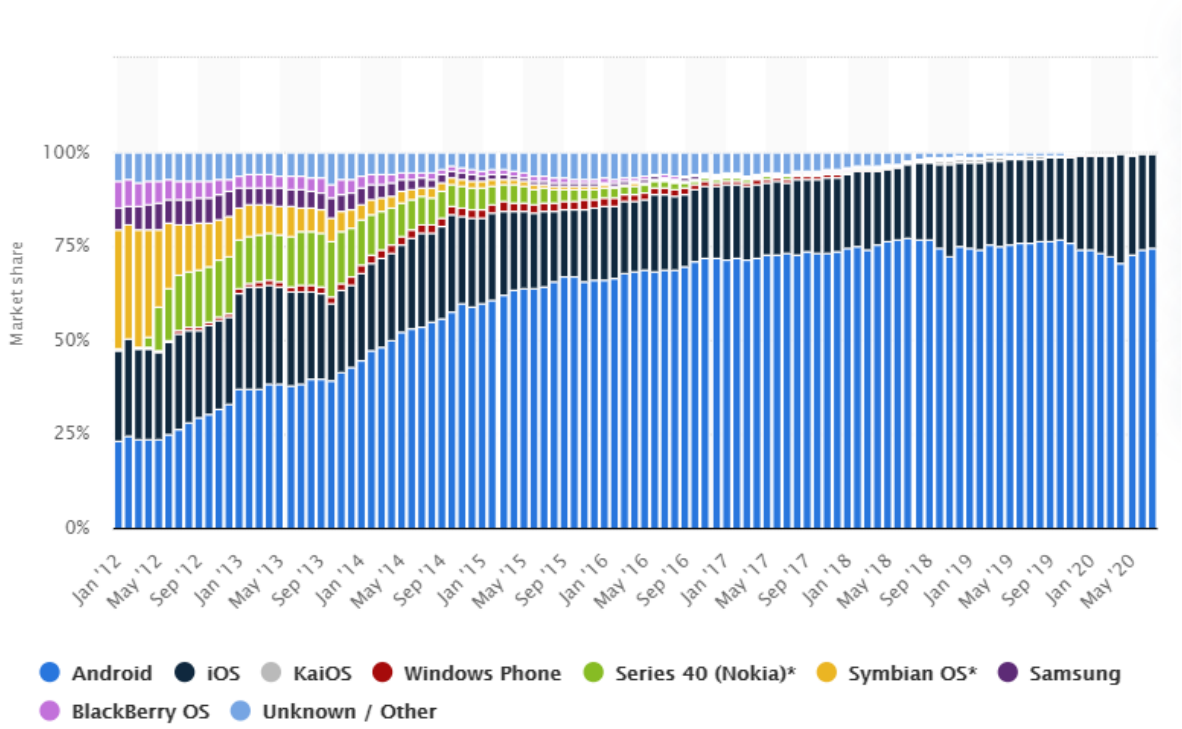


Figura 2.7 – Divisão do mercado de SOs *mobile* de 2012 a 2020 (STATISA, 2020).

Uma outra tecnologia ainda pouco abordada em 2018 se mostrou uma tendência em 2020: assistentes virtuais comandados por voz ou, em inglês, "*smart speakers*".

A Figura 2.8 mostra o impacto na indústria de assistentes de voz. Não se trata de um indicador de consumo, mas sim da adoção e intenção expressa por profissionais em apoiar os ecossistemas. Cada plataforma recebeu uma pontuação de -250 a 250. O produto de maior impacto do segmento em 2020 é Alexa, da Amazon, pioneira em *smart speakers*. Em seguida está o Google Assistente. (VOICEBOT, 2020).

Alexa tem ótima aplicabilidade no ambiente doméstico e integração com outros serviços da Amazon. Já o Google Assistente tem compatibilidade Android, IOS e serviços próprios do *Google*.

Dentre aplicações capazes de alterar o ambiente doméstico, existem vários projetos de inovação que podem em algum ponto ter seu controle/monitoramento utilizando *smartphones*, como espelhos inteligentes, robôs (humanoides ou não) capazes de auxiliar em algumas tarefas domésticas.

Na Figura 2.9 pode ser visto o protótipo de mobília inteligente, também um projeto com origem no do *MIT Media Lab*. Idealizado com o princípio de arquitetura robótica, se propõe a ser o essencial em habitações pequenas, como apartamentos em cidades

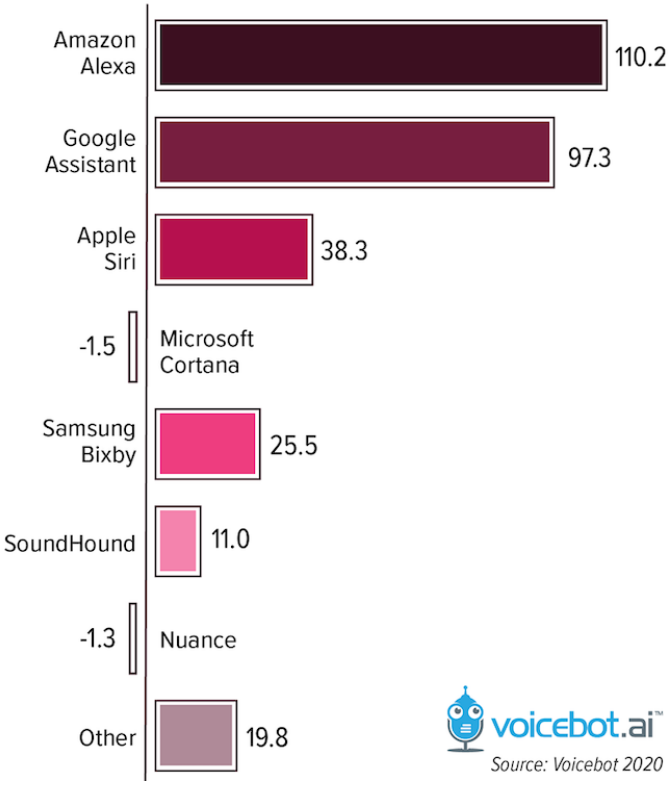


Figura 2.8 – Impacto das plataformas de *smart speakers* (VOICEBOT, 2020).

grandes. A mobília interativa se move no ambiente, controlada por gestos. Se torna cama, mesa, banca, extensão do banheiro (LARREA-TAMAYO, 2015).

Um demonstrativo deste protótipo de mobília inteligente pode ser visto em `https://bit.ly/youtubeCityHome`.

Figura 2.9 – Projeto “CityHome” (LARREA; LARSON, 2016).

2.2.2 **Tecnologia Assistiva**

Segundo a Organização Mundial da Saúde, mais de 1 bilhão de pessoas vive com alguma forma de deficiência. A maioria destas pessoas são idosos ou pessoas com deficiência. Este número tem uma tendência de aumento, devido envelhecimento das populações e declínio natural de funções. Estima-se que o número de pessoas que precisarão de produtos assistivos ultrapasse 2 bilhões em 2050 (OMS, 2017).

A tecnologia assistiva pode auxiliar pessoas com impedimentos de longo prazo de natureza física, mental, intelectual ou sensorial na realização de atividades básicas da vida diária com níveis de autonomia (BRASIL, 2015).

São considerados produtos assistivos quaisquer produtos, dispositivos, equipamentos, instrumentos ou *softwares*, cujo propósito primário seja manter ou melhorar a funcionalidade e a independência individuais, promovendo o bem-estar. Os produtos assistivos também são usados para prevenir a deficiência e condições secundárias de saúde (OMS, 2017).

A Lista de Produtos Assistivos de Alta prioridade da OMS inclui itens como: *software* de comunicação, assistente pessoal digital, telefone móvel simplificado, dispositivo de comunicação por vídeo, detector de queda, dentre outros. Estes itens, alguns considerados “tecnologias vestíveis”, podem ser integrados ao contexto de casas inteligentes.

Segundo Majumder et al. (2017) algumas possibilidades de aplicação voltadas para tecnologia assistiva:

- Segurança: Detecção de quedas e emergências;
- Saúde: Telemetria, monitoramento físico, reabilitação;
- Social: Conectividade com amigos e família;
- Mobilidade: Locomoção, autonomia em tarefas que exigem força.

A Figura 2.10 ilustra o funcionamento de um sistema de detecção de quedas onde, na casa do usuário existem dispositivos para monitoramento e sensoriamento de possíveis quedas, que se comunicam por redes com centros de emergência. Caso ocorra uma queda, um aviso é direcionado ao centro de emergência, via SMS, ligação ou algum outro meio de notificação.

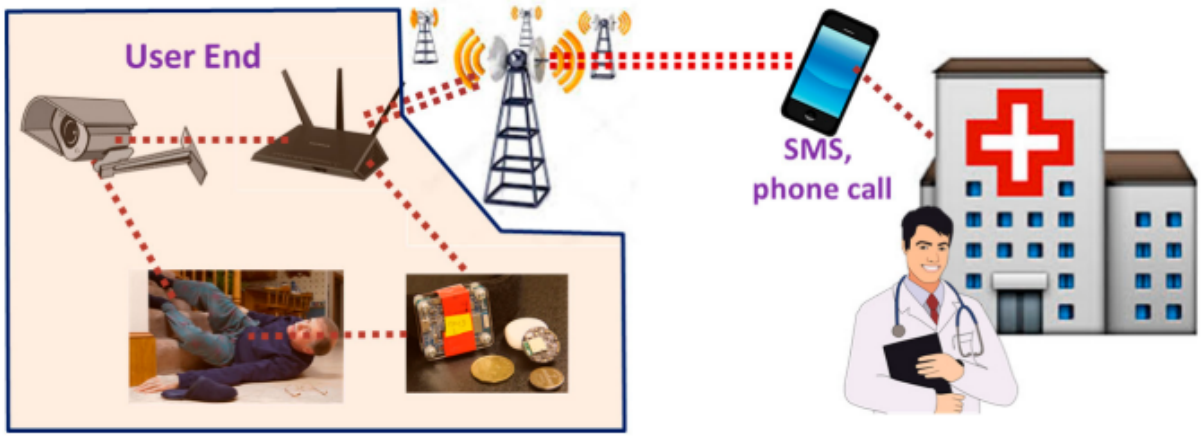


Figura 2.10 – Esquema de funcionamento de sistema de detecção de quedas (MAJUMDER et al., 2017).

2.3 Fundamentação Teórica

Esta seção apresenta as principais ferramentas empregadas para o desenvolvimento deste trabalho de conclusão de curso.

2.3.1 Redes

O termo “rede de computadores” pode ser definido na norma ISO/IEC 7498-1 como “Um conjunto de um ou mais computadores, software associado, periféricos, terminais, operadores humanos, processos físicos, meios de transferência de informação, entre outros componentes, formando um conjunto autônomo capaz de executar o processamento e a transferência de informações” (ISO 7498-1:1994…, 1996).

Existem protocolos específicos para cada aplicação de rede, seja ela local (LAN), intermediária (MAN) ou de longa distância (WAN), que variam em abrangência e alcance. A Figura 2.11 mostra algumas comunicações sem fio.

2.3.2 Arquitetura de Redes

Um ponto fundamental para alcançar a comunicação entre diferentes dispositivos é a compatibilidade entre as redes por onde a informação passa. Em um sistema de automação residencial a informação passa por diversas conversões de protocolo, já que os dispositivos que a compõe (atuadores, sensores, controladores, sistemas embarcados, *gatways*, entre outros) não possuem um protocolo único padrão.

Figura 2.11 – Algumas das plataformas de comunicação sem fio e sua abrangência em rede (MAJUMDER et al., 2017).

Tucic et al. (2014) consideram que uma arquitetura centralizada (Figura 2.12a)) é composta por um dispositivo embarcado, em modo de entidade central (como um *hub*, roteador), que conecta entidades de *endpoints* (como atuadores e sensores) em um nível de rede local. Também consideram que quando há comunicação entre mais de uma entidade central ou entre sistemas logicamente independentes, trata-se de uma arquitetura descentralizada (Figura 2.12b)).

Os desafios de implementação se sistemas de automação residencial esbarram na dimensão, com infraestrutura variável (ambientes que necessitam de *retrofit* ou implementação pontual), tolerância de funcionamento entre momentos de falha, *delays* entre comunicações e uso efetivo dos recursos.

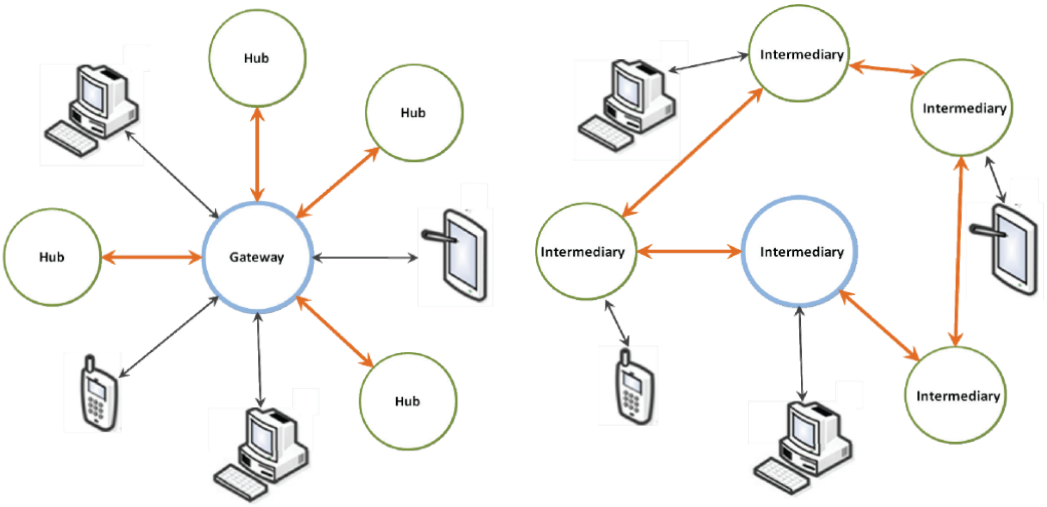


Figura 2.12 – Arquiteturas de rede: centralizada e descentralizada (TUCIC et al., 2014).

As redes podem ser ponto a ponto, cliente-servidor e/ ou combinação de ambos.

2.3.3 **Redes Sem Fio**

O termo redes sem fio, do inglês *wireless networks*, engloba diferentes tipos de comunicação: via satélite, infravermelho, rádio *broadcast*, Wi-Fi, dentre outras. O protocolo Wi-Fi refere-se a tecnologias WLAN que seguem o protocolo IEEE 802.11.

Os padrões do protocolo seguem técnicas de transmissão específicas. Por exemplo, o 802.11n opera em 2.4 GHz ou 5 GHz, com taxas de até 600Mbps.

Uma rede sem fio pode ser composta por: placas de rede, concentradores, *switches* e roteadores. Cada placa de rede possui um identificador de controle de acesso à mídia, denominado endereço MAC. Para estabelecer uma conexão entre dispositivos, o provedor da rede fornece um endereço IP.

2.3.4 **Dart e Flutter**

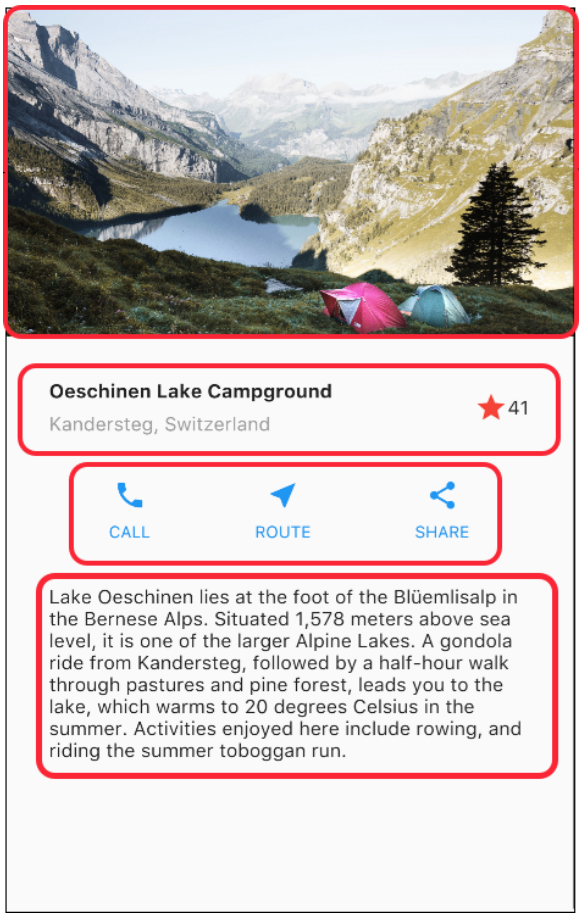
Dart foi a linguagem de programação escolhida para o desenvolvimento do aplicativo *mobile* deste trabalho. Trata-se uma linguagem otimizada para o cliente para aplicativos em múltiplas plataformas: é possível usar Dart para desenvolvimento *mobile*, *desktop*, *server* e *web*. Desenvolvida pela *Google*, é uma linguagem de programação orientada a objetos, baseada em classes, *open source*, com sintaxe semelhante a *C*. O objetivo desta linguagem é que um código que seja econômico, não denso.

É possível com Dart construir aplicativos *mobile* nativos iOS e Android. Para outros tipos de plataformas, é necessário utilizar o kit de desenvolvimento de software (SDK) Dart. Já para o desenvolvimento *mobile*, é recomendado utilizar o Flutter, um SDK com ferramentas UI, também desenvolvido pela *Google* (DART, 2020).

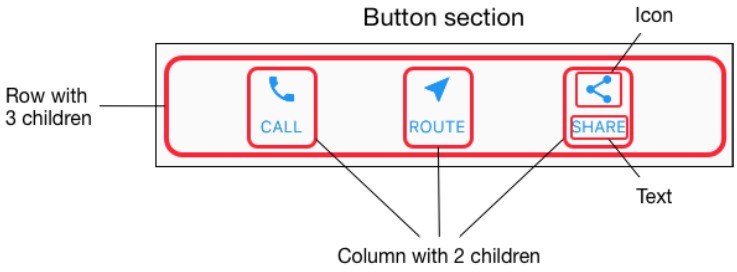
O Flutter é um *framework* do *Google* que viabiliza a construção de aplicativo nativo em diferentes sistemas operacionais. Possibilita uma boa experiência ao usuário, disponibilizando elementos visuais “bonitos”, praticidade para *designers*, devido fidelidade entre aplicação multiplataforma e fácil prototipagem e promete ao desenvolvedor ser uma ferramenta fácil e redutora de custos e complexidade.

Um código Flutter é construído com *widgets*, estrutura inspirada em React (interface JavaScript para UI). Os *widgets* descrevem a aparência de sua visualização de acordo com sua configuração e estado atuais (FLUTTER, 2020).

Com a Figura 2.13 abaixo é possível entender o mecanismo de Layout do Flutter. Primeiro identifica-se ou projeta-se uma coluna (sendo que o projeto é livre e existem inúmeras formas de se fazer layouts em Flutter), com os elementos maiores.



(a) Elementos maiores.



(b) Elementos menores.

Figura 2.13 – Diagramas de Layout (FLUTTER, 2020).

Em 2.13a) são destacados quatro elementos dentro de uma coluna. Em 2.13b) a linha chamada *Button section* apresenta outros três elementos como derivadas do *widget* linha. E cada um destes elementos é um *widget* coluna, onde a primeira linha é um ícone e a segunda linha é um texto.

2.3.5 **Firebase**

Firebase é uma ferramenta atualmente desenvolvida pelo *Google* que auxilia o desenvolvimento de aplicativos. Oferece infraestrutura, gerenciamento de marketing, performance e distribuição de aplicativos. São mais de dezessete serviços do Firebase, para desenvolvimento, qualidade e expansão de produtos. O Firebase possui versões gratuitas e pagas. (FIREBASE, 2020a).

O Cloud Firestore é um destes serviços. Trata-se de um banco de dados não relacional (NoSQL) hospedado na nuvem, que oferece sincronização em tempo real, suporte off-line, além de consultas eficientes a dados. Possui uma cota gratuita de 1 GB de dados armazenados e 50.000 leituras de documentos por dia.

O Firebase Auth é um serviço que oferece diversos métodos de autenticação: e-mail/senha, uso direto de sistema de contas próprio, ou provedores de terceiros, como o Google, Facebook, Microsof, Twitter, dentre outros. Os recursos deste serviço são gratuitos, exceto a autenticação por telefone. A figura 2.14 mostra o logotipo do Firebase.

Figura 2.14 – Logotipo Firebase (FIREBASE, 2020a).

2.3.6 **ESP-32**

ESP32 é um módulo microcontrolador (MCU) Wi-Fi, Bluetooth, *Bluetooth Low Energy* (BLE), de ampla variedade de aplicações, desde de redes de sensores de baixa potência a tarefas mais exigentes, como codificação de voz, *streaming* de música e decodificação de MP3 (EXPRESSIF, 2019).

No centro deste módulo está o chip ESP32-D0WDQ6. O chip embutido é projetado para ser escalável e adaptativo. Existem dois núcleos da CPU que podem ser controlados individualmente, e a frequência do clock da CPU é ajustável de 80 MHz a 240 MHz.

O ESP-32 pode ser programado em diversas IDEs, como a SDK da Espressif, MicroPython, Visual Studio, Arduino IDE e outros editores. Programas ESP32 são escritos em linguagem C (KURNIAWAN, 2019).

O ESP-32 suporta os protocolos Wi-Fi 802.11 b/g/n (802.11n a 150 Mbps) e Bluetooth v4.2 BR/EDR e especificações BLE (EXPRESSIF, 2019). A Figura 2.15 mostra um destes módulos.

Figura 2.15 – ESP32 (FILIPEFLOPE, 2020)

2.3.7 **ESP-NOW**

O ESP-NOW é um protocolo de comunicação sem fio definido pela Espressif. Permite a comunicação de dispositivos sem o uso do WiFi. O protocolo é semelhante ao wifi de 2,4 GHz, baixa potência. O emparelhamento entre dispositivos é um passo necessário antes da comunicação. Após o emparelhamento, a conexão é segura e ponto a ponto, sem a necessidade de *handshake* (reconhecimento de dispositivos antes da transmissão de dados).(ESPRESSIF, 2021a).

2.3.8 **Módulo HW-316**

O módulo HW-316 é um módulo de relés de quatro canais, acionados por sinais de 5V. Ele pode ser usado para controlar vários aparelhos e equipamentos de altas correntes. Trata-se de uma interface padrão entre o controlador e o sistema. Pode ser controlado diretamente pelo MCU (TECHNOLOGY, 2015).

Na tabela 2.1 encontram-se algumas especificações deste módulo, ilustrado na Figura 2.16.

Tabela 2.1 – Características do módulo de relés

Características	
Tensão de operação	5VDC (VCC e GND)
Tensão de sinal	5VCD (IN1, IN2, IN3 e IN4)
Corrente típica de operação	20mA
Dimensões	2mm x 54mm x 17mm
Cada relé possui 6 terminais (VCC, IN1, IN2, IN3, IN4, GND)	
Contato do relé permite tensão de até 30 VDC a 10A ou 250VAC a 10A	

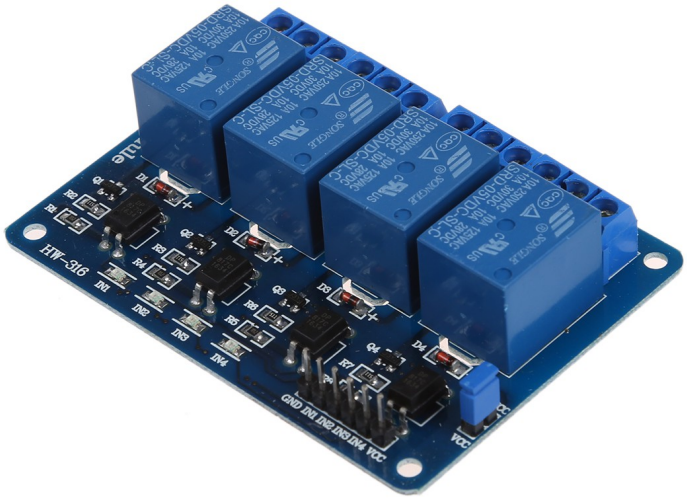


Figura 2.16 – Módulo de relés HW-316 (ELETROGATE, 2020).

Capítulo 3

Metodologia

Este capítulo apresenta as ferramentas utilizadas e as etapas realizadas para o desenvolvimento do protótipo de sistema de automação residencial.

A Figura 3.1 mostra a algumas das relações que compõe o projeto, quando estiver em funcionamento.

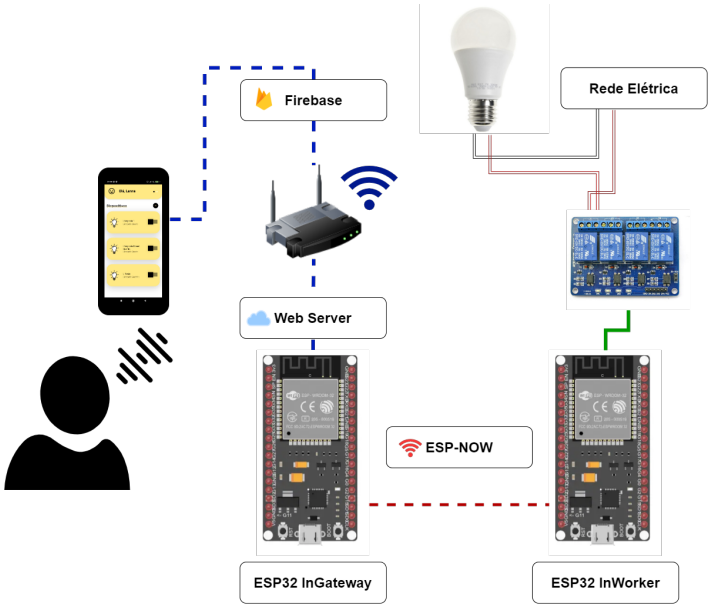


Figura 3.1 – Relações do projeto.

O usuário realiza o comando de acionamento pelo *smartphone*, que está conectado à Internet. Os dados são alterados no banco de dados remoto Firebase. O ESP32 InGateway, depois da etapa de configuração, com um WebServer, recebe os dados do Firebase e envia ao ESP32 InWorker utilizando protocolo ESP-NOW. Por sua vez, este último altera o estado do relé, conectado a uma lâmpada e a rede elétrica.

3.1 Materiais e Métodos

Nesta seção estão dispostos os materiais e métodos utilizados, os *softwares* de *design*, prototipagem, IDEs, os componentes eletrônicos e itens diversos. Também trata dos passos para o desenvolvimento e integração das partes do projeto.

O aplicativo *mobile* desenvolvido é denominado *InCasa*. Para sua concepção, foi utilizado o *FluidUI*, ferramenta online, com versão gratuita, para prototipagem de aplicativos (FLUID UI, 2020). Para elaboração de diagramas e fluxogramas foi utilizado o *Draw.io*, aplicação também online e gratuita para este fim (DRAW IO, 2020).

A programação do aplicativo foi realizada na IDE *Android Studio 4.1*, utilizando a linguagem *Dart* e o *framework Flutter* (ANDROID, 2020).

Os microcontroladores escolhidos para o protótipo de automação são um par de ESP32. Para programação dos ESP32, foi utilizada a IDE *Arduino 1.8.13*, na linguagem C++ (ARDUINO, 2020).

Como componentes da parte eletrônica, foram utilizados cabos de dados USB micro B, fios elétricos, *protoboards*, e um módulo com quatro relés, um soquete, uma lâmpada e um multímetro. A alimentação dos controladores foi feita diretamente no Notebook. Mas ambos necessitam de uma alimentação de 5V.

Os testes do aplicativo foram realizados em um *smartphone* com SO Android. Para testes do aplicativo em iOS, seria necessário ter acesso a um computador macOS (FLUTTER, 2021b,c).

3.2 Aplicativo InCasa

3.2.1 **Requisitos**

Cada aplicação *mobile* possui requisitos diferentes. O processo de identificação de requisitos do aplicativo é denominado “projeto de sistema”. Ao determinar o que é desejado na aplicação, é possível “projetar padrões”. Ambos são ferramentas que entregam a arquitetura final de uma aplicação (BIZZOTTO, 2020).

Como passo inicial para levantamento de requisitos funcionais, foi desenvolvido no *Fluid* um esboço com as telas pretendidas do aplicativo. Este esboço encontra-se disponível em `https://bit.ly/fluidincasa`. Para navegar pelas telas do esboço, é

necessário clicar em *Restart Preview* e clicar nos botões de navegação do aplicativo simulado.

A seguir estão listados os requisitos funcionais do aplicativo:

- Página de *Login*:
  - Login com Google;
  - Login com Email;
- Página *Home*:
  - Acesso à página de perfil, configurações e *logout*;
  - Acesso à página para adicionar dispositivos;
  - Cadastro de novos dispositivos;
  - Listagem de dispositivos;
  - Edição de dispositivos;
  - Exclusão de dispositivos;
  - Modo voz;
- Página de Perfil, Configurações e *Logout*;
  - Entrada do nome do usuário
  - Configurações de rede;
  - *Logout* do aplicativo;

O código-fonte do projeto desenvolvido pode ser encontrado:

- *InCasa* - Disponível em `https://github.com/lannagif/incasa`.

A Figura 3.2 mostra um resumo da árvore de *widgets* do aplicativo desenvolvido com Dart/Flutter. Em azul, são classes relacionadas a interface do usuário. Em rosa, são classes que indicam acesso de escopo com Provider. Amarelo claro indicam a origem dos respectivos Providers. *Material App*, de verde, é uma classe pré definida em Flutter que inicializa os componentes de *design* da aplicação.

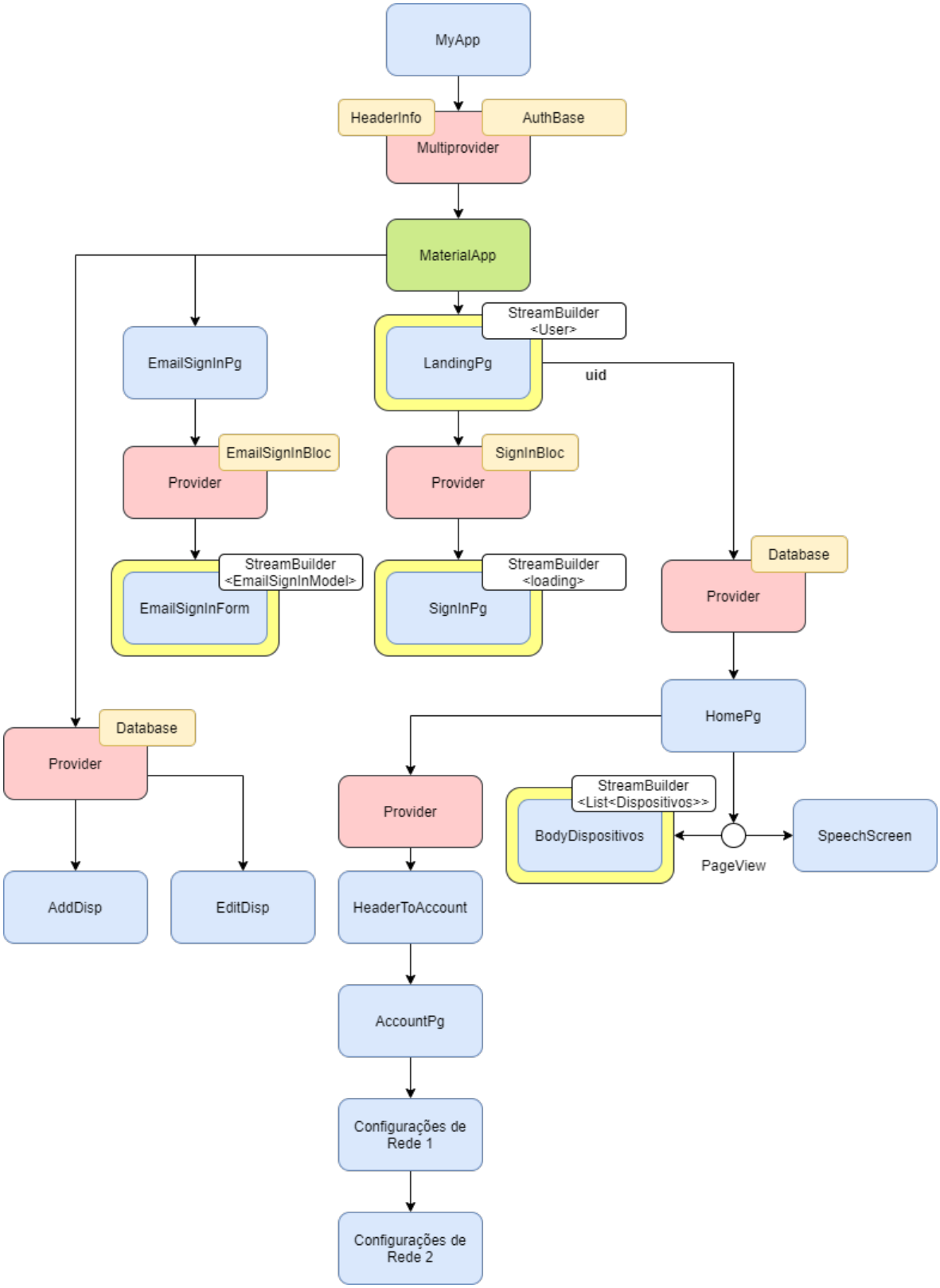


Figura 3.2 – Resumo da árvore de *widgets* do aplicativo.

3.2.2 **Autenticação**

Para este projeto, ao realizar o primeiro acesso, o usuário deverá criar uma conta. Se já possuir uma conta, deverá inserir suas credenciais. Trata-se de uma autenticação que controla o acesso do usuário ao restante do aplicativo. Após realizar a autenticação, a tela que solicita os formulários de *login* não aparece novamente ao abrir o aplicativo até que seja feito o *logout* pelo usuário.

O pacote *Firebase Authentication* fornece serviços de *backend* para autenticar usuários no aplicativo. A documentação oficial mostra os os passos para adicionar este serviço a um projeto Flutter (FIREBASE, 2020b).

Neste trabalho optou-se por utilizar dois destes provedores compatíveis com os serviços de autenticação disponíveis: conta do Google e email/senha.

A Figura 3.3 mostra o funcionamento das requisições para entrada no *InCasa*.

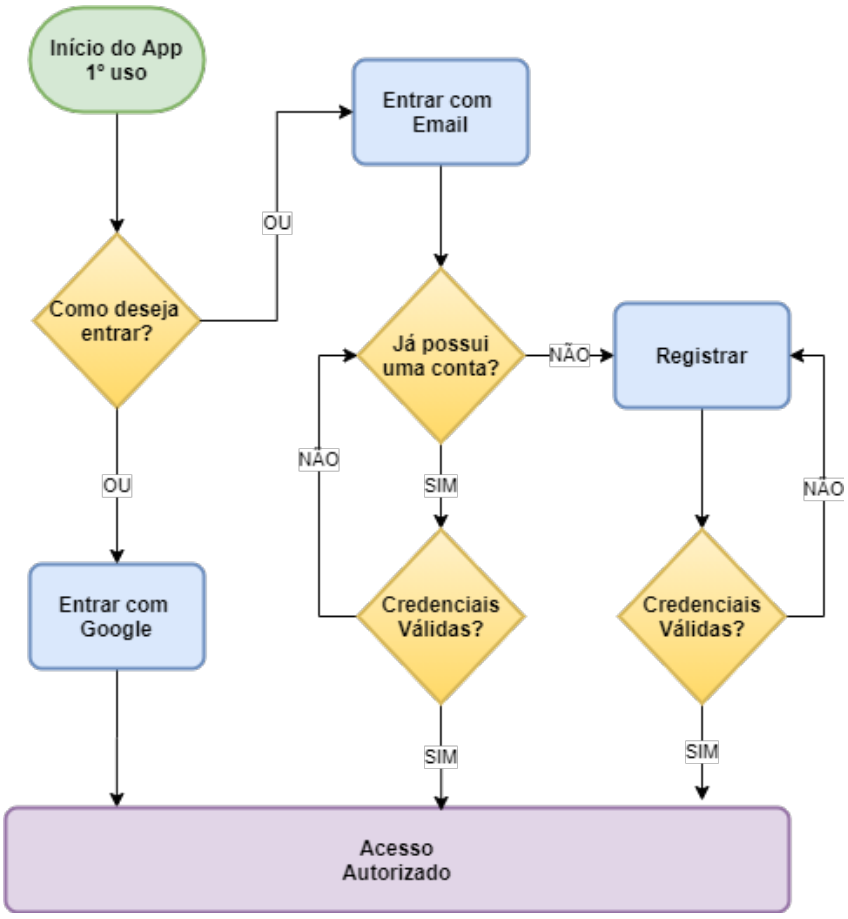


Figura 3.3 – Fluxograma de Login.

A classe *Auth* implementa *AuthBase*, que contém os métodos de entrada no aplicativo. Nesta classe também existe um método *get currentUser* que retorna a id do

usuário criado no Firebase.

3.2.3 Esquema de Dados

Nesta seção é descrita a criação e armazenamento dos dados do aplicativo. O pacote *Cloud Firestore* é um banco de dados de nuvem NoSQL. É uma ferramenta flexível e escalonável para armazenamento e sincronização. A documentação oficial fornece os os passos para adicionar este BaaS, a um projeto Flutter. Mantém os dados em sincronia através de *listeners* atualizados em tempo real (FIREBASE, 2020c).

Na Figura 3.4 abaixo, encontra-se a relação de dados implementados no *Cloud Firestore*. Também foram criadas coleções de tipo e cômodo do dispositivo, para escolha do usuário durante cadastro e edição de dispositivo.

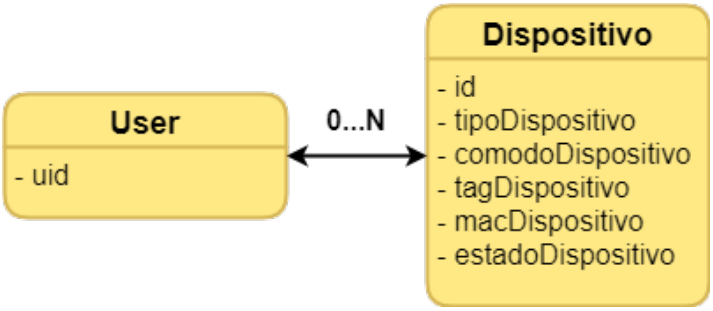


Figura 3.4 – Esquema de banco de dados.

Uma das formas de acesso aos dados no Cloud Firestore é utilizar o caminho do documento ou coleção. Na Figura 3.5 é mostrado um caminho de dados genérico. Coleções e documentos se alternam, sendo que coleções estão sempre em posições ímpares e documentos, em posições pares.

Figura 3.5 – Caminho de dados no Cloud Firestore.

Para o projeto, foi criada a classe *APIPath*, que é usada em *Database*. O caminho assim declarado facilita a implementação da rotina de criação, leitura, atualização e exclusão (CRUD) de dispositivos. No código abaixo, é mostrada a classe que direciona os caminhos definidos no Flutter diretamente no Firebase.

```
1 class APIPath{
2   static String dispositivo(String uid, String dispositivoID) =>
3   'users/$uid/dispositivos/$dispositivoID';
4   static String dispositivos(String uid) => 'users/$uid/dispositivos';
5 }
```

Código-fonte 3.1 – Classe APIPath.

Os *widgets* que realizam as operações CRUD não se comunicam diretamente com o serviço do Firestore. Para manter a camada de serviços separada das camadas de UI, foi criada a classe *FirestoreService*. Esta classe localiza as instâncias desejadas através de uma declaração *"instance = FirestoreService._()"*. A vantagem desta classe é a sua modularidade, caso seja necessário trocar o serviço de banco de dados.

Finalmente, foi criada a classe *Database* contendo os métodos *setDispositivo*, *deleteDispositivo* e *dispositivoStream*. Utilizando o caminho de *APIPath* e os métodos criados em *FirestoreService*, estas relações de dados estão expostas somente ao que se refere a dispositivos, não tendo acesso a outras instâncias do BD.

O uso de *Stream* para listar os dispositivos permite a atualização dos dados da aplicação para o Cloud Firestore em tempo real. No nível de interface, ao listar os dispositivos dentro de um *StreamBuilder*, a UI muda quando os dados são alterados.

Para que o que foi descrito acima funcione, comunicando a aplicação com o BD, foi necessário definir o modelo do dispositivo que se encontra na classe *Dispositivo* abaixo:

```
1 class Dispositivo{
2  Dispositivo({@required this.id, @required this.tipo, @required this.
     comodo, @required this.tag, this.mac, this.estado});
3  final String id;
4  final String tipo;
5  final String comodo;
6  final String tag;
7  final String mac;
8  final int estado;
```

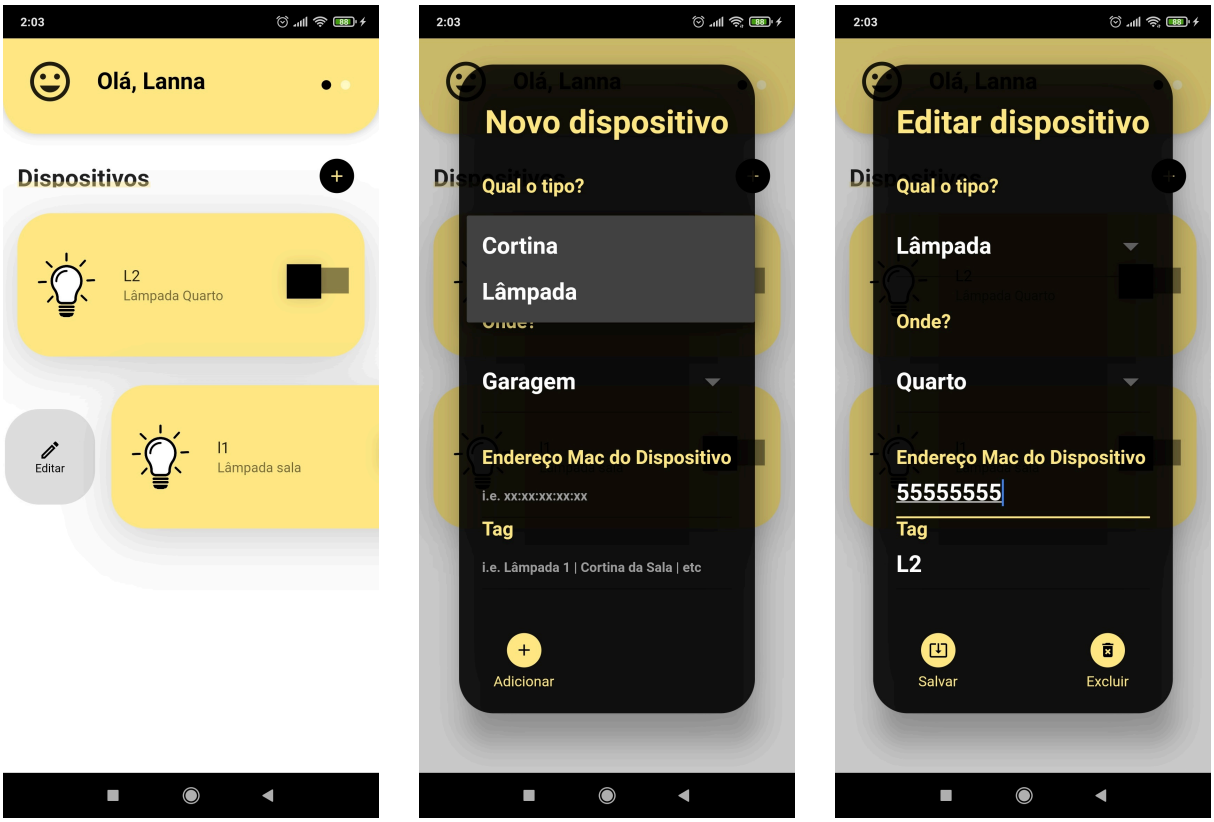
```
9   ... }
```

Código-fonte 3.2 – Dispositivo.

Para lidar com a entrada de dados do dispositivo, foi feita a UI na classe *AddDisp*. Dentro dela, um formulário é chamado para realizar o cadastro ou edição do dispositivo. Um formulário tem definido uma chave global que permite validar e salvar os dados inseridos no *widgets TextFormField* e *DropDownMenuItem*.

Este formulário é criado dentro de um *StatefulBuilder* aninhado em um *AlertDialog*.

A Figura 3.6 mostra a manipulação dos dados na UI. Em 4.1a, é exibida a página inicial *HomePg* direcionada a *BodyDispositivos*. Esta última contém um *StreamBuilder* do tipo Lista de Dispositivos, que exibe os *Cards* dos dispositivos cadastrados. Para cadastrar um dispositivo, basta clicar no ícone com símbolo de adição, e a interface exibe tela de 3.6b. Para editar ou excluir um dispositivo, como na tela de 3.6c, é necessário deslizar o *Card* para a direita e clicar em editar.



(a) Listagem. (b) Cadastro. (c) Edição.

Figura 3.6 – Manipulação dos dispositivos na UI.

A relação dos dados nas camadas de interface e de serviço pode ser vista na Figura 3.7.

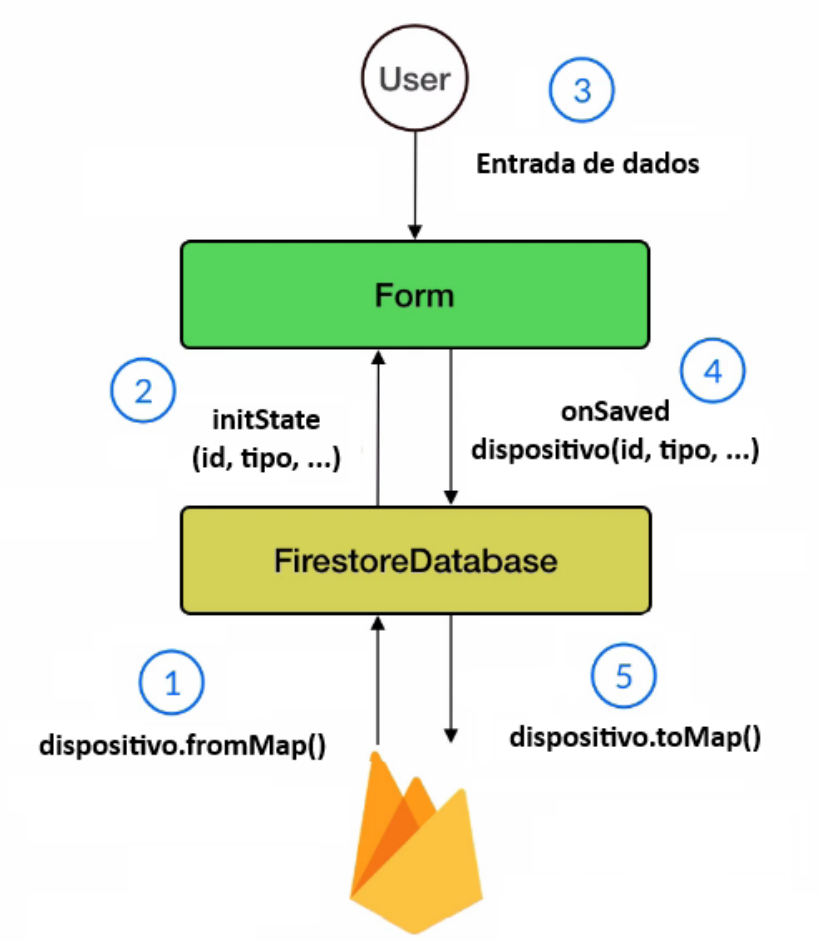


Figura 3.7 – Relação dos dados em UI e BD. Adaptado (BIZZOTTO, 2020)

Os passos que envolvem a manipulação do mapas (relação de chave e valor) de dispositivo estão enumerados:

1. **Ler** construtor de um mapa do dispositivo, a partir do modelo, no Cloud Firestore;
2. **Ler** estado inicial do dispositivo armazenado no BD;
3. **Escrever** entrada de dados do usuário via formulário;
4. **Escrever** objeto do modelo dispositivo é criado/editado;
5. **Escrever** conversão do modelo em um par de chave e valor no BD.

3.2.4 **Modo Voz**

Para desenvolvimento do modo de acionamento por voz, utilizou-se o pacote *Speech To Text*. Trata-se de uma biblioteca que possibilita a capacidade de reconheci-

mento de voz específica do dispositivo. Os casos de uso de destino para esta biblioteca são comandos e frases curtas, não incluindo conversação contínua ou sempre em escuta.

Este pacote utiliza a localização do dispositivo para definir o idioma de reconhecimento de voz padrão. Mas suporta qualquer idioma que esteja instalado no dispositivo (CSDCORP, 2021). Foi utilizado o pacote *Avatar Glow* que exibe uma animação no fundo de um *widget* (APGAPG, 2021).

Foi criada a classe *VoiceMode*, que alterna com *BodyDispositivos* no *PageView* de *HomePg*. Dentro de *VoiceMode* é chamada a classe *SpeechScreen*.

O parâmetro *confidence* mede o grau de precisão da resposta. O nível de confiança da solicitação de transcrição completa é indicado como um número entre 0,0 e 1,0 na resposta enviada. Para o trabalho, foi configurado em 0,8.

Algumas palavras foram reservadas em listas para realizar o acionamento: acender, acenda, ligar, ligue, apagar, apague, desligar e desligue. Foi designado um botão para ativar a entrada de áudio, com o método *_listen()*, e outro para finalizar a entrada de áudio, com o método *_execute()*.

Este último contém o método *changeState()*, que altera no Firebase o estado (ligado ou desligado, 1 ou 0) do dispositivo.

3.2.5 **Perfil, Configurações de Rede, e Logout**

Alguns requisitos funcionais da aplicação desenvolvida foram atribuídos em uma página denominada *AccountPg*, elaborada para mostrar informações de perfil, configurações de rede e *logout* da aplicação.

Uma parte de *AccountPg* é exibida em *HomePg*. *AccountPg* está aninhada em um *SlidingUpPanel*, *widget* verticalmente arrastável (JAIN, 2020).

3.2.5.1 **Perfil**

A opção personalizável pelo usuário disponível é a entrada de texto que define o nome do “morador(a)” na UI.

Como se trata uma entrada de texto cujo valor deve ser exibido em diferentes estados da tela do aplicativo, foi criada a classe *HeaderInfo* com *ChangeNotifier*. Esta

classe contém o *get* e *set* que recebe e modifica a entrada de texto, e também o método *notifyListeners()* para consolidar a alteração da entrada de texto.

Optou-se por utilizar o armazenamento persistente de dados simples para guardar o valor da entrada de texto. Significa que se o usuário desinstalar a aplicação, este valor também será excluído. Foi utilizado o pacote *Shared Preferences* nesta seção (FLUTTER, 2021d).

3.2.5.2 Configurações de Rede

Para configurar o dispositivo ESP32 *Gateway* (que será abordado na seção 3.3), é necessário ter acesso a alguns dados do usuário: identificação de usuário no Firebase (uid), identificação da rede wifi local (SSID) e a senha da rede local (pwd). Esta configuração é feita em duas telas, definidas nas classes *WifiInform* e *WifiInformEsp*.

A classe *WifiInfom* contém o *widget Connectivity*, que permite que o aplicativo descubra informações de rede do dispositivo (no caso, *smartphone* do usuário) (FLUTTER, 2021a). O pacote *Permission Handler* também foi utilizado pois é necessário conceder permissão de localização para usar o *Connectivity*. O *Permission Handler* viabiliza solicitações de permissões do usuário após a instalação do aplicativo, enquanto ele está em execução, para habilitar recursos específicos (BASEFLOW, 2021).

O usuário é orientado a inserir a senha da rede local, para que este dado seja transmitido ao ESP32 *Gatway/Servidor* e este seja conectado à internet.

A classe *WifiInformEsp* também verifica em qual rede o usuário está conectado, pois ele é orientado a se conectar na rede *InGateway*, definida como a rede do ESP32 *InGateway* em Modo AP. Além disso, recebe os dados uid, ssid e pwd via provider das respectivas bases (uid de *AuthBase*, ssid e pwd de *WifiInform*).

Ao clicar em “OK” o método *_sendToEsp()* é chamado. Este método pode ser visto no Código-fonte abaixo. Os parâmetros anteriormente citados nas configurações de rede são passados como pares de valor-chave para URI.

```
1 _sendToEsp() async {
2       var queryParameters = {
3         'uid' : getUid(),     //de AuthBase
4         'ssid': getSsid(),    //da rede local
5         'pwd': getPwd(),      //da rede local
6       };
```

```
7         var uri = Uri.http('192.168.0.143', 'setup', queryParameters);
8         var response = await http.get(uri);
9         if (response.statusCode == 200) {
10          //
11          return ;
12        } else {
13          throw Exception('Erro');
14        }
15      }
```

Código-fonte 3.3 – Método _sendToEsp().

3.2.5.3 *Logout*

Para sair da aplicação foi utilizado um provider de *AuthBase*, a mesma classe de serviço que define os métodos para autenticação dos usuários. Basta que o usuário toque em “Sair do InCasa”.

3.2.6 Permissões

Alguns *widgets* precisam acessar dados do usuário que não são acessíveis sem que este conceda permissões para o aplicativo. O modo como estas permissões são solicitadas depende do SO do dispositivo *mobile* do usuário.

O Firebase disponibiliza as instruções para as configurações de ambos Android e iOS para utilização de seus pacotes (FIREBASE, 2021a,b).

O pacote *Connectivity* requer acesso a internet, localização aproximada e localização precisa (FLUTTER, 2021a). O pacote *Speech To Text* requer permissões de acesso a internet, e gravação de áudio (CSDCORP, 2021).

No caso do Android, estão relacionadas no código-fonte abaixo (ANDROID, 2021).

```
1 <uses-permission android:name="android.permission.INTERNET"/>
2 <uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
3 <uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION"/>
4 <uses-permission android:name="android.permission.RECORD_AUDIO" />
```

Código-fonte 3.4 – Permissões solicitadas no Manifesto do Android.

3.3 Protótipo de Automação

Nesta seção são detalhados os procedimentos para realização do protótipo de automação. Os códigos-fonte do protótipo estão disponíveis em:

- *InCasa_ino* - Disponível em `https://github.com/lannagif/incasa_ino`.

3.3.1 **ESP32 InGateway**

As configurações de rede abordadas na seção 3.2.5.2 tem como base um dispositivo que atua entre o banco de dados da aplicação e o dispositivo de automação. Este dispositivo intermediário foi denominado ESP32 InGateway.

3.3.1.1 **Processo de configuração**

A Figura 3.8 mostra os passos programação do ESP InGateway na rotina de configuração, denominada *Setup*.

- **Inicialização Serial:** a taxa de transmissão de dados foi definida em 115200 *baud rate* (ARDUINO, 2021c).
- **Inicialização da EEPROM:** apesar do uso da biblioteca *EEPROM*, trata-se de uma memória *flash* e não de uma EEPROM real. Por ser uma memória não volátil, é tratada como tal, abstraindo da biblioteca citada. Contém a alocação da capacidade de memória não volátil dedicada ao armazenamento da estrutura de *Setup*, ssid, pwd e uid (SANTOS; SANTOS, 2021).
- **Verificação UID e SSID:** verifica se as variáveis uid e ssid de *Setup* já foram instanciadas/inicializadas no ESP. Caso não encontre, inicializa *Wifi Ap*. Caso encontre, busca o método para conectar no *Wifi Ap*.
- **Inicialização do Wifi Ap:** cria uma rede com nome “InGateway” para que o celular se conecte. Os testes foram realizados com a rede local para facilitar o processo de conexão.
  - **Aguardar *Setup*:** este método aguarda pela conexão do cliente no *Web Server*. Enquanto o cliente estiver conectado e disponível, uma variável

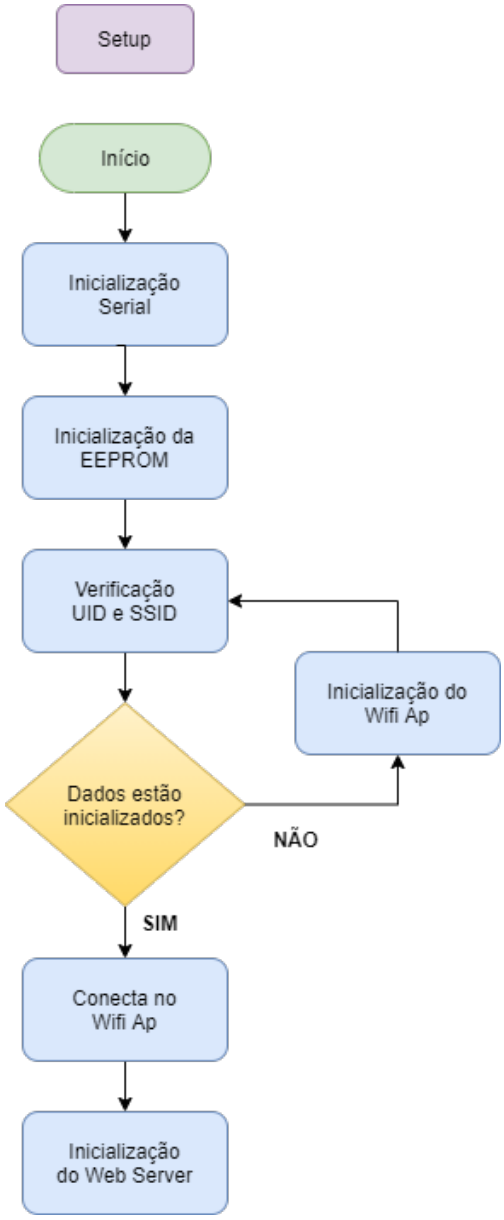


Figura 3.8 – Rotina de configuração do ESP 32 InGateway.

String é criada para reter dados de entrada. Uma variável *char* recebe a leitura do cliente. De acordo com esta chamada, direciona a chamada HTTP na requisição de *Setup* e mostra no *Serial Monitor* da *IDE* a resposta (ARDUINO, 2021a,d; ESPRESSIF, 2021b; MDN, 2021).

- **Conexão no Wifi Ap:** Conecta na rede local. Se as variáveis já estão inicializadas, o método *connect_to_ap()* exibe, no *Serial Monitor*, o endereço de ip da conexão.

3.3.1.2 **Firebase**

Depois de realizada a configuração que envolve o *Web Server*, é realizada uma configuração que permite a troca de dados entre o ESP InGateway e o Firebase. Para isso, foi utilizada a biblioteca *Firebase_ESP_Client* (MOBIZT, 2021).

Alguns parâmetros são necessários para criação do objeto de dados do Firebase: *FIREBASE_HOST*, *FIREBASE_PROJECT_ID*, *API_KEY* e *FIREBASE_CLIENT_EMAIL*. São encontrados no *console* do Firebase.

Inicialmente, ocorre uma verificação da existência do usuário. O caminho base para localização dos usuários é o mesmo definido na seção 3.2.2 Autenticação. Um caminho para a coleção de dispositivos também é definido no código do ESP32 InGateway.

A função *millis()* define a latência da resposta do sistema. De acordo com o tempo de leitura do estado, o *loop* do Firebase verifica os documentos (ARDUINO, 2021b).

A conversão dos dados do Firebase de e para JSON utiliza a estrutura & (ponteiro) (ARDUINO, 2021e).

A Figura 3.9 mostra a etapa de configuração do ESP32 InGateway feita com os parâmetros uid, ssid e pwd passados pelo navegador.

Figura 3.9 – Configuração InGateway via navegador.

A Figura 3.10 a seguir mostra a configuração e conexão de ESP32 InGateway via *_sendToEsp*, diretamente do aplicativo.

```
21:13:23.954 -> new client
21:13:23.954 -> GET /setup?uid=Ae3OgzRi6lezOqJEQhHdMZKRcne2&ssid=CASA&pwd=32261474 HTTP/ssid=>'CASA'
21:13:23.954 -> uid=>'Ae3OgzRi6lezOqJEQhHdMZKRcne2'
21:13:23.954 -> pwd=>'32261474'
21:13:23.954 -> Writing to EEPROM value:'CASA' => address:0
21:13:23.954 -> Writing to EEPROM value:'32261474' => address:1
21:13:23.954 -> Writing to EEPROM value:'Ae3OgzRi6lezOqJEQhHdMZKRcne2' => address:2
21:13:23.954 -> 1.1
21:13:23.954 -> user-agent: Dart/2.10 (dart:io)
21:13:23.954 -> accept-encoding: gzip
21:13:23.954 -> content-length: 0
21:13:23.954 -> host: 192.168.0.143
21:13:23.954 ->
21:13:23.954 -> client disonnected
21:15:02.367 -> Read from EEPROM value:'CASA' => addr: 0
21:15:02.367 -> Read from EEPROM value:'32261474' => addr: 1
21:15:02.367 -> Read from EEPROM value:'Ae3OgzRi6lezOqJEQhHdMZKRcne2' => addr: 2
21:15:02.367 -> Initilized
21:15:02.367 -> Values:
21:15:02.367 ->          Uid: Ae3OgzRi6lezOqJEQhHdMZKRcne2
21:15:02.367 ->          PWD: 32261474
21:15:02.367 ->          SSID: CASA
21:15:02.458 -> Connecting to Wi-Fi...
21:15:03.389 -> Connected with IP: 192.168.0.143
21:15:03.389 ->
```

Figura 3.10 – Configuração InGateway via *_sendToEsp* do aplicativo.

As imagens acima podem ser vistas em `https://github.com/lannagif/incasa_ino/tree/main/imgs`.

3.3.1.3 **ESP-NOW**

Optou-se pela utilização do protocolo *ESP-NOW* entre os ESPs do projeto.

Para que o ESP32 InGateway hospede tanto o Web Server quando o protocolo ESP-NOW o seguinte requisito deve ser atendido: os placas embarcadas, no caso ESP32 InGateway e ESP32 InWorker precisam estar no mesmo canal de Wifi.

O canal Wifi do ESP32 InGateway é atribuído automaticamente pelo seu roteador da rede local e funciona como ponto de acesso (*SoftAP*) e estação (*Station*).

3.3.2 **ESP32 InWorker**

A Figura 3.11 mostra os passos para programação do ESP32 InWorker.

- **Inicialização Serial:** a taxa de transmissão de dados foi definida em 115200 Bd/s (*baud rate* por segund) (ARDUINO, 2021c).
- **Configurações de Saída:** definição do pino que recebe o canal de sinal do relé.

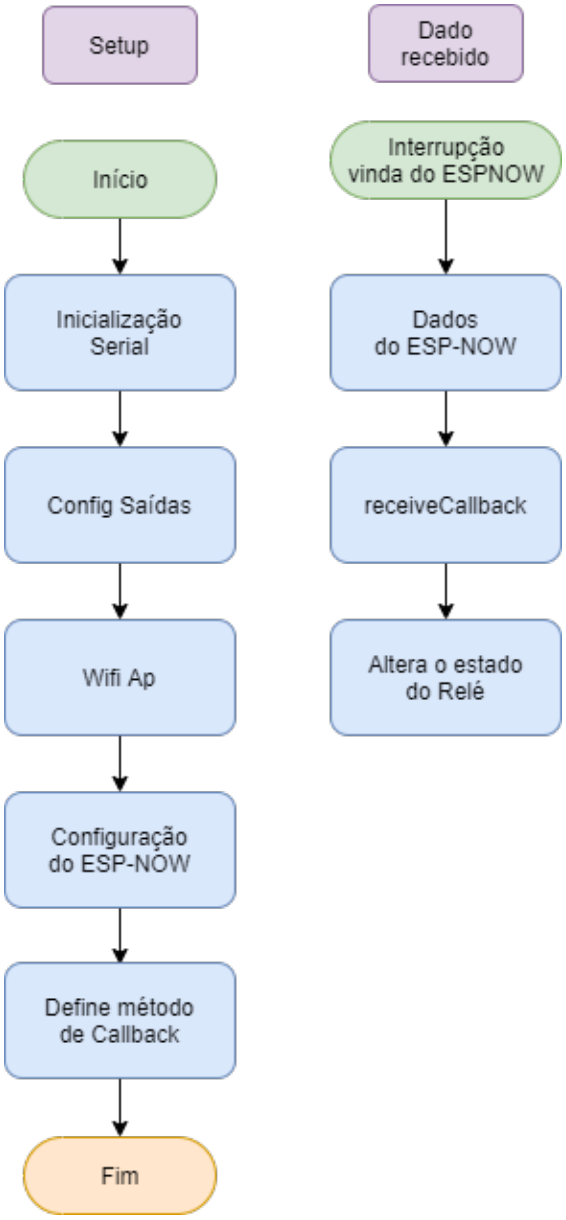


Figura 3.11 – Programação do ESP 32 InWorker.

- **Wifi Ap:** o ESP32 é colocado em modo *Acess Point*.
- **Configuração do ESP-NOW:** O ESP-NOW é inicializado. Contém *esp_now_register_recv_cb*, método que registra uma função de *Callback* (ESPRESSIF, 2021a).

O tratamento do dado recebido pelo ESP-InWorker se trata de uma função *Callback*.

- **Dados do ESP-NOW:** os dados são recebidos pelo protocolo.

- **receiveCallback:** responsável por atribuir o método que será chamado após interrupção do MCU.
- *Altera o estado do relé:* resultado esperado para o protótipo.

A Figura 3.12 mostra a saída no *Serial Monitor* para o ESP32 InWorker. Ao mudar o estado do dispositivo no aplicativo *InCasa*, estado passa para 1 e altera o estado do relé que está conectado à lâmpada e a rede elétrica. Vídeos do projeto podem ser vistos em: `http://bit.ly/tcc-lanna-vídeos`

```
21:08:01.242 -> Mode: AP
21:08:01.242 -> Channel: 1
21:08:01.242 -> SSID (4): CASA
21:08:01.242 -> Passphrase (8): 32261474
21:08:01.289 -> BSSID set: 0
21:08:01.289 -> AP MAC: FC:F5:C4:19:A5:F9
21:09:52.094 -> Packet received from: MAC Address: {0xFC,0xF5,0xC4,0x19,0xAC,0x8};
21:09:52.140 -> Destination: MAC Address: {0xFC,0xF5,0xC4,0x0,0x4,0x6C};
21:09:52.140 ->
21:09:52.140 -> Estado: 0
21:09:53.958 -> Packet received from: MAC Address: {0xFC,0xF5,0xC4,0x19,0xAC,0x8};
21:09:53.958 -> Destination: MAC Address: {0xFC,0xF5,0xC4,0x0,0x4,0x6C};
```

Figura 3.12 – Saída do ESP 32 InWorker.

3.4 Custos de Projeto

A Tabela 3.1 a seguir mostra os custos para a realização deste trabalho de conclusão de curso.

Tabela 3.1 – Tabela de custos do projeto (orçamento).

Item	Quantidade	Custo (R$)
Aplicativo		
Celular com SO Android	1	Da autora
Notebook com Softwares	1	Da autora
Eletrônica		
ESP 32	2	55,00
Módulo relé 4 canais	1	25,00
Lâmpada LED 9W	1	6,00
Soquete para Lâmpada	1	7,00
Cabos e conectores	n	Da autora
Multímetro	1	Da autora
Total		**148,00**

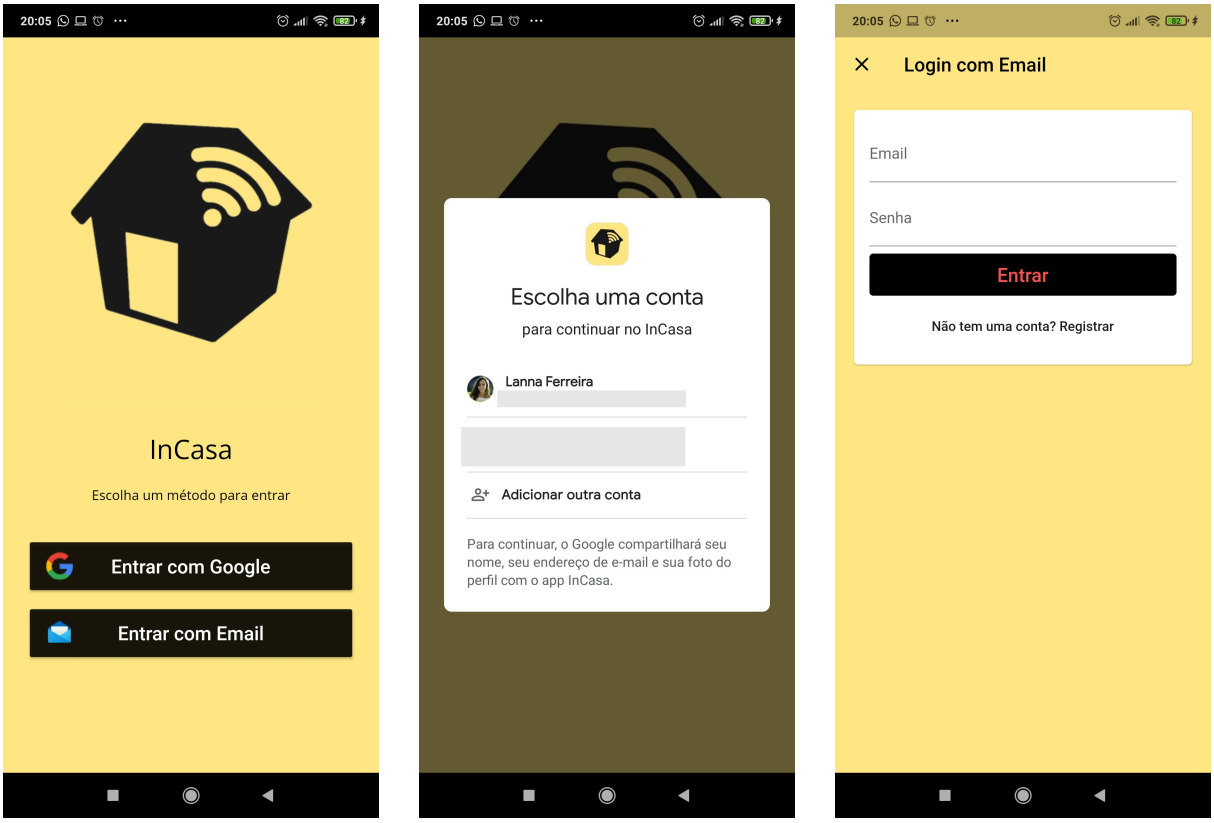
Capítulo 4

Resultados e Discussões

Neste capítulo são apresentados os resultados obtidos no decorrer do trabalho.

4.1 Aplicativo InCasa

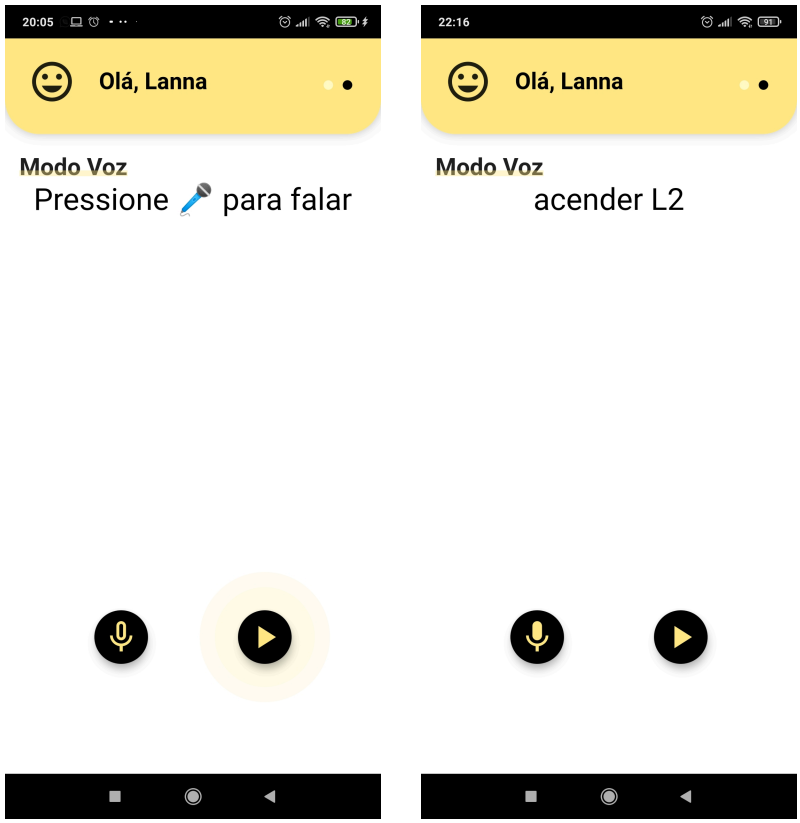
Na Figura 4.1 abaixo, são exibidas a tela inicial do InCasa e as telas com as opções de *login*: com o Google e com Email/Senha.



(a) Tela inicial. (b) Entrar com o Google. (c) Entrar com Email.

Figura 4.1 – Configurações de rede.

As telas de cadastro de dispositivos foram exibidas na seção 3.2.3. Na Figura 4.2 são exibidos o estado inicial do modo voz e a transcrição do áudio de entrada.

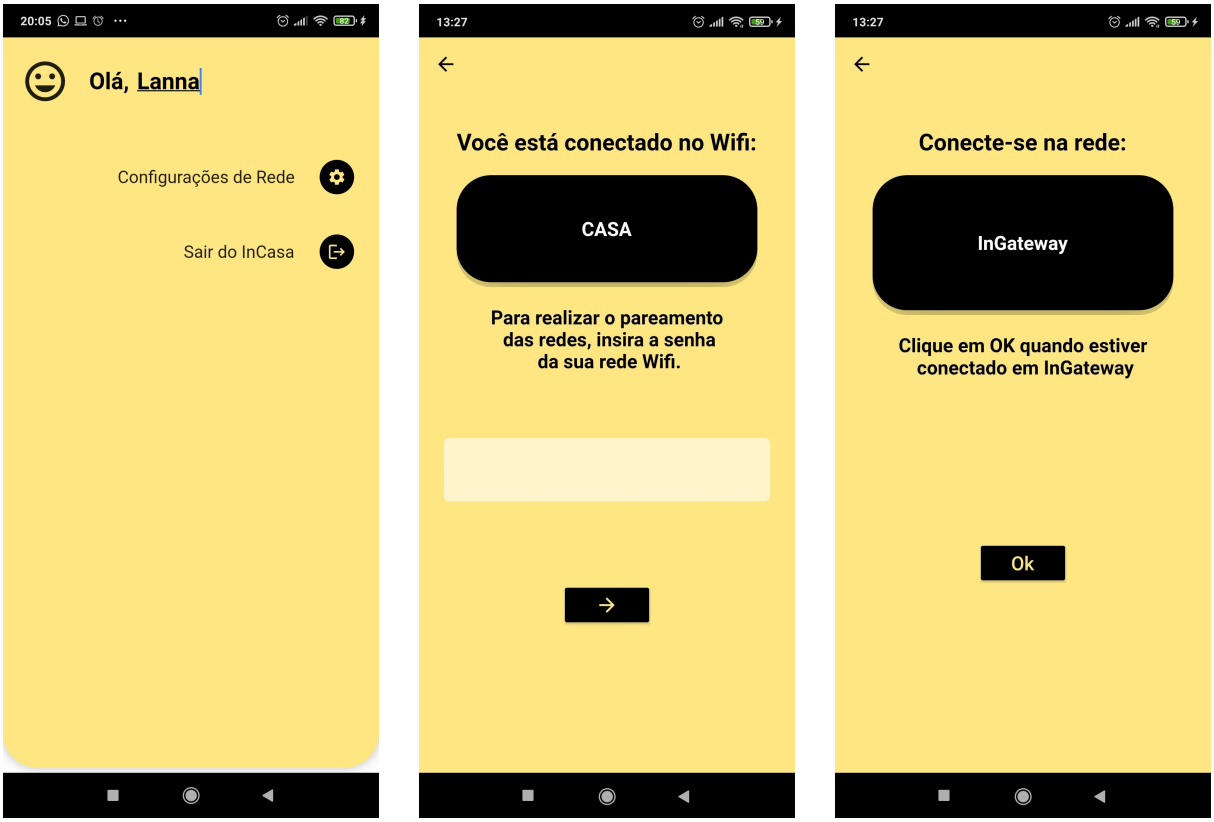


(a) Estado Inicial. (b) “Fala em texto”.

Figura 4.2 – Modo Voz.

Na Figura 4.3a é exibida a página de Perfil, Configurações de Rede, e Logout, abordada na seção 3.2.5.

Na Figura 4.3b é mostrada a primeira etapa de configuração do ESP-32 InGateway. E na Figura 4.3c a etapa onde o usuário confirma os dados e os envia ao *Webserver*.



(a) Conta. (b) Configurações 1. (c) Configurações 2.

Figura 4.3 – Perfil, Configurações de Rede, e Logout.

4.2 Protótipo de Automação

O funcionamento do projeto pode ser visto em:

- `http://bit.ly/TCC_II_Lanna`

Na Figura 4.4 encontra-se o protótipo desenvolvido.

Figura 4.4 – Protótipo desenvolvido.

4.3 Discussões

Um detalhe importante refere-se ao custo dos serviços. Para o projeto, a fim de minimizar o tempo entre o acionamento pelo usuário e a mudança de estado do relé, o intervalo de leituras do Cloud Firestore foi configurado para 500ms.

Assim, uma resposta *rápida* é possível. Considerando que em um dia $2 \times 60 \times 60 \times 24 = 172,8 \times 10^3$ seja a média esperada de requisições. O limite gratuito atual para leituras diárias do Firebase é 50 mil. Ao ultrapassar esse valor, é cobrado US$ $0,06/100$ mil.

Capítulo 5

Considerações finais

Neste capítulo são feitas considerações finais sobre o desenvolvimento deste trabalho de conclusão de curso.

5.1 Conclusões

Este trabalho teve como objetivo o projeto de sistema de automação residencial comandado por aplicativo, com função de reconhecimento por voz. O primeiro passo para sua realização foi a pesquisa, levantando informações sobre origem, demanda e perspectivas gerais sobre o tema. Percebeu-se como a integração de tecnologias para interação com o domicílio pode trazer autonomia para pessoas com mobilidade reduzida, em um sentido de tecnologia assistiva.

Quanto à funcionalidade de comando via reconhecimento de voz, foi possível implementála ao projeto alcançando o objetivo de direcionar o acionamento de dispositivos pelo aplicativo, com a ressalva de que os dispositivos sejam cadastrados com uma "tag" de única palava.

Em relação às ferramentas e métodos, o projeto levou a um estudo prático do Flutter, que se mostrou uma ótima ferramenta para desenvolvimento *mobile*. O uso do *Cloud Firestore* aplicado ao protótipo levantou a necessidade de uma maior pesquisa sobre o custo x benefício desta ferramenta.

Ao acionar o dispositivo, a latência da rede local influencia o tempo de resposta. O que gera essa latência é, em grande parte, a função *millis*, o ruído das redes WiFi e o canal o qual o ESP32 InGateway é conectado.

O projeto desenvolvido pode servir de ponto de partida para projetos onde o cliente busca tais funcionalidades. Mas maiores estudos e testes são necessários para que este seja caracterizado um produto final.

5.2 Proposta de Continuidade

Como proposta de continuidade para o projeto, é sugerido implementar novos tipos de dispositivos. Diante das ferramentas do banco de dados, é possível também gerar relatórios de gerenciamento de dispositivos e de utilização dos usuários.

Também é possível investigar os perfis de utilização e utilizar aprendizado de máquina para analisar o perfil do usuário, levantar rotinas do uso de dispositivos e informar ao usuário informações como consumo energético eficiente.

Referências

ALIVE, MIT Media Lab. **Aritificial Life Interactive Video Environment (ALIVE)**. 1995. Disponível em: <https://vismod.media.mit.edu/vismod/demos/smartroom/>.

ALOI, G. et al. Enabling IoT interoperability through opportunistic smartphone-based mobile gateways. **Journal of Network and Computer Applications**, Elsevier, v. 81, July, p. 74–84, 2017. ISSN 10958592. DOI: 10.1016/j.jnca.2016.10.013. Disponível em: <http://dx.doi.org/10.1016/j.jnca.2016.10.013>.

ANDROID. **Android Manifest Permissions**. 2021. Disponível em: <https://developer.android.com/reference/android/Manifest.permission>.

_____. **Android Studio**. 2020. Disponível em: <https://developer.android.com/studio/intro>.

APGAPG. **Avatar Glow**. 2021. Disponível em: <https://pub.dev/packages/avatar_glow>.

ARDUINO. **Arduino**. 2020. Disponível em: <https://www.arduino.cc/en/software>.

_____. **Arduino Data Type Char**. 2021. Disponível em: <https://www.arduino.cc/reference/en/language/variables/data-types/char/>.

_____. **Arduino Function Millis**. 2021. Disponível em: <https://www.arduino.cc/reference/en/language/functions/time/millis/>.

_____. **Arduino Serial Begin**. 2021. Disponível em: <https://www.arduino.cc/reference/en/language/functions/communication/serial/begin/>.

_____. **Arduino Serial Write**. 2021. Disponível em: <https://www.arduino.cc/reference/en/language/functions/communication/serial/write/>.

ARDUINO. **Arduino Structure &**. 2021. Disponível em: <https://www.arduino.cc/reference/en/language/structure/pointer-access-operators/reference/>.

ASTHON, Kevin. That ' Internet of Things ' Thing. **RFID Journal**, p. 4986, 2010. ISSN 00280836. Disponível em: <http://www.rfidjournal.com/article/print/4986>.

BASEFLOW. **Permission Handler**. 2021. Disponível em: <https://pub.dev/packages/permission_handler>.

BERSCH, Rita. **Tecnologia Assistiva SNPD**. [S.l.: s.n.], 2017. P. 20. Disponível em: <http://www.pessoacomdeficiencia.gov.br/app/publicacoes/tecnologia-assistiva>.

BIZZOTTO, Andrea. **Flutter & Firebase: Build a Complete App for iOS & Android**. 2020. Disponível em: <https://www.udemy.com/course/flutter-firebase-build-a-complete-app-for-ios-android/>.

BOWDEN, Sue; OFFER, Avner. Household Appliances and the Use of Time: The United States and Britain Since the 1920s. **The Economic History Review**, [Economic History Society, Wiley], v. 47, n. 4, p. 725–748, 1994. ISSN 00130117, 14680289. Disponível em: <http://www.jstor.org/stable/2597714>.

BRASIL. Lei nº 13.146:2015. **Diário Oficial da União**, v. 127, n. 1, p. 2, 2015. Disponível em: <https://legislacao.presidencia.gov.br/atos/?tipo=LEI%7B%5C&%7Dnumero=13146%7B%5C&%7Dano=2015%7B%5C&%7Dato=c4aUTW65UNVpWT495>.

BRODY, Paul; PURESWARAN, Veena. **Device democracy: Saving the future of the Internet of Things - IBM Global Business Services Executive Report**. New York, 2015. P. 3–25. Disponível em: <http://m2mworldnews.com/download/white-papers/IBM-Saving-the-future-of-IoT.pdf>.

CALVELLO, Mara. **A Complete History of Computers: From the 1800s to Now**. 2019. Disponível em: <https://learn.g2.com/history-of-computers#computers-present-day>.

CASEY, Michael A; GARDNER, William G; BASU, Sumit. Vision Steered Beam-forming and Transaural Rendering for the Arti cial Life Interactive Video Environment , (ALIVE) 1 Vision Steered Beam-Forming. **Artificial Life**, p. 1–28, 1995. Disponível em: <http://alumni.media.mit.edu/%7B~%7Dsbasu/papers/aes95.pdf>.

CORDEIRO, Giovanni; BAGGIO, Cláudia; FRANÇA, Luiz Antonio. Comportamento do consumidor de imóveis em 2040. **ABRAINC e Deloitte**, 2019. Disponível em: <https://www.abrainc.org.br/wp-content/uploads/2019/09/Abrainc-Pesquisa-v10.pdf>.

CSDCORP, Corner Software Development Corp. **Speech To Text**. 2021. Disponível em: <https://pub.dev/packages/speech_to_text>.

DART. **Dart Documentation**. 2020. Disponível em: <https://dart.dev/>.

DRAW IO. **Draw Io**. 2020. Disponível em: <https://app.diagrams.net/>.

ELETROGATE. **Módulo Relé 4 Canais 5v com Optoacoplador**. 2020. Disponível em: <https://www.eletrogate.com/modulo-rele-4-canais-5v>.

ESPRESSIF. **ESP-NOW**. 2021. Disponível em: <https://docs.espressif.com/projects/esp-idf/en/stable/esp32/api-reference/network/esp_now.html>.

_____. **ESP32 HTTP Server**. 2021. Disponível em: <https://docs.espressif.com/projects/esp-idf/en/latest/esp32/api-reference/protocols/esp_http_server.html>.

EXPRESSIF. **ESP32-WROOM-32 Datasheet**. Jan. 2019. Disponível em: <https://www.espressif.com/en/products/socs/esp32>.

FALL, Kevin R; STEVENS, W Richard. **TCP/IP Illustrated, Volume 1: The Protocols**. 2 ed. [S.l.]: Addison-Wesley, Pearson, 2012. P. 80–84. ISBN 0-321-33631-3, 978-0-321-33631-6, 9780201633467, 0201633469, 9780201633542, 020163354X, 9780201634952, 0201634953.

FILIPEFLOPE. **ESP 32 Filipeflope**. 2020. Disponível em: <https://www.filipeflop.com/produto/modulo-wifi-esp32-bluetooth/>.

FIREBASE. **Firebase**. 2020. Disponível em: <https://firebase.google.com/>.

_____. **Firebase Android**. 2021. Disponível em: <https://firebase.google.com/docs/cloud-messaging/android/client?hl=pt-Br>.

_____. **Firebase Authentication**. 2020. Disponível em: <https://firebase.flutter.dev/docs/auth/usage>.

FIREBASE. **Firebase Cloud Firestore**. 2020. Disponível em: <https://firebase.flutter.dev/docs/firestore/overview>.

_____. **Firebase IOS**. 2021. Disponível em: <https://firebase.google.com/docs/cloud-messaging/ios/client?hl=pt-Br>.

FLUID UI. **Fluid UI**. 2020. Disponível em: <https://www.fluidui.com/>.

FLUTTER. **Connectivity**. 2021. Disponível em: <https://pub.dev/packages/connectivity>.

_____. **Flutter Documentation**. 2020. Disponível em: <https://flutter.dev/>.

_____. **Flutter Install**. 2021. Disponível em: <https://flutter.dev/docs/get-started/install>.

_____. **Flutter IOS**. 2021. Disponível em: <https://flutter.dev/docs/get-started/flutter-for/ios-devs>.

_____. **Shared Preferences**. 2021. Disponível em: <https://pub.dev/packages/shared_preferences>.

IBGE. **Pesquisa Nacional por Amostra de Domicílios Contínua - PNAD Contínua 2018 - Acesso à Internet e à Televisão e Posse de Telefone Móvel Celular para Uso Pessoal**. [S.l.: s.n.], 2018. P. 1–146. Disponível em: <https://www.ibge.gov.br/estatisticas/multidominio/ciencia-tecnologia-e-inovacao/17270-pnad-continua.html?=%7B%5C&%7Dt=downloads>.

ISLAM, Nayeem; WANT, Roy. Smartphones: Past, Present, and Future. **IEEE Pervasive Computing**, v. 13, n. 4, p. 89–92, out. 2014. ISSN 1536-1268. DOI: 10.1109/MPRV.2014.74. Disponível em: <http://ieeexplore.ieee.org/document/6926722/>.

ISO 7498-1:1994. v. 2000. Geneva, CH, jun. 1996.

JAIN, Akshath. **Sliding Up Panel**. 2020. Disponível em: <https://pub.dev/packages/sliding_up_panel>.

JONES, Owen Daly; CAREY, Rachel. Interactive TV. In: CHI '00 extended abstracts on Human factors in computing systems - CHI '00. New York, New York, USA: ACM Press, 2000. P. 306. ISBN 1581132484. DOI: 10.1145/633292.633475. Disponível em: <http://delivery.acm.org/10.1145/640000/633475/p306-daly-jones.pdf?ip=157.82.4.178%7B%5C&%7Did=633475%7B%5C&%7Dacc=ACTIVE%20SERVICE%7B%5C&%7Dkey=D2341B890AD12BFE.925D03907F1B22EF.4D4702B0C3E38B35.4D4702B0C3E38B35%7B%5C&%7DCFID=945796695%7B%5C&%7DCFTOKEN=41044110%7B%5C&%7D%7B%5C_%7D%7B%5C_%7Dacm%7B%5C_%7D%7B%5C_%7D=1496910052%7B%5C_%7Dc1b12db8ad698e1df00b6%20http://portal.acm.org/citation.cfm?doid=633292.633475>.

KOSKELA, Tiiu; VÄÄNÄNEN-VAINIO-MATTILA, Kaisa. Evolution towards smart home environments: empirical evaluation of three user interfaces. **Personal and Ubiquitous Computing**, v. 8, n. 3-4, p. 234–240, jul. 2004. ISSN 1617-4909. DOI: 10.1007/s00779-004-0283-x. Disponível em: <http://link.springer.com/10.1007/s00779-004-0283-x>.

KURNIAWAN, Agus. **Internet of Things Projects with ESP32: Build exiting and powerful IoT projects using the all-new Expresif ESP32**. 1. ed. [S.l.]: Packt Publishing, 2019. (1290319). ISBN 978-1-78995-687-0. Disponível em: <http://gen.lib.rus.ec/book/index.php?md5=7f42216d1545d2ea2b5c48c01639d2ce>.

LARREA, Hasier; LARSON, Kent. **Project CityHome**. 2016. Disponível em: <https://www.media.mit.edu/projects/OLD_cityhome2/overview/>.

LARREA-TAMAYO, Hasier. **ARkits : Architectural Robotics Kits**. 2015. F. 2–124. Tese (Doutorado) – Massachusetts Institute of Technology. Disponível em: <https://dspace.mit.edu/handle/1721.1/98627>.

LIU, Na; YU, Ruifeng. Identifying design feature factors critical to acceptance and usage behavior of smartphones. **Computers in Human Behavior**, Elsevier B.V., v. 70, p. 131–142, mai. 2017. ISSN 07475632. DOI: 10.1016/j.chb.2016.12.073. Disponível em: <https://linkinghub.elsevier.com/retrieve/pii/S0747563216309013>.

LOGAN, Robert J et al. Living Room Culture—An Anthropological Study of Television Usage Behaviors. **Proceedings of the Human Factors and Ergonomics Society**

Annual Meeting, v. 39, n. 5, p. 326–330, out. 1995. ISSN 2169-5067. DOI: 10.1177/154193129503900507. Disponível em: <http://journals.sagepub.com/doi/10.1177/154193129503900507>.

MAJUMDER, Sumit et al. Smart Homes for Elderly Healthcare—Recent Advances and Research Challenges. **Sensors**, v. 17, n. 11, p. 2496, out. 2017. ISSN 1424-8220. DOI: 10.3390/s17112496. Disponível em: <http://www.mdpi.com/1424-8220/17/11/2496>.

MDN, Web Docs. **Uma visão geral do HTTP**. 2021. Disponível em: <https://developer.mozilla.org/pt-BR/docs/Web/HTTP/Overview>.

MEADOWS-KLUE, Danny. Inside the Smart Home. **Interactive Marketing**, v. 5, n. 3, p. 307–308, jan. 2004. ISSN 1463-5178. DOI: 10.1057/palgrave.im.4340249. Disponível em: <http://link.springer.com/10.1057/palgrave.im.4340249>.

MEHBOOB, Usama; ZAIB, Qasim; USAMA, Chaudhry. Survey of IoT Communication Protocols Techniques, Applications, and Issues. **Flow Research Inc, Pakistan**, p. 1–80, 2016. Disponível em: <http://xflowresearch.com/wp-content/uploads/2016/02/Survey-of-IoT-Communication-Protocols.pdf>.

MOBIZT. **Firebase Arduino Client Library for ESP8266 and ESP32**. 2021. Disponível em: <https://github.com/mobizt/Firebase-ESP-Client>.

MURATORI, José Roberto; DAL BÓ, Paulo Henrique. Capítulo I Automação residencial: histórico, definições e conceitos. **O setor elétrico**, v. 62, p. 70–77, 2011. Disponível em: <http://static2.voltimum.com/sites/www.voltimum.com.br/files/pdflibrary/04%7B%5C_%7Dautomacao%7B%5C_%7Dresidencial1.pdf>.

OMS. Lista de Produtos Assistivos Prioritários. **Organização Mundial da Saúde**, p. 1–12, 2017.

PENTLAND, A. Smart rooms, smart clothes. In: PROCEEDINGS. Fourteenth International Conference on Pattern Recognition (Cat. No.98EX170). [S.l.]: IEEE Comput. Soc, 1998. v. 2, p. 949–953. ISBN 0-8186-8512-3. DOI: 10.1109/ICPR.1998.711845. Disponível em: <http://ieeexplore.ieee.org/document/711845/>.

PEW RESEARCH CENTER DATA. **Internet, social media use and device ownership in U.S**. Set. 2018. Disponível em: <https://www.pewresearch.org/fact-tank/2018/09/28/internet-social-media-use-and-device-ownership-in-u-s-have-plateaued-after-years-of-growth/?utm_source=CNN+Media%5C%3A+Reliable+Sources%5C&utm_campaign=a00bd02216-EMAIL_CAMPAIGN_2018_09_11_04_47_COPY_01%5C&utm_medium=email%5C&utm_term=0_e95cdc16a9-a00bd02216-82344741>.

ROMKEY, John. Toast of the IoT: The 1990 Interop Internet Toaster. **IEEE Consumer Electronics Magazine**, v. 6, n. 1, p. 116–119, jan. 2017. ISSN 2162-2248. DOI: 10.1109/MCE.2016.2614740. Disponível em: <http://ieeexplore.ieee.org/document/7786805/>.

SANTOS, Bruno P; SILVA, Lucas A M et al. Internet das coisas: da teoria à prática. In: MINICURSOS SBRC-Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuıdos. v. 31. ed. Belo Horizonte: [s.n.], 2016. cap. 1, p. 50. Disponível em: <https://homepages.dcc.ufmg.br/%7B~%7Dmmvieira/cc/papers/internet-das-coisas.pdf>.

SANTOS, Rui; SANTOS, Sara. **ESP32 Flash Memory**. 2021. Disponível em: <https://randomnerdtutorials.com/esp32-flash-memory/>.

STATISA. **Mobile operating systems' market share worldwide from January 2012 to July 2020**. 2020. Disponível em: <https://www.statista.com/statistics/272698/global-market-share-held-by-mobile-operating-systems-since-2009/>.

TECHNOLOGY, Handson. User Guide - 4 Channel 5V Optical Isolated Relay Module. **Occupational Health & Safety**, v. 74, n. 2, p. 24, 2015. ISSN 03624064. Disponível em: <http://search.ebscohost.com/login.aspx?direct=true%7B%5C&%7Ddb=bth%7B%5C&%7DAN=16274161%7B%5C&%7Dsite=ehost-live>.

TUCIC, Milan et al. Networking layer for unifying distributed smart home entities. In: 2014 22nd Telecommunications Forum Telfor (TELFOR). [S.l.]: IEEE, nov. 2014. P. 368–371. ISBN 978-1-4799-6191-7. DOI: 10.1109/TELFOR.2014.7034426. Disponível em: <http://ieeexplore.ieee.org/document/7034426/>.

VELOCITY WERX. **WHAT WAS YOUR FIRST CELLPHONE?** 2018. Disponível em: <https://www.velocitywerx.com/news/2018/6/14/what-was-your-first-cellphone>.

VOICEBOT. **Voice Platform Impact Rating**. Mai. 2020. Disponível em: <https://voicebot.ai/2020/05/11/voice-industry-professionals-say-amazon-alexa-is-having-the-biggest-impact-followed-by-google-with-everyone-else-far-behind-new-report/>.