UNIVERSIDADE FEDERAL DO CEARÁ

CAMPUS CRATEÚS

CURSO DE GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

FRANCISCO DAVID NASCIMENTO SOUSA

DESENVOLVIMENTO DE UM LEITOR DE TELA PARA DEFICIENTES VISUAIS NO SISTEMA OPERACIONAL ANDROID

CRATEÚS

2023

FRANCISCO DAVID NASCIMENTO SOUSA

DESENVOLVIMENTO DE UM LEITOR DE TELA PARA DEFICIENTES VISUAIS NO SISTEMA OPERACIONAL ANDROID

Trabalho de Conclusão de Curso apresentado ao Curso de Graduação em Ciência da Computação do Campus Crateús da Universidade Federal do Ceará, como requisito parcial à obtenção do grau de bacharel em Ciência da Computação.

Orientador: Prof. Me. Francisco Anderson de Almada Gomes

CRATEÚS

2023

Dados Internacionais de Catalogação na Publicação
Universidade Federal do Ceará
Sistema de Bibliotecas
Gerada automaticamente pelo módulo Catalog, mediante os dados fornecidos pelo(a) autor(a)

S696d Sousa, Francisco David Nascimento.
Desenvolvimento De Um Leitor De Tela Para Deficientes Visuais No Sistema Operacional Android Crateús 2023 / Francisco David Nascimento Sousa. – 2023.
72 f. : il. color.

Trabalho de Conclusão de Curso (graduação) – Universidade Federal do Ceará, Campus de Crateús, Curso de Ciência da Computação, Crateús, 2023.
Orientação: Prof. Me. Francisco Anderson de Almada Gomes.

1. Acessibilidade Digital. 2. Ferramenta Assistida. 3. TalkBack. 4. Desenvolvimento Acessível. 5. Leitor de Tela. I. Título.

CDD 004

FRANCISCO DAVID NASCIMENTO SOUSA

DESENVOLVIMENTO DE UM LEITOR DE TELA PARA DEFICIENTES VISUAIS NO SISTEMA OPERACIONAL ANDROID

Trabalho de Conclusão de Curso apresentado ao Curso de Graduação em Ciência da Computação do Campus Crateús da Universidade Federal do Ceará, como requisito parcial à obtenção do grau de bacharel em Ciência da Computação.

Aprovada em: 29 de Julho de 2023

BANCA EXAMINADORA

Prof. Me. Francisco Anderson de Almada Gomes (Orientador)
Universidade Federal do Ceará (UFC)

Prof. Dr. Agebson Rocha Façanha
Instituto Federal do Ceará (IFCE)

Prof. Me. Ítalo Mendes da Silva Ribeiro
Universidade Federal do Ceará (UFC)

Ma. Vitória Regina Nicolau Silvestre
Fiserv Brasil

À minha família, por sua capacidade de acreditar e investir em mim. Mãe, seu cuidado e dedicação foi que deram, em alguns momentos, a esperança para seguir. Pai, sua presença significou segurança e certeza de que não estou sozinho nessa caminhada.

AGRADECIMENTOS

Ao Prof. Me. Anderson de Almada por me acompanhar e orientar em meu trabalho de conclusão de curso.

Ao Prof. Dr. Tobias Rafael Fernandes Neto, coordenador do Laboratório de Sistemas Motrizes (LAMOTRIZ) onde este *template* foi desenvolvido.

Ao Doutorando em Engenharia Elétrica, Ednardo Moreira Rodrigues, e seu assistente, Alan Batista de Oliveira, aluno de graduação em Engenharia Elétrica, pela adequação do *template* utilizado neste trabalho para que o mesmo ficasse de acordo com as normas da biblioteca da Universidade Federal do Ceará (UFC).

À Prof. Me. Lisieux Marie Marinho dos Santos Andrade por ter repassado um pouco do seu conhecimento para os alunos na disciplina de Projeto de Pesquisa Científica e Tecnológica (PPCT), conhecimento esse que teve grande importância na construção do presente trabalho.

À banca examinadora, em nome do Prof. Dr. Agebson Façanha, Prof. Me. Ítalo Ribeiro e da Ma. Vitória Silvestre pelo tempo dedicado a leitura desse trabalho, e também pelas contribuições e correções que aumentaram o valor do presente trabalho.

Ao meu pai Francisco das Chagas Oliveira Sousa e minha mãe Iramir Soares do Nascimento, por terem prestado todo apoio necessário, tanto no âmbito acadêmico, como na minha vida pessoal. Eles nunca deixaram de acreditar em mim, independente da situação. Além disso, gostaria de agradecer minha tia Marial Natilde do Nascimento, por compartilhar moradia comigo durante a graduação.

Aos amigos que estiveram comigo durante a graduação, por terem compartilhadas opiniões e experiências. Além disso, agradecer também por todos aqueles que estiveram presentes comigo em grupos de trabalho durante a graduação. Além do mais, gostaria de agradecer também aos meus colegas de trabalho que contribuíram bastante para minha formação profissional e pessoal.

À Prof. Ma. Lílian de Oliveira Carneiro por ter me proporcionado a primeira experiência de ensino dentro da universidade. Além disso, queria agradecer ao Prof. Me. André Meireles de Andrade por ter estado e contribuído durante 2 anos da minha formação profissional e pessoal.

Agradecer também a todos aqueles, tanto no contexto da universidade, quanto fora que de alguma forma contribuíram para minha formação.

"O homem não teria alcançado o possível se, repetidas vezes, não tivesse tentado o impossível."

(Max Weber)

RESUMO

É bastante notório que o uso de dispositivos móveis cresceu de forma substancial nos últimos anos, principalmente em decorrência da evolução de *hardware* e *software*. Nesse sentido, as atividades do dia a dia foram adaptadas completa ou de forma parcial para os *smartphones*, e essas atividades são feitas através de aplicações, que cresceram em quantidade tanto quanto os *smartphones*. Soluções móveis são usadas para tarefas simples como lembrete de remédios, assistir vídeos e ouvir música. Além disso, o uso para entretenimento como jogos e plataformas de conteúdos pagos também ganharam destaque. No entanto, há usos em operações mais complexas e delicadas como pagamentos e transferências bancárias, ou até mesmo soluções relacionadas ao monitoramento de pessoas através de plataformas, que fazem uso de sensores. Apesar dessa popularização, nem todas as aplicações são acessíveis para pessoas com deficiência, e para resolver esses problemas surgiram as tecnologias assistivas. Os usuários com deficiências visuais possuem problemas ao usar as ferramentas, como por exemplo: verbosidade da leitura, consciência e localização dentro das aplicações. Além disso, algumas ferramentas apresentam inúmeros problemas, o que acaba gerando o não uso das mesmas, ou quando utilizadas, os desenvolvedores empregam apenas funcionalidades básicas como leitura de texto, que pode ser inserida sem necessariamente criar uma implementação. Esse trabalho foca no desenvolvimento de uma ferramenta de leitura assistida para o sistema operacional *Android*, focado em problemas encontrados na literatura e entrevistas. Foram analisadas as ações de: *Verbosidade das Informações, Inserção de Textos, Copiar e Colar, Feedback em Caixas de Texto*, e foi possível concluir que qualquer funcionalidade deve ser adicionada com o máximo de customização e o mais configurável possível, além de não atrapalhar na autonomia e usabilidade da aplicação, ou seja, não adicionar opções obrigatórias, sempre deixando usuário escolher o que ele deseja usar e quando.

Palavras-chave: Accessibilidade. *Smartphone*. TalkBack

ABSTRACT

It is quite noticeable that the use of mobile devices has grown substantially in recent years, mainly due to the evolution of hardware and software. In this sense, daily activities have been fully or partially adapted to smartphones, and these activities are carried out through applications, which have grown in quantity as much as smartphones. Mobile solutions are used for simple tasks such as medication reminders, watching videos, and listening to music. In addition, the use of mobile devices for entertainment, such as games and paid content platforms, has also gained prominence. However, there are uses in more complex and delicate operations, such as payments and bank transfers, or even solutions related to people monitoring through platforms that use sensors. Despite this popularization, not all applications are accessible, and to address these issues, accessibility technologies have emerged. However, users with visual impairments face issues when using these tools, such as verbosity of reading, awareness, and location within applications. In addition, some tools present numerous problems, resulting in non-usage or only basic functionalities being employed, such as text-to-speech, which can be inserted without necessarily creating an implementation. This work focuses on developing an assisted reading tool for the Android operating system, focusing on issues found in the literature and interviews. It was analyzed regarding the *Verbosity of Information, Text Input, Copy and Paste, and Feedback in Text Boxes*, and it was possible to conclude that any functionality should be added with maximum customization and configurability without hindering the autonomy and usability of the application, i.e., not adding mandatory options, always allowing the user to choose what they want to use and when.

Keywords: Accessibility. Smartphones. TalkBack

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

API	Application Programming Interface
CLI	Command Line Interface
GUI	Graphical User Interface
HTML	HyperText Markup Language
IoT	*Internet of Things*
PoC	Proof of Concept
UI	User Interface
W3C	World Wide Web Consortium
WCAG	Web Content Accessibility Guidelines
Web	World Wide Web
ABNT	Associação brasileira de normas técnicas
AVC	ataque vascular celebral
OMS	Organização mundial da saúde

SUMÁRIO

1 INTRODUÇÃO

1.1 Contextualização

Nos últimos anos a demanda por desenvolvimento móvel tem se tornado crescente, isso pode ser observado na quantidade de aplicações móveis que estão presentes dentro das duas principais lojas oficiais do mercado (Google Play - Android e App Store - Apple) (CISCO, 2017; VIDAL, 2015; TAM *et al.*, 2020). Em ambos os casos, em 2021 havia em torno de 14 bilhões de dispositivos móveis e com previsão de crescimento[1]. Além disso, em relação as aplicações, existem 72% dos sistemas operacionais dos dispositivos móveis do mundo voltados para o *Android*, enquanto que 26% usam iOS e os outros 2% ficam distribuídos entre *Windows Phone*, *Nokia*, *Samsung* e outros.

Nos últimos anos a adoção de *smartphones* cresceu de forma vertiginosa, principalmente em decorrência da evolução de *hardware* e *software* possibilitando aos usuários com menos recursos financeiros adquirirem dispositivos móveis (BRIEDE-WESTERMEYER *et al.*, 2020). Em 2010 em torno de 300 milhões de *smartphones* eram vendidos por ano para usuários finais, enquanto que em 2017 por volta de 1.5 bilhões estavam sendo vendidos[2]. Assim, o uso de soluções móveis teve um crescimento em decorrência da aquisição e do uso de *smartphones*, além das aplicações (BRIEDE-WESTERMEYER *et al.*, 2020).

1.2 Justificativa

Soluções móveis são usadas para tarefas simples como, lembrete de remédios, assistir vídeos e ouvir música. Além disso, também popularizou-se o uso para entretenimento como jogos e plataformas de conteúdos pagos. No entanto, há usos em operações mais complexas e delicadas como pagamentos e transferências bancárias, ou até mesmo soluções relacionadas a monitoramento de pessoas através de plataformas de *Internet of Things* (*IoT*) (TAIVALSAARI; MIKKONEN, 2018), que fazem uso de sensores para detectar riscos como quedas ou ataque vascular celebral (AVC) (CASILARI *et al.*, 2016).

Entretanto, desenvolver e manter uma aplicação móvel é uma tarefa complexa, tendo em vista os inúmeros desafios técnicos e de negócio. Desafios como segurança, qualidade, heterogeneidade dos dispositivos e usabilidade são dificuldades que desenvolvedores encontrarão

[1] <https://www.statista.com/statistics/245501/multiple-mobile-device-ownership-worldwide/>
[2] <https://www.statista.com/statistics/245501/multiple-mobile-device-ownership-worldwide/>

durante a construção das soluções móveis (MARTÍNEZ-PÉREZ *et al.*, 2015; ZHU *et al.*, 2014; NAYEBI *et al.*, 2012). Além disso, existe outra grande dificuldade que tem se mostrado importante no desenvolvimento de aplicações modernas, que é a acessibilidade (ALAJARMEH, 2021; SIEBRA *et al.*, 2015).

A acessibilidade, no contexto de desenvolvimento móvel, é a capacidade de uma aplicação fornecer para qualquer pessoa, independente da deficiência ou dificuldade, mecanismos para o uso de forma adequada e completa (GOV.BR, 2022). A discussão sobre aplicações móveis accessíveis ganhou destaque primeiro pela grande quantidade de aplicações e *smartphones* no mundo, mas também por que, segundo o Relatório Mundial de Deficiências da Organização mundial da saúde (OMS), por volta de 15% da população mundial vive com alguma forma de deficiência (BICKENBACH, 2011; JONES, 1981). Portanto, pessoas com deficiência constituem uma parcela considerável dos usuários de aplicações no mundo e aproximadamente 86.3% dos usuários deficientes usam as tecnologias e aplicações através de dispositivos móveis[3].

Com o intuito de fornecer suporte ao desenvolvimento de aplicações acessíveis tanto para o *Android* quanto ao *iOS*, foram construídas ferramentas leitores de tela. Na plataforma *Android*, existe o *Talkback*[4], que é uma ferramenta responsável por facilitar a interação com os elementos e funcionalidades da tela para usuários com deficiência visual. No *iOS*, existe o *VoiceOver*[5]. Essas ferramentas são mantidas por suas respectivas empresas e comunidades.

Segundo Vendome *et al.* (2019), foi realizado um estudo empírico de acessibilidade em aplicações *Android* em 2018 da Universidade de Oxford, da análise de 13 mil aplicações, aproximadamente 50.08% possuía suporte a acessibilidade em todos os elementos, embora 46.25% das aplicações possuíssem pelo menos metade dos elementos sem suporte a acessibilidade e desse percentual, aproximadamente 5 mil aplicações não possuíam nenhum suporte a acessibilidade. Além disso, aproximadamente 98% dos códigos-fonte dos 13 mil aplicações não possuem nenhuma referência para as bibliotecas de acessibilidade, ou seja, embora haja um *framework* como o *Talkback*, os desenvolvedores usam apenas funcionalidades básicas como leitura de texto, que podem ser inseridas sem necessariamente criar uma implementação. Além disso, fóruns relacionados ao desenvolvimento de software, como por exemplo *Stackoverflow*[6], relata inúmeros problemas com ferramentas de acessibilidade, o que pode justificar a falta de adoção. Assim, apesar da acessibilidade ser uma dificuldade tão importante quanto as citadas

[3] <https://webaim.org/projects/screenreadersurvey/>
[4] <https://developer.Android.com/guide/topics/ui/accessibility>
[5] <https://developer.apple.com/documentation/accessibility/supporting_voiceover_in_your_app>
[6] <https://stackoverflow.com/>

anteriormente, ela não recebe a devida importância no desenvolvimento. Com relação ao *Talkback*, os usuários com deficiências também possuem problemas ao usar a ferramenta, como por exemplo verbosidade da leitura, consciência e localização dentro das aplicações Rodrigues *et al.* (2015b).

1.3 Objetivos

1.3.1 Objetivo Principal

Com isso, este trabalho possui como objetivo principal o desenvolvimento de uma ferramenta de leitura assistida para o sistema operacional *Android*, focado em problemas encontrados na literatura e entrevistas realizadas. Salienta-se que, o trabalho não consiste em construir uma solução completa como o *Talkback* ou *VoiceOver*, mas sim identificar problemas relacionados ao uso do *Talkback* no *Android* (ver seção 4.3) e implementar uma ferramenta leitora de tela que resolva-os.

1.3.2 Objetivos Específicos

- Entrevistar usuários com deficiência visual, a fim de elencar quais problemas (requisitos) eles enfrentam na utilização das ferramentas;
- Identificar as causas da falta de acessibilidades no desenvolvimento de software;
- Fornecer orientações gerais sobre o desenvolvimento de aplicações móveis acessíveis;
- Validar a ferramenta com relação aos requisitos dos usuários; e
- Comparar a ferramenta com o *Talkback* em relação aos requisitos e uma prova de conceito.

1.4 Estrutura do documento

Este trabalho está organizado em seis capítulos, conforme descrito a seguir:

- **Capítulo 2 -** Pesquisa bibliográfica sobre o campo de interesse da pesquisa, a fim de compreender os conceitos e elementos fundamentais do trabalho: Desenvolvimento de Software para Dispositivos Móveis e Acessibilidade;
- **Capítulo 3 -** Análise e comparação de pesquisas que estão relacionadas ao presente trabalho;
- **Capítulo 4 -** Apresenta a proposta do trabalho, explicando a metodologia adotada

para a realização do mesmo, como foi conduzido, além da implementação da solução;

- **Capítulo 5 -** Apresenta os resultados encontrados para validação da solução; e
- **Capítulo 6 -** Apresenta conclusão do trabalho, bem como as limitações encontradas. Além disso, os trabalhos futuros para melhoria da solução.

2 FUNDAMENTAÇÃO TEÓRICA

Com o intuito de embasar a leitura deste trabalho, neste Capítulo será feito uma revisão da literatura juntamento com aspectos relevantes para o presente trabalho. A Seção 2.1 apresenta as definições relacionadas a deficiência e classificações. A Seção 2.2 aborda aspectos relevantes ao entendimento de acessibilidade. A Seção 2.3 aponta os diversos direcionamentos em guias para conteúdo acessível. A Seção 2.4 apresenta ferramentas para acessibilidade. Por fim, a Seção 2.5 discute sobre as razões para a falta de acessibilidade nas aplicações.

2.1 Deficiência

A deficiência é uma condição que faz parte do indivíduo que pode ser temporária ou permanente (BICKENBACH, 2011). Deficiência é um conceito amplo e possui várias subdivisões. Deficiências podem ser totais ou parciais e podem ter diferentes tipos, dependendo do tipo de condição que o individuo apresenta. O Decreto Nº 5.296 de 2 de dezembro de 2004 (BRASIL, 2004) descreve quatro tipos de deficiências mais comuns e que são importantes para esse trabalho, sendo elas a deficiência visual, auditiva, motora e cognitiva:

- **Deficiência Auditiva**: Caracterizado como a perda total ou parcial da habilidade de escutar sons em um ou nos dois ouvidos. Além disso, ela pode ser classificada em níveis com perda leve, moderada, acentuada, severa, profunda e perda total de audição;
- **Deficiência Visual**: Caracterizado como a perda total ou parcial da habilidade de enxergar em um ou nos dois olhos. Além disso, existem duas subdivisões importantes nessa deficiência que são o grupo da baixa visão, quando a perda da visão impacta nas atividades cotidianos, e existe o grupo da cegueira que é caracterizada pela perda total da visão;
- **Deficiência Motora**: Caracterizado como a perda total ou parcial da habilidade motora decorrente de lesões neurológicas ou neuromusculares. Além disso, ela possui vários tipos, como: Paralisia cerebral, Hemiplegia, Tetraplegias, Paraplegias, Paraparesia, Monoplegia, Monoparesia, Tetraparesia, Triplegia, Hemiplegia, Hemiparesia, Patologias degenerativas do sistema nervoso central e Amputações; e
- **Deficiência Cognitiva**: Caracterizado por alterações no funcionamento intelectual. Esse tipo de deficiência pode influenciar outras áreas, como comunicação, habilidades sociais e acadêmicas.

Diante disso, como o presente trabalho foca em pessoas com deficiência visuais,

assim nasce a necessidade de classificar o grau de deficiência de cada individuo. A Sociedade Brasileira de Visão Subnormal Leal (2017) utiliza uma classificação da deficiência visual apresentada na Tabela 1.

Tabela 1 – Classificação de Acuidade Visual

Tipo	Snellen	Decimal	Auxílio(s)
Visão Normal	20/12 a 20/25	1,5 a 0,8	Bifocais Comuns
Próxima Do Normal	20/30 a 20/60	0,6 a 0,3	Bifocais Mais Fortes Lupas De Baixo Poder
Baixa Visão Moderada	20/80 a 20/150	0,25 a 0,12	Lentes Esfero prismáticos Lupas Mais Fortes
Baixa Visão Severa	20/200 a 20/400	0,10 a 0,05	Lentes Asféricas Lupas De Mesa Alto Poder
Baixa Visão Profunda	20/500 a 20/1000	0,04 a 0,02	Lupa Montada Telescópio Magnificação Vídeo Bengala / Treinamento O-M
Próximo À Cegueira	20/1200 a 20/2500	0,015 a 0,008	Magnificação Vídeo Livros Falados Braille, Aparelhos Saída De Voz Bengala / Treinamento O-M
Cegueira Total	SPL	SPL	Aparelhos Saída De Voz Bengala / Treinamento 0-M

Fonte: Leal (2017)

O trabalho proposto foca principalmente em pessoas com **Baixa Visão Severa até Cegueira Total**, pois são os usuários que usam as ferramentas assistidas, segundo dados do censo demográfico do Instituto Brasileiro de Geografia e Estatística (IBGE) de 2010 (IBGE, 2010).

Além disso, existe o conceito de performance funcional [1] que trata das barreiras e experiências na qual usuários com deficiência enfrentam, cada deficiência impõe dificuldades que podem ou não serem exclusivas. Cada barreira possui critérios de aceitação que garantem que aquele problema foi resolvido e que o aplicativo vai fornecer uma boa experiência apesar da deficiência. A performance funcional trata não apenas da deficiência física, mas sim do que pode ser feito para contorna a situação.

Assim, com a crescente massa de aplicações sendo desenvolvidas e com a enorme porcentagem de aplicações que não fornecem acessibilidade para pessoas com deficiência visual, a accessibilidade se tornou uma funcionalidade requisitada e obrigatória nas aplicações móveis.

[1] https://www.w3.org/TR/wcag-3.0-explainer/#functional-needs

2.2 Acessibilidade Digital

Acessibilidade é um termo discutido nas mais diversas esferas sociais, além disso também possui definições amplas e genéricas (COUNCIL *et al.*, 2005). Segundo Henry (2006), a acessibilidade digital significa que pessoas com deficiências podem acessar e usufruir do conteúdo na *Web*, no entanto esse conceito também pode ser entendido para aplicações móveis. Ainda segundo o artigo anterior, ele define com mais precisão, impondo que pessoas com qualquer tipo de deficiência devem ser capazes de perceber, entender, navegar e interagir com o conteúdo.

Embora fornecer acessibilidade aparente ser simples, a grande variedade de deficiências implica em um trabalho maior para construção de aplicações acessíveis, além disso esse esforço adicional é muitas vezes colocado em segundo plano (ALSHAYBAN *et al.*, 2020a). Conforme *TechTrends (2008)*, dentre os quatro tipos de deficiências citados anteriormente, existem vários desafios no que diz respeito as aplicações móveis, e pode-se citar os seguintes:

- Existem pessoas que possuem dificuldades motoras, ou seja, precisam de mecanismo que auxiliem na interação com os elementos na tela, por exemplo, comandos de voz ou equipamentos externos;
- Existem pessoas que possuem dificuldades na visão, ou seja, precisam que a interface de modo geral seja visível mesmo para quem possui problemas para enxergar. Além disso, caso o indivíduo possua perda total da visão, o conteúdo da tela deve ser possível de ser mostrado através de áudio para o usuário por meio de uma ferramenta assistiva; e
- Existem pessoas que possuem dificuldades auditivas, ou seja, todo o *feedback* da tela deve ser visual, por exemplo, se a aplicação possui notificações, elas não devem ser apresentadas apenas na forma de áudio.

2.3 Guias para conteúdo acessível

Existem guias para o desenvolvimento de conteúdo acessível em aplicações. O grupo de acessibilidade na Web do *World Wide Web Consortium (W3C)* chamado de *Web Content Accessibility Guidelines (WCAG)*[2] desenvolveu um guia que traz várias dicas e regras que desenvolvedores podem seguir com intuito de construir uma aplicação acessível, embora o *WCAG* não se restrinja ao desenvolvimento móvel. Existe o SiDi (2015), que possui orientações

[2] <https://www.w3.org/WAI/standards-guidelines/wcag/>

sobre o desenvolvimento acessível. Por fim, a Associação brasileira de normas técnicas (ABNT) possuia um conjunto de 54 requisitos para uma aplicação ser considerada acessível (ABNT, 2022). Considerando as referências citadas, as principais característica para que uma aplicação possua uma boa acessibilidade são:

- ***Design* Minimalista**: O *design* de uma aplicação para pessoas com deficiência precisa ser simples, pois os usuários montam um mapa mental da aplicação, assim muitos elementos na tela podem impossibilitar o usuários de construí-lo. Vale destacar que o objetivo não é limitar as habilidade criativas de *design*, mas sim diminuir a complexidade para usuários com deficiência visual. Por exemplo, a *User Interface (UI)* pode ser complexa e quando o modo de acessibilidade estiver ativado, ela pode ignorar os elementos que não fazem sentido neste contexto;
- **Fluxo Natural**: Embora para usuários comuns o posicionamento dos elementos e ordem, de modo geral, possa não representar um grande problema, o mesmo não ocorre com usuários com deficiência visual. Por exemplo, em uma listagem o leitor de tela pode parar de ler o fluxo natural dos elementos da lista (cima para baixo ou esquerda para direita) e mostrar um botão. Assim, a recomendação é sempre o fluxo natural;
- **Coerência Externa**: Ao invés de criar padrões ou modificar os padrões existentes, a recomendação é sempre seguir os existentes, por exemplo, o título da página deve sempre ser o primeiro elemento no modo de acessibilidade, mesmo que visualmente não seja. Nesse sentido, a chance do usuários conhecer é maior, uma vez que são padrões de outras aplicações e da plataforma;
- **Coerência Interna**: Garantir que a aplicação possua padrões internos. Por exemplo, se todos os botões de confirmação possuem o texto “confirmar”, não deve haver outros botões com o texto “aceitar”, principalmente para pessoas com deficiência visual, pois uma interface confusa impossibilita a construção do mapa mental.

O *WCAG* possui uma influência enorme na comunidade de accessibilidade. A maioria das ferramentas de qualidade de software relacionados com acessibilidade de aplicações usam orientações dele, para definir os critérios de aceitação e quais são os pontos relevantes para acessibilidade, conforme (SILVA *et al.*, 2019). Uma vez definida uma visão geral sobre o *WCAG*, as próximas subseções serão sobre as principais orientações para aplicações acessíveis. Todas as orientações descritas nas próximas subseções possuem 4 elementos básicos segundo WCAG (2016b), que são:

- **Princípio**: São princípios de acessibilidade de uma forma mais genérica possível. São quatro princípios no total, sendo eles perceptividade, operabilidade, entendibilidade e robustez;
- **Guias**: Para cada princípio há uma série de guias e tutorias que vão auxiliar a cumprir o referente princípio. No total são 13 guias;
- **Critério de aceitação**: É o resultado final depois de aplicar os princípios, pois o simples fato de aplicar determinados princípios não garante uma boa acessibilidade. Faz-se necessário verificar se os critérios de aceitação foram cumpridos; e
- **Conteúdo para tecnologia**: Para cada guia e princípio há conteúdos técnicos que vão auxiliar na aplicações deles.

Embora haja 13 guias e 4 princípios, nas próximas seções serão apresentados apenas 5 guias que correspondem a 2 princípios considerados mais importantes para o presente trabalho. Cada subseção abaixo é um guia.

2.3.1 Textos Alternativos

A perceptividade é a característica que uma aplicação possui de ter seus elementos perceptíveis por qualquer usuário. Dentro da perceptividade tem-se os textos alternativos. Textos alternativos servem para descrever melhor os elementos para quem possui algum tipo de deficiência, por exemplo, em imagens um usuário com deficiência visual não é capaz de identificar o que a imagem quer dizer, porém a aplicação deve prover uma forma de descrever aquela imagem para esses usuários.

Figura 1 – Conteúdo acessível no Android Studio.

Fonte: <https://www.petz.com.br/blog/curiosidades/piercing-para-cachorro/>

Listagem 1 – Inserindo textos alternativos em elementos no Android

```
<ImageView
    android:contentDescription="@string/dog_description"
    />
```

Fonte: Elaborada pelo autor

A Figura 1 mostra a imagem de um cachorro, porém para prover acessibilidade é necessário na etapa de codificação do sistema indicar que aquela imagem possui conteúdo de texto alternativo. A Listagem 1 apresenta um exemplo de como adicionar conteúdo alternativo usando a plataforma Android. Os textos alternativos são uma técnica simples e útil para melhorar a acessibilidade de imagens e elementos visuais de forma geral (FERATI, 2016).

Embora o exemplo acima seja simples, o texto alternativo pode ser usado para vários outros componentes, como listas, menus, gráficos e dentre outros. No entanto, nem todos os elementos precisam de conteúdo alternativo. Existem elementos que não necessitam de uma descrição visual. Por exemplo, um ícone de voltar para tela “Home”, não há necessidade de descrever que o ícone é uma flecha de direção a esquerda. Nesse caso, o conteúdo alternativo deve descrever algo como “Clique aqui para voltar a tela inicial”, que é a ação relacionado aquele elemento, ou seja, a informação importante para o usuário.

No caso de textos, é comum usar o valor do texto com conteúdo alternativo para ele mesmo. Entretanto, existe alguns casos que há necessidade do conteúdo alternativo, por exemplo, suponha uma tela de login que possui um botão com um texto “Salvar”, alterar o texto alternativo para “Clique aqui para submeter seu pedido” aumentaria a usabilidade, embora em ambos os casos a acessibilidade exista.

Dentro de uma aplicação que busca ser acessível, o conteúdo textual é fundamental, assim como a qualidade. Na figura 1, foi mostrada uma imagem com um cachorro, portanto, não é recomendado definir como texto alternativo o valor “Imagem”, mas sim “Foto de um cachorro”. O texto alternativo deve agregar valor a interface, caso contrário, uma interface repleta de texto sem sentido prejudica a experiência de navegação. Como orientação, é recomendável usar palavras e termos que sejam da linguagem humana, além de prover tradução para diferentes idiomas. Dessa forma, o usuário pode trocar a linguagem da interface sem precisar de outra versão do sistema. Caso dentro do texto haja termos que não fazem parte do padrão popular, a aplicação deve fornecer uma espécie de glossário, onde os usuários podem buscar o significado

de forma fácil.

2.3.2 Adaptatividade

A adaptatividade consiste em dizer se uma aplicação é possível de ser adaptada para diferentes cenários. Por exemplo, um site pode ser adaptado para plataformas *desktop* e móvel, assim o esperado é que haja a remoção de alguns elementos com o intuito de preencher a tela do dispositivo móvel. No entanto, a plataforma móvel não pode perder informações que são importantes para a compreensão do conteúdo do site (ZEMEL, 2015).

No contexto de desenvolvimento móvel, existem desafios para prover a adaptabilidade de uma aplicação (KIM, 2013; SILVA, 2014). Por exemplo, existem diferentes tipos de tela, assim uma aplicação pode apresentar os elementos grandes em dispositivos com alta densidade de pixel, porém em dispositivos menores, um usuário com problemas visuais pode não enxergar adequadamente. Portanto, a recomendação é sempre usar unidades relativas e evitar desenvolver com base em apenas um dispositivo (MULLINS, 2015).

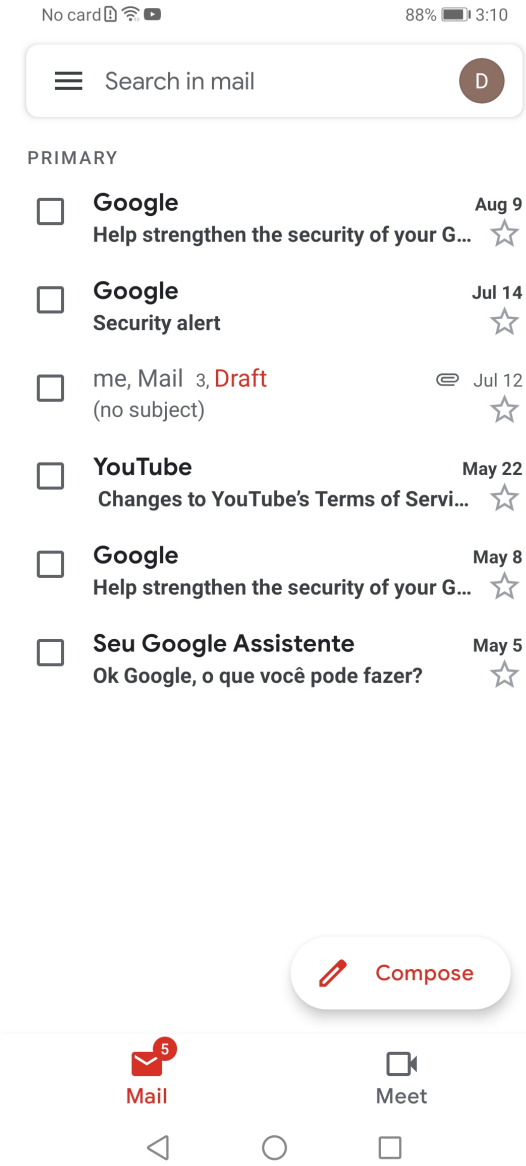
A adaptabilidade de uma aplicação também pode ser um fator importante no momento de escolher uma plataforma. Por exemplo, determinada aplicação pode conter uma versão móvel e uma versão Web. No entanto, também pode ser feita apenas uma versão Web que seja responsiva entre *desktops* e *smartphones* (TURNER-MCGRIEVY *et al.*, 2017).

Outro problema que existe é a complexidade dos elemento visíveis. Quanto maior a quantidade de elementos na tela, menor será o tamanho desses elementos, assim usuários com problemas visuais podem ter dificuldade em procurar elementos.

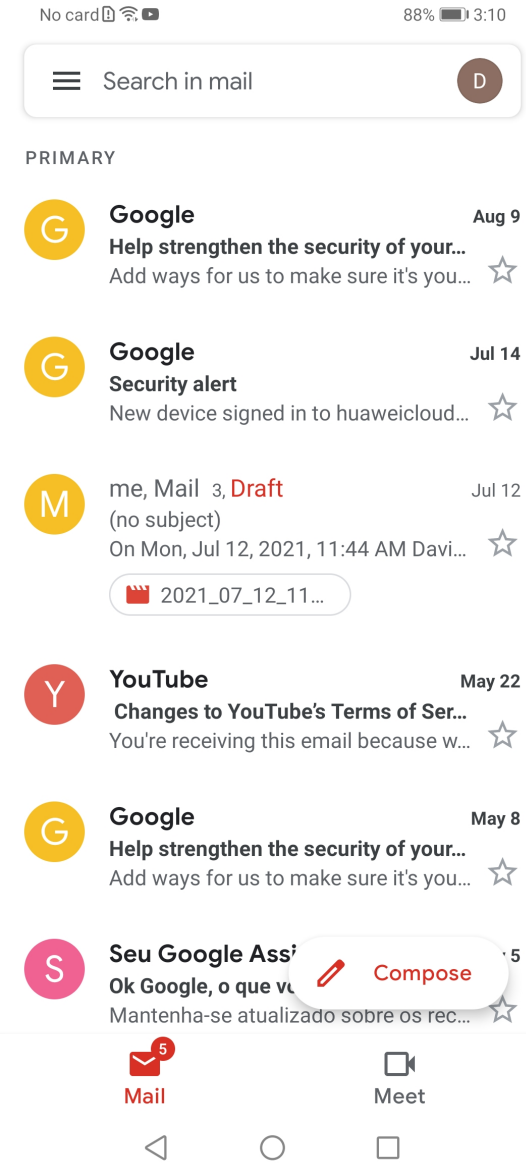
O problema citado acima é um exemplo de quando prover uma interface alternativa com menos elementos, mas que não perca as informações nem estrutura, uma vez que não interessante realizar mudanças bruscas na interface, pois prejudica a usabilidade de forma geral. A figura 2 mostra um exemplo da aplicação *Gmail*[3].

[3] <https://play.google.com/store/apps/details?id=com.google.android.gm>

Figura 2 – Interface detalhada e compacta do Gmail



(a) Layout compacto



(b) Layout detalhado

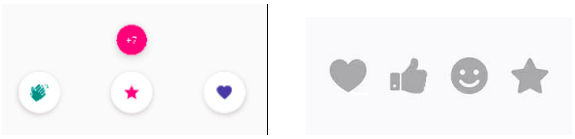
Fonte: Elaborada pelo autor

O exemplo acima mostra uma aplicação com dois layout diferentes que preservam a estrutura e os dados mostrados. Assim, um usuário que optar por utilizar um recurso de acessibilidade como o zoom, não terá problemas com a interface. No aplicativo *Gmail*, isso é feito através de um menu de configuração, em que o usuário pode optar por uma interface detalhada ou compacta.

Outro problema comum na adaptação é a orientação de dispositivos móveis, ou seja, se uma aplicação está na vertical ou na horizontal. A recomendação é que a tela seja adaptável a esse comportamento. Na plataforma Android, é possível fornecer uma codificação para as duas

orientações e ter um controle maior sobre customizações e adaptações.

Figura 3 – Preenchimento de botões



(a) Layout compacto (b) Layout padrão

Fonte: Elaborada pelo autor

Por fim, a Figura 3 demostra o problema de "Qual o propósito desse elemento". Uma interface pode ter diferentes elementos e comportamentos, assim o usuário não deve ficar em dúvida sobre o que é passível de interação ou não. Na figura 3a claramente o usuário é capaz de perceber que os ícones também são botões e qual sua área clicável. Já na figura 3b até mesmo um usuário sem nenhuma deficiência poderia ter dificuldade em identificar que os ícones na verdade são botões. Na figura 3b é possível que o leitor de tela não reconheça o ícone como botão, e não forneça o *feedback* adequado para o usuário.

2.3.3 Distinguibilidade

Distinguibilidade é a capacidade de uma aplicação possuir elementos que são facilmente distinguíveis entre si, usando de recursos visuais como o contraste entre cores e elementos WCAG (2016a). A principal orientação nesse caso é um alto contraste com o plano de fundo da aplicação, que na maioria das vezes pode ser branco. Um alto contraste no plano de fundo fornece meios para que os usuários identifiquem os elementos que estão nas camadas superiores com mais facilidade.

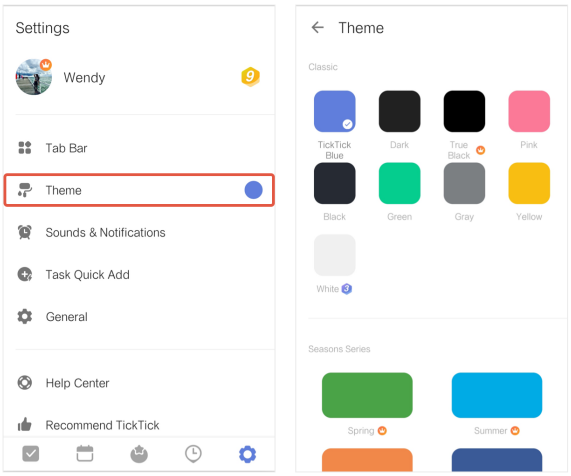
Nesse sentido, as cores possuem papel fundamental na distinção entre elementos na interface, por exemplo, quando o usuário está prestes a realizar uma operação de vendas no sistema, haverá um botão vermelho, para cancelar, e outro com a cor primária do sistema (possivelmente azul ou verde) para confirmar a operação. Assim, o usuário identificará com mais facilidade se esse padrão se mantém pelo resto do sistema. No entanto, as cores não podem ser a única forma de diferenciar elementos das interfaces, pois usuários com daltonismo não diferenciam vermelho do verde, por exemplo. Além disso, usuário com deficiência visual também não saberão distinguir entre as duas opções. Assim, o recomendável é sempre fornecer algum tipo de mecanismo de distinção alternativa, como textos.

Outro problema a respeito de distinguibilidade é a reprodução de vídeo ou áudio

esporadicamente. Assim, qualquer conteúdo de áudio ou vídeo que reproduza por mais de 3 segundos deve possuir um mecanismo fácil de controlar *WCAG (2016c)*. Usuários com deficiência motora ou visual terão problemas caso não seja possível desligar de forma fácil. Além disso, de acordo com o princípio da interferência do *WCAG*, uma aplicação não pode interferir de forma esporádica no fluxo de uso do usuário. Outra orientação é fornecer uma forma fácil de customizar a aplicação incluindo tamanho de texto, temas e distribuição dos elementos, pois cada usuário possui necessidades diferentes e isso deve ser customizável para eles.

A Figura 5 mostra um exemplo de como uma aplicação deve ser customizável para os mais diferentes usuários. Essa aplicação fornece suporte para customização de fonte, contraste e até mesmo diferentes visões para o gerenciamento de tarefas.

Figura 4 – Customização de temas na aplicação TickTick.



Fonte: <https://support.ticktick.com/hc/en-us/articles/360016274112-Change-themes>

Por fim, também há o contraste da aplicação. No entanto, o contraste não é restrito apenas ao plano de fundo, ele também trata de aspectos como texto e imagens. Segundo o *WCAG*, a taxa de contraste, que é um atributo que indica o quão duas cores são contrastantes, os textos e imagens devem está em uma proporção de de 7 para 1 de taxa contraste com o plano de fundo WCAG (2016d). O desenvolvedor pode verificar a taxa de contraste usando uma ferramenta como o *coolors*[4], que é uma site que calcula a taxa de contraste entre duas superfícies. Além disso, assim como o textos da aplicação, as imagens também precisam fornecer um mecanismo de contraste, por exemplo, quando o alto contraste for habilitado as imagens devem ser trocadas, a menos que a imagem em si já possua esse recurso.

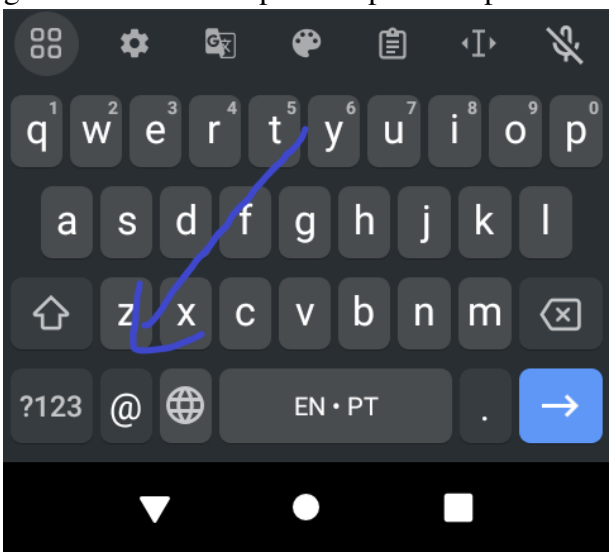
[4] <https://coolors.co/contrast-checker/6a92c0-acc8e5>

2.3.4 Teclado do Usuário

Nesta Subseção serão mostradas algumas orientações sobre o uso do teclado para aplicação acessíveis. A primeira delas é o foco inicial. Usuários com deficiência visuais muitas vezes possuem dificuldades para chegarem em determinado elemento editável. Por exemplo, em uma tela de formulário, o foco deve iniciar no elemento a ser preenchido primeiro, assim os usuário não precisaram navegar manualmente até a primeira caixa de texto. Outra orientação importante é a passagem do foco para o próximo elemento preenchível. Assim como no exemplo anterior, uma vez que o usuário preencheu o primeiro campo de texto, ele está apto a ir ao próximo, no entanto não é necessário que ele faça de maneira manual, basta habilitar o foco no próximo elemento, e quando o usuário pressionar a tecla *tab* ou no caso do Android a tecla *Enter*, ele moverá o foco para o próximo elemento progredindo no preenchimento do formulário.

Sobre codificação, é preciso especificar qual o tipo do campo de texto, assim quando o teclado for aberto, ele estará configurado. Por exemplo, um campo de *e-mail* já vai ser aberto com o símbolo de @ e o dispositivo completará automaticamente com *e-mails* previamente usados.

Figura 5 – Teclado específico para campos de email.



Fonte: Elaborado pelo autor

Isso garante uma melhor experiência, pois usuários com deficiência visual obtém *feedback* do leitor de tela e já vão saber o tipo de campo de texto a ser preenchido no momento. A Figura 6 mostra de forma aplicada como configurar o foco inicial de um campo de texto e o tipo. O nome de usuário é o primeiro a ser preenchido, por isso recebe a propriedade *autofocus*. Conforme a sequência descrita pela propriedade *tabindex*, o usuário pode avançar nos elementos.

Nesse caso, o uso dos recursos do *HyperText Markup Language (HTML)*, como as propriedades *autofocus* e *tabindex* são necessários para fornecer uma boa usabilidade no teclado.

Figura 6 – Configurando o foco inicial do teclado usando HTML.

```
<body>
    <form action="/login">
        Username:
        <input type="text" autofocus tabindex="1">
        Email:
        <input type="email" tabindex="2">
        Password:
        <input type="password" tabindex="3">
    </form>
</body>
```

Fonte: Elaborada pelo autor

2.3.5 Navegação

A navegação é um fator importante em qualquer tarefa que as pessoas fazem no cotidiano, quando essas pessoas possuem deficiências visuais, o desafio é maior. No contexto físico, pessoas com deficiência visual criam mapas mentais dos ambientes reais para facilitar a locomoção, como dito anteriormente, e além disso a criação dos mapas mentais deve ser fácil e intuitiva (FAÇANHA *et al.*, 2020). No desenvolvimento de aplicações móveis, as aplicações devem fornecer meios para os usuários navegarem, encontrar conteúdo e determinar onde eles estão (TAKAGI *et al.*, 2007). No entanto, essa afirmação é bastante genérica e nos próximos parágrafos serão discutidas orientações específicas.

Nos sites modernos, é comum quando o usuário entra pela primeira vez, existir um guia que ensina as principais funcionalidade da aplicação em questão. No entanto usuários com deficiência podem não querer passar pelo tutorial inteiro, assim a orientação é sempre prover uma forma de pular o tutorial WCAG (2016e). A Figura 7 mostra um exemplo onde a aplicação fornece uma maneira para pular essa etapa.

Figura 7 – Pulando tutoria inicial de um site.

Fonte: <https://nea.gov.bh/HTML/WAS/2.4.1%20Bypass%20Blocks.html>

Além disso, o sistema também deve prover um mecanismo para informar onde ele está, assim os leitores de tela podem identificar de forma clara qual a tela atual WCAG (2016f). Isso pode ser feito através de títulos nas páginas ou telas de uma aplicação. Além disso, a ordem do foco também é uma opção. Com relação as páginas Web, com o lançamento do *HTML* versão 5[5], várias funcionalidades foram adicionadas e dentre elas estão os elemento semânticos. Por exemplo, um leitor de tela pode inicialmente procurar por cabeçalhos e parágrafos ao invés de textos comuns, isso é possível através dos elementos semânticos que diferenciam a importância entre dois texto.

Figura 8 – Elementos semânticos do *HTML*.

```
<section class="about">
    <article>
        <header>
            <h2>Quem somos</h2>
        </header>
        <div class="content">
            <p>Lorem ipsum dolor sit amet, consectetur, adipiscing elit.</p>
            <p>
                Curabitur suscipt ultrices est sit amet ultricies. <br />
                Nam dui ipsum. varius at porta eget.
            </p>
        </div>
    </article>
</section>
```

Fonte: Elaborada pelo autor

O exemplo acima mostra a ideia de elementos semânticos. Há os textos “Quem somos” e “Loren ipsum...”, no entanto, o primeiro está dentro de uma *tag* com a importância de cabeçalho, assim os leitores de tela encontrarão o “Quem somos”, antes dos parágrafos, embora todos sejam textos.

Muitas vezes as aplicações usam de recurso como os gestos, que de forma geral, deixam a aplicação mais dinâmica Ruiz *et al.* (2011). No entanto, usuários com deficiência visual podem enfrentar dificuldades ao tentar replicar esse gestos. Como orientação, a ideia é sempre prover outra forma de executar a mesma operação. Por exemplo, voltar para a página

[5] <https://developer.mozilla.org/en-US/docs/Glossary/HTML5>

inicial pode ser feito puxando da esquerda para direita, no entanto deve haver um botão para fazer isso também. Além disso, sempre usar gestos simples e estáticos que não dependam de *feedback* na tela, ou seja, o gesto mudar conforme algo surgir na tela. Segundo o *WCAG*, não há problema em haver meios alternativos de entrada de dados pelo usuário, como os gestos, no entanto quando isso prejudica a usabilidade da aplicação deve ser desabilitada. Por exemplo, a ferramenta assistiva *TalkBack* possui gestos reservados e caso o aplicativo use esses gestos, os desenvolvedores devem desabilitar os gestos quando o modo de acessibilidade estiver ligado.

Como forma de fazer uma síntese dos guias mostrados acima, a Tabela 2 mostra o princípio, o guia, uma explicação e um exemplo de uso.

Tabela 2 – Resumo dos princípios

Princípio	Guia	Explicação	Uso
Perceptividade	Texto Alternativo	Qualquer texto ou imagem precisa possuir um texto alternativo e boa descrição	Os textos alternativos podem ser aplicados no *HTML*. Também é possível integrar o conteúdo acessível no *Android*
Perceptividade	Adaptatividade	A aplicação deve ser adaptável a todo tipo de tamanho de tela e gosto do usuário	Desenvolver aplicações que são responsivas para World Wide Web (*Web*).
Perceptividade	Distinguibilidade	O usuário não deve ter dúvida no que é elemento de interação e, além disso, ele deve diferenciar cada elemento	*WebAIM* é um site para verificar o grau do contraste das cores dentro da aplicação.
Operabilidade	Teclado de Usuário	Todos os campos de textos devem estar configurados de acordo com tipo da entrada do dado e, além disso, a ordem do foco também deve ser a ordem da tela	Escolhendo o tipo correto para determinado campo de texto no *HTML*. Além disso, também é possível configurar a ordem do foco no *HTML*.
Operabilidade	Navegação	Todas as telas precisam ter títulos claros, elementos na tela como cabeçalhos, devem está com a semântica correta, cada tela deve ter múltiplas formas de navegação dentro dela.	Construir páginas semânticas no *HTML*.

Fonte: Elaborada pelo autor

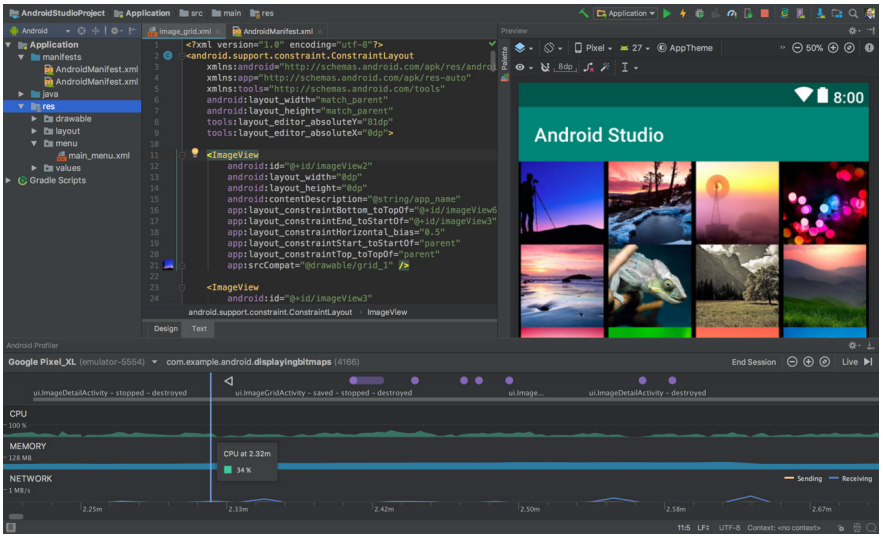
2.4 Ferramentas de Acessibilidade

Existem uma gama de ferramentas para desenvolver aplicações acessíveis, bem como fazer uma inspeção de acessibilidade. Nessa seção serão apresentadas ferramentas que contribuem para a criação de uma aplicação acessível.

2.4.1 Android Studio

O *Android Studio* é uma IDE, ou seja, ele possui todas as ferramentas necessárias para construção de aplicações móveis para o sistema operacional *Android*. O *Android Studio* é a ferramenta oficial da *Google* e usa vários *frameworks* de forma integrada. Para desenvolvimento nativo é a ferramenta mais utilizada, além disso conta com o suporte para as linguagens de programação *java* e *kotlin*, além do gerenciamento de projeto usando o *gradle*.

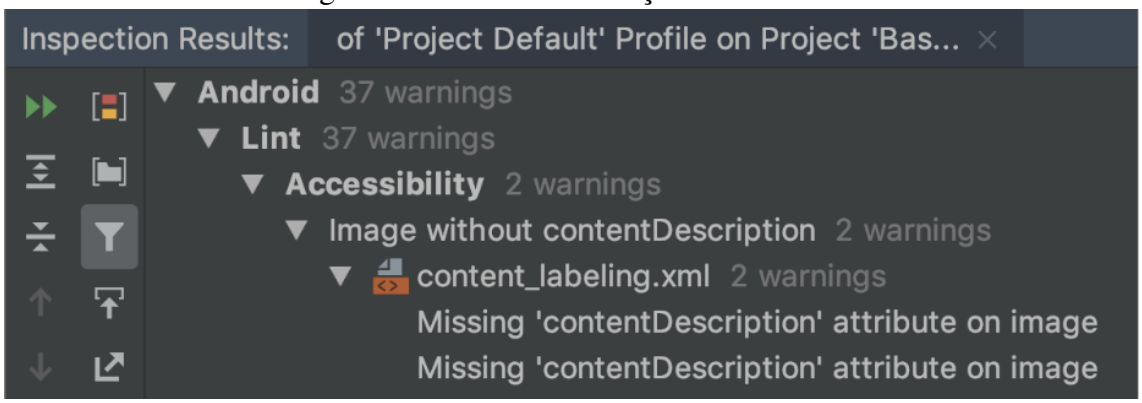
Figura 9 – Ambiente Android Studio



Fonte: Elaborada pelo autor

Android Studio possui ferramentas para verificar e testar a acessibilidade, uma delas é o *Lint*[6], que é uma ferramenta que utiliza de análise estática para encontrar possíveis erros. O *Lint* de código do *Android Studio* permite que desenvolvedores sejam avisados sobre eventuais trechos de códigos que não fornecem suporte à acessibilidade.

Figura 10 – Falta de descrição em elementos



Fonte: Elaborada pelo autor

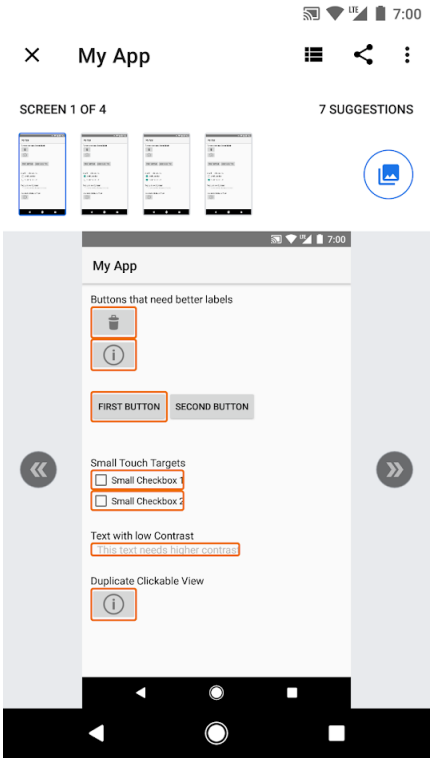
6 <https://developer.android.com/studio/write/lint>

Na figura 10, o *Android Studio* notifica que existe uma imagem sem descrição. Além disso, segundo o site de teste de acessibilidade oficial do *Android*[7], esse processo pode ser automatizado para facilitar o trabalho da equipe de testes.

2.4.2 Scanner de Accessibilidade

Outra ferramenta é o *Scanner* de Accessibilidade[8]. Ela é uma aplicação e tem as mesmas funcionalidades que o *Lint* do *Android Studio*, porém ao invés de fazer a verificação sobre o código, ele verifica a aplicação já instalada, a figura 11 mostra uma exemplo de uso do *Scanner* de Accessibilidade.

Figura 11 – Scanner de Accessibilidade



Fonte: Elaborada pelo autor

Na figura 11, o *Scanner* está verificando uma aplicação chamada *My App*, e identificou problemas com as descrições de alguns botões que possuem apenas ícones. O *Scanner* de Accessibilidade foi construído usando o *framework* de Teste de Accessibilidade para *Android*[9]. Esse *framework* é escrito em Java e segue as orientações da comunidade sobre a acessibilidade que estão presentes no *W3C*, e faz a verificação de forma automatizado. Portanto, caso seja

[7] <https://developer.Android.com/guide/topics/ui/accessibility/testing>
[8] <https://play.google.com/store/apps/details?id=com.google.Android.apps.accessibility.auditor&hl=pt-brRealmRealm>
[9] <https://github.com/google/Accessibility-Test-Framework-for-Android>

necessário, é possível criar uma ferramenta de teste de accessibilidade que atenda as necessidades específicas.

2.4.3 TalkBack

O *TalkBack*[10] é a ferramenta oficial para acessibilidade no *Android*. Ele usa de recursos como leitura assistiva para expor os dados da tela para usuários com deficiência visual. O *TalkBack* também suporta interações através de gestos. As principais funcionalidades do *TalkBack* são:

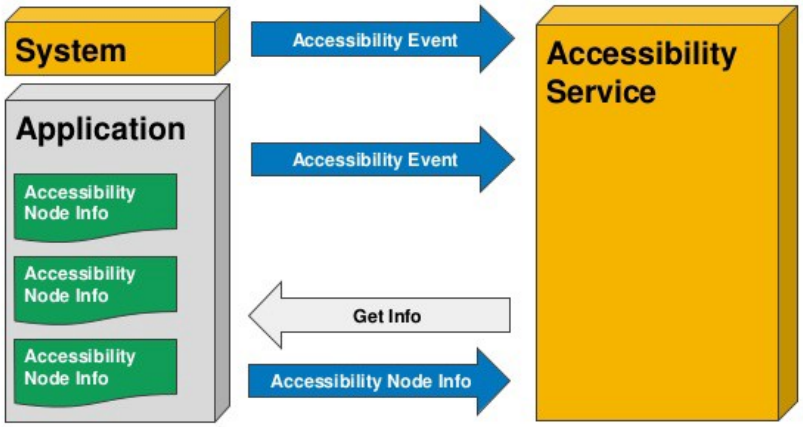
- Leitura assistiva de elementos da tela;
- Gesto de arrastar da esquerda para direita para voltar ao elemento anterior;
- Gesto de arrastar da direita para esquerda para avançar para o próximo elemento;
- Pressionar duas vezes a tela para simular um clique no elemento que está em foco; e
- Gesto de arrastar com os dois dedos pressionados ao mesmo tempo para cima ou para baixo, com intuito de fazer a navegação vertical.

O *TalkBack* também pode auxiliar pessoas que não possuem deficiências. Por exemplo, pessoas com dificuldade de leitura podem usar o *TalkBack* para facilitar o entendimento e interação com a tela, assim usam do recurso de ferramenta assistiva para compensar essa dificuldade (DÍAZ-BOSSINI; MORENO, 2014). Apesar das vantagens oferecidas, os gestos também são um fator preponderante para usuários desistirem das ferramentas de acessibilidade. De acordo com (RODRIGUES *et al.*, 2015a), de 5 participantes, apenas 1 era capaz de executar gesto de forma correta nos tutorias de início do *TalkBack*, ou seja, usuários iniciantes possuem barreiras na adoção de tais tecnologias.

Além disso, o *Talkback* possui uma arquitetura peculiar, conforme a fugura 12

[10] <https://support.google.com/accessibility/android/topic/3529932>

Figura 12 – Arquitetura da ferramenta de acessibilidade do *TalkBack*.



Fonte: <https://medium.com/mindorks/a-complete-guide-to-accessibility-service-part-2-ec2bf4b693b1>

O *Talkback* consiste em um serviço padrão do *Android*, mas com funções adicionais. Quando a aplicação recebe um evento de acessibilidade, ou seja, toque na tela ou gesto, a aplicação emite esse evento para o serviço de acessibilidade. Nesse sentido, o serviço de acessibilidade processa esse evento e retorna um resultado para aplicações, esta por sua vez, também retorna um *feedback* para o usuário, assim como mostrado na Figura 12. A título de exemplo, imagine que o usuário toque 5 vezes na tela consecutivamente, esse evento de 5 toques consecutivos é enviado para o serviço de acessibilidade, além disso o serviço pode ter sido feito de tal maneira que caso receba um evento de acessibilidade que descreve 5 toques consecutivos na tela, o serviço retorna para tela um pedido para que fale o seguinte texto: “Você tocou 5 vezes na tela consecutivamente”.

2.5 Falta de Acessibilidade nas Aplicações

Diante de tudo que foi apresentado, é claro que prover acessibilidade é um requisito essencial para a construção de uma solução de software, no entanto, ainda há uma questão pertinente, por que nem todas as aplicações são acessíveis ou possuem uma boa qualidade no que diz respeito a acessibilidade? De acordo com (VENDOME *et al.*, 2019), de 13 mil aplicações, aproximadamente, metade possui acessibilidade em um nível adequado classificado pelo próprio autor, sendo portanto, um valor muito baixo.

Um dos motivos poderia ser a falta de cultura dentro das empresas, ou seja, elas simplesmente não consideram importante que suas aplicações sejam accessíveis. Essa tese é

baseado no fato de que a maioria dos desenvolvedores alegam que as empresas consideram isso como um bônus, ou seja, se "sobrar tempo" será feito a adição de acessibilidade. Abaixo estão quatro frases que corroboram com tal ideia, elas foram retirada de uma entrevista conduzida no trabalho de (ARRUDA, 2020), que estudou a relação entre desenvolvedores e diretrizes de acessibilidade:

1. "a principal barreira são decisões de negócio que não levam em consideração os benefícios que a acessibilidade traz";
2. "a maior barreira é que o cliente talvez não compre essa ideia ou priorize outras coisas, a barreira principal é a prioridade";
3. "desde a primeira reunião técnica eu fui lá e levantei: 'Gente, mas o app vai ter 80 mil usuários, será que a gente não precisa fazer acessibilidade, trabalhar com isso e tal' ai eles: 'ah então, não vai ter tempo' [...] porque na época a gente iria fazer o app em 3 meses";
4. "a falta de percepção das pessoas de valorizar isso, pessoas que digo tomadores de decisão, tipo clientes não priorizam de fato a acessibilidade".

Como mostrado acima, a maioria das empresas e clientes não priorizam a etapa de acessibilidade, de fato quem toma as decisões normalmente não precisa de aplicações acessíveis. No entanto, a cultura das empresas não é um fator absoluto na falta de acessibilidade das soluções de software, o estudo de (ALSHAYBAN *et al.*, 2020b) mostrou que também há problemas técnicos no desenvolvimento de soluções acessíveis, sendo os principais destacados aqui no trabalho:

1. **Texto clicável**: Quando um elemento de texto é clicável, o leitor de tela reconhece normalmente, porém quando somente uma parte do texto é clicável, os leitores não identificam a área clicável corretamente o que gera problemas no uso da aplicação. Por exemplo, o texto "Aproveite nossa promoção e **compre agora**", em muitos casos apenas o "compre agora" é clicável, mas o leitor não identifica ele ou considera o texto como elemento de toque;
2. **Elementos clicáveis aninhados**: Acontece quando tanto o elemento pai quando o filho podem receber clique, por exemplo, um *card* de um produto que tem um botão comprar dentro, ambos são clicáveis;
3. **Caixa texto**: Como as caixa de texto são elementos dinâmicos é difícil escolher uma descrição consistente para o elemento, o correto é sempre que o usuário editar o campo de texto, sua descrição também seja alterado. A título exemplo, se um usuário editou o valor

para “Ana”, a descrição deveria ser “Campo de texto para preencher o nome, com o valor Ana”;

4. **Elementos customizados**: Quando o elemento é nativo do sistema operacional o leitor não encontra problemas como em botões ou texto, no entanto em componentes customizados o leitor não é capaz de gerar um conteúdo alternativo, nesse caso o ideal é que o desenvolvedor apenas entregue o texto alternativo pronto para o leitor, de modo que ele apenas reproduza. Por exemplo, em um componentes de gráfico não útil o leitor informar que existe um retângulo com 50*px* de altura, uma descrição mais apropriada seria: “O percentual de aplicações sem acessibilidade é de 50%”, isso é regra do próprio componente, por isso exige um esforço adicional na implementação.
5. **Ordem de leitura**: Os leitores usam o ordem que os elementos foram construídos no layout, ou seja, a ordem hierárquica. No entanto, a ordem em que os usuários fazem a leitura das informações na tela pode ser diferente, para conciliar a ordem hierárquica e a ordem de leitura, e isso exige um esforço adicional na criação nos layout.

Portanto, a falta de acessibilidade é causada tanto pelos os desafios da solução técnica como pela falta de cultura das empresas, além disso os desafios técnicos também impactam nas decisões de gerentes, que possuem a falsa ideia que não vão conseguir entregar o produto “principal” se houver uma parte do tempo focada em acessibilidade.

2.6 Considerações Finais

Este Capítulo apresentou os conceitos básicos sobre os temas do trabalho proposto: deficiência, acessibilidade, desenvolvimento acessível e ferramentas disponíveis. Sobre deficiência, foram apresentadas definições a cerca da deficiência auditiva, visual, motora e cognitiva. Com relação a acessibilidade foram apresentados vários conceitos relacionados a como criar uma aplicação acessível, juntamente com o guia para desenvolvimento, que foca em perceptividade, interações com áudio, adaptatividade de tela, teclado do usuário para entradas e navegação. Por fim, foram apresentados várias ferramentas, com foco maior no *TalkBack* que é a principal ferramenta para acessibilidade no *Android* usada para leitura de tela, com suporte para interações através de gestos. Além do estudo de problemas relacionados a acessibilidade em aplicações móveis, o trabalho proposto tem o intuito de propor melhorias para ferramenta *TalkBack*, por meio de um conjunto de requisitos desejados por usuários com deficiência visual. Existem vários trabalhos que fazem o uso dessa abordagem e serão apresentados no próximo capítulo.

3 TRABALHOS RELACIONADOS

Esse Capítulo apresenta os trabalhos relacionados ao trabalho proposto, o qual discutem essencialmente sobre acessibilidade tendo em vista tanto aspectos da literatura quanto problemas em abertos com relação a acessibilidade.

3.1 Trabalho de Schnelle-Walka *et al.*

O trabalho de Schnelle-Walka *et al.* (2014) tem como foco identificar desafios que ainda existem no uso do *smartphone* por pessoas com deficiência visual. A metodologia usada nesse trabalho foi através de um estudo de caso com um conjunto de pessoas, ou seja, cada usuário utilizaria a aplicação e seriam anotados os aspectos pertinentes na usabilidade das aplicações por parte dos deficientes visuais. Dentre os principais aspectos observados por esse trabalho é possível destacar:

- **Primeiros passos**: Embora tecnologias como *TalkBack* e *VoiceOver* forneçam instruções de como usar o recurso de acessibilidade, muitos usuários ainda reportam que por diversas vezes acabam desistindo de usar os *smartphone*;
- **Descobrimento das funcionalidades**: Muitos usuários reportaram que não conseguem descobrir quais opções estão disponíveis para exploração;
- **Assistência externa**: Embora muitos operações sejam possíveis usando leitores de tela, eventualmente os usuários precisam solicitar ajuda para alguma ação, por exemplo, trocar o idioma; e
- **Consistência**: Dentro da aplicação a consistência é um problema, afinal quando uma aplicação tem diferentes padrões o usuário com deficiência visual não consegue memorizar determinados pontos.

Memorizar é uma etapa fundamental no uso de aplicação por deficientes visuais, assim os usuários são capazes de criar um modelo mental da aplicação. Esse tipo de habilidade que os usuários criam ao usar uma aplicação é chamada de mapa mental e é usada em várias esferas sociais, além disso o uso de mapa mental tornou-se uma ferramenta bastante útil na acessibilidade (SCHNELLE-WALKA *et al.*, 2014).

Como resultado desse trabalho, foram obtido 13 problemas fundamentais relatados pelos usuários, que são:

- Rótulos inadequados ou inexistentes;

- Mudar texto de caixa de texto;
- Não encontrar a opção desejada;
- Não conseguir fazer *login* em aplicações ou sites;
- Ficar perdido na aplicação;
- Falta de responsividade;
- Caixa de texto muito lenta;
- Não sabe quais gestos são suportados pela ferramenta;
- Sons sem significado específicos;
- Determinados gestos são difíceis de serem feitos;
- Não consegue executar rolagem em listas;
- Selecionar opção sem intenção; e
- Leitor de tela é lento.

3.2 Trabalho de Balansin

O trabalho de Balansin (2011) possui como objetivo especificar e implementar uma ferramenta de acessibilidade. O trabalho inicia com uma revisão teórica sobre ferramentas assistidas, leitores de tela e sua importância para o público deficiente visual. O trabalho também aborda sobre os tipos de leitores de tela, que são:

- **Tipo CLI**: Uma Command Line Interface (*CLI*) é uma aplicação que possui suas interações de entrada através do teclado, e a saída, através de um monitor. Nesse tipo de aplicação o leitor de tela tinha como funcionalidade apenas ler os textos de saída já que não há outro tipo de interações. Como exemplo de leitores de telas para *CLI* existem o *Expeak* [1] e o *DOSVOX* [2], esses projetos foram remodelados e foram criados quando o sistema operacional *Windows* não suportava interface gráfica.
- **Tipo GUI**: Uma Graphical User Interface (*GUI*) é uma aplicação onde há uma interface gráfica, ou seja, são as aplicações executadas em computadores e celulares no cotidiano atual. Como exemplos existem as ferramentas já citadas como *Talkback* e *VoiceOver*.

Como o título do trabalho supõe, um dos objetivo é fazer um levantamento de requisitos a serem implementados pela ferramenta, no entanto, ele não deixa isso de forma explícita e comenta apenas que sua ferramenta deve possuir as mesmas funcionalidade de outros

[1] https://livreeaberto.com/espeak-texto-para-fala-linux
[2] http://intervox.nce.ufrj.br/dosvox/

leitores. Entretanto, durante o texto ele discute sobre implementações de cliques de botões e leitura de texto usando síntese de voz, que são os requisitos primordiais de qualquer leitor de tela. Com relação a implementação, ele decidiu implementar a ferramenta para um ambiente gráfico *desktop* no sistema operacional *Linux* e usando a linguagem *Python*.

A metodologia usado no trabalho foi comparar a solução desenvolvida com a ferramenta de leitura de tela para *linux*, o *ORCA* [3]. O resultado do trabalho foi obtido a partir dessa comparação, de modo geral, ele implementou as funcionalidades principais para um usuário deficiente visual usar, no entanto, o leitor desenvolvido no trabalho não é completo quando comparado ao *ORCA*. Também vale destacar que o fato do leitor não ser completo não é um problema, afinal a ferramenta desenvolvida pelo presente trabalho também não é, o importante é a contribuição tanto teórica quando prática do desenvolvimento da ferramenta.

3.3 Trabalho de Vendome *et al.*

O trabalho de Vendome *et al.* (2019) é um estudo de caso sobre a acessibilidade em aplicações *Android*, e elencar os principais problemas que levam a equipe de desenvolvimento em não fornecer acessibilidade em aplicações. A metodologia usada nesse trabalho para fazer a inspeção de acessibilidade foi usando automação. A primeira parte foi um levantamento das aplicações, das quais foram selecionadas 13 mil aplicações da *google play*. Durante esse estudo, foi observado que aproximadamente 50% possuíam algum suporte à acessibilidade. No entanto, aproximadamente 46% possuíam ao menos um elemento sem conteúdo alternativo, e dos 13 mil aplicações, aproximadamente, 5 mil não possuíam nenhum conteúdo alternativo.

Além disso, dos 13 mil aplicações, apenas algo em torno de 280 aplicações possuem referências para Application Programming Interface (*API*)s de acessibilidade, note que isso não quer dizer que apenas os 280 possuíam acessibilidade, mas sim que, internamente, as aplicações não faziam uso das *API*s de acessibilidade. Além do mais, o estudo de caso acima não tinha o objetivo de avaliar a qualidade da acessibilidade fornecida, mas sim apenas identificar o uso de tais ferramenta de acessibilidade. Os critérios usados seguem o *framework* de acessibilidade do *Android*.

Por outro lado, esse artigo também faz um levantamento de problemas encontrados em fóruns, como o *Stackoverflow*. A metodologia usada nessa pesquisa foi realizar buscas por meio de uma *string*. Essa *string* possui informações para encontrar questões relacionadas ao

[3] https://help.gnome.org/users/orca/stable/introduction.html.pt_BR

TalkBack, além disso esse levantamento é especificamente para *issues* relacionadas ao *TalkBack*. Os principais problemas encontrados foram erros de pronúncia e compatibilidades com funcionalidades da própria aplicação. Algumas das conclusões desse trabalho foram listadas abaixo:

- Desenvolvedores devem usar mais ferramentas para dar suporte à acessibilidade;
- Baixo suporte para teste de acessibilidade; e
- Nem todos os problemas podem ser resolvidos usando o leitor de tela.

3.4 Trabalho de Bezerra

O trabalho de Bezerra (2021) possui como objetivo fazer uma análise comparativa entre os *frameworks* de desenvolvimento móvel Nativo (*Android*), *Ionic* e *React Native*, observando como essas ferramentas se comportam quando o desenvolvedor não fornece nenhum suporte à acessibilidade.

A metodologia usada nessa pesquisa foi através de uma Proof of Concept (*PoC*), ou seja, a mesma aplicação foi construída com as tecnologias Nativa, *Ionic* e *React Native*. A prova de conceito é uma aplicação de *Fitness*, isto é, uma aplicação onde é possível visualizar e adicionar dados de exercício físico. Para fazer a pesquisa, o trabalho passou por algumas etapas. A primeira foi a pesquisa exploratória que fornecia uma espécie de fundamentação possibilitando o uso e desenvolvimento com as ferramentas citadas. A segunda fase foi o desenvolvimento da *PoC* em todas as tecnologias, no entanto sem fornecer qualquer suporte à acessibilidade.

A terceira fase teve como objetivo fazer uma análise inicial, ou seja, como essas tecnologias se comportavam, uma vez que na etapa de desenvolvimento nenhum suporte à acessibilidade foi adicionado. Como resultado relevante dessa fase, pode-se destacar que o *Ionic* foi o que mais tratou a falta de acessibilidade deixada pelo desenvolvedores.

Depois da avaliação inicial, foi feita a implementação da acessibilidade em cada uma das plataformas, uma vez finalizada a implementação, foi feita uma segunda análise. No entanto, um resultado encontrado nessa segunda análise foi que embora na primeira etapa o *Ionic* foi o que mais corrigia erros de falta de acessibilidade, após o tratamento desses erros em tempo de desenvolvimento, o *Ionic* passou a ter mais problemas que as outras plataformas.

Por fim, foi feita uma análise com um usuário com deficiência visual com um total de 10 perguntas. Nessa etapa não foi incluso a versão com *Ionic*, além disso a aplicação com *Android* Nativo e *React Native* obtiveram 100% e 97.5% respectivamente de aprovação do

usuário.

3.5 Comparação dos Trabalhos Relacionados

A Tabela 3 busca fazer um comparativos entre os trabalhos relacionados usando como base os seguintes aspectos:

- ***TalkBack***: Se o trabalho aborda o *TalkBack*, seja como implementação ou estudo;
- **Autoria própria**: Se como resultado do trabalho uma solução relacionada com acessibilidade foi entregue, além das contribuições teóricas;
- **Entrevista com usuários**: Caso tenha sido feito estudos de casos com usuários, como entrevistas ou análises de uso. Não se enquadram trabalhos que usam resultados de outros trabalhos; e
- **Entrevista antes e depois**: Se nas entrevistas feitas pelo trabalho (caso exista), foram feitas entrevistas antes para análise e levantamento de requisitos e depois para garantir que os resultados e requisitos da primeira fase foram cumpridos. Não se enquadram trabalhos que possuem apenas uma etapa de entrevista.

Tabela 3 – Comparação dos Trabalhos Relacionados

Trabalho	*TalkBack*	Autoria própria	Entrevistas com usuários	Entrevista antes e depois
Trabalho de Schnelle-Walka *et al.*	Sim	Não	Sim	Não
Trabalho de Balansin	Não	Sim	Sim	Não
Trabalho de Vendome *et al.*	Sim	Não	Não	Não
Trabalho de Bezerra	Sim	Não	Sim	Não
Proposta	**Sim**	**Sim**	**Sim**	**Sim**

Fonte: Elaborada pelo autor

3.6 Considerações Finais

Esse capítulo apresentou os trabalhos relacionados, que envolvem o desenvolvimento de aplicações acessíveis. Os trabalhos abordam problemas em abertos com relação a acessibilidade. O próximo capítulo apresenta a proposta do trabalho, assim como a metodologia adotada, como os passos da metodologia foram realizados e detalhes da implementação.

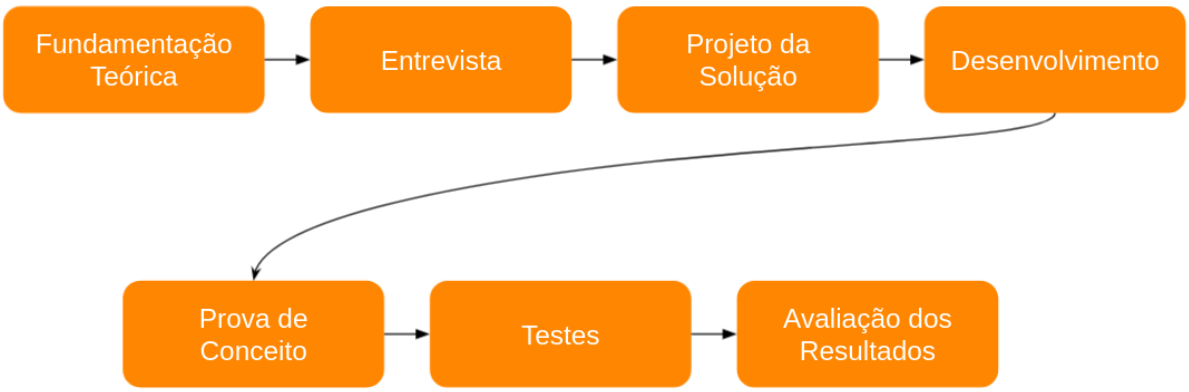
4 PROPOSTA

Neste capítulo será apresentado a proposta deste trabalho, além da metodologia adotada. A proposta consiste na construção de uma ferramenta leitora para acessibilidade na plataforma *Android*. A Seção 4.1 apresenta a metodologia adotada. A Seção 4.2 apresenta as entrevistas e requisitos iniciais levantados para a solução. A Seção 4.3 apresenta problemas e limitações encontradas do *Talkback*. A Seção 4.4 apresenta o projeto, arquitetura e detalhes de implementação da solução.

4.1 Metodologia

A metodologia usada para extração de dados do usuário no presente trabalho, possui uma abordagem qualitativa e de caráter exploratório, pois busca investigar com mais detalhes os reais problemas de usabilidade por parte dos usuários com deficiência visual. Foi adotado entrevistas para coleta de dados. Além disso, foi identificado quais procedimentos são executados para contornar determinadas situações com o intuito de melhorar esse processo. Como mostrado na Figura 13, o presente trabalho possui 7 etapas, em que a primeira etapa consistiu no embasamento teórico sobre o tema.

Figura 13 – Etapas do trabalho



Fonte: Elaborada pelo autor

A segunda etapa da pesquisa são as entrevistas juntamente com as observações. O intuito das entrevistas com pessoas com deficiência visual foi identificar as dificuldades encontradas por elas quando utilizam as aplicações móveis usando ferramentas leitoras de tela, focando no *Talkback*. No total foram 3 entrevistados e todos com experiência avançada em uso de leitores de tela, é considerado avançado quem tem pelo menos 5 anos de uso, a tabela 4

descreve melhor o perfil dos entrevistados.

Tabela 4 – Perfil dos entrevistas - Entrevistas Iniciais

#	Sexo	Idade	Tempo de uso	Usuários *Talkback*	Classificação de acuidade visual
Usuário 01	Masculino	45	> 15 anos	Sim	Baixa visão profunda
Usuário 02	Feminino	30	> 10 anos	Sim	Cegueira total
Usuário 03	Masculino	30	> 8 anos	Sim	Cegueira total

Fonte: Elaborada pelo autor

A partir das análises das entrevistas e da exploração desse resultados, a terceira etapa teve por objetivo refinar os requisitos e analisar as hipóteses sobre as quais as entrevistam buscam validar. Os requisitos foram necessários para a construção da ferramenta de acessibilidade.

Uma vez que tais requisitos foram levantados e de posse deles, a quarta etapa foi iniciada. Essa etapa teve como objetivo desenvolver o leitor de tela e a prova de conceito. Para fins didáticos o leitor de tela construído no presente será referenciado como *blid reader*, o significado de *blid* vem do inglês *blind* que significa cego e *reader* que significa leitor, ou seja, "leito cego", pois o trabalho é um leitor de tela para usuários com deficiência visual.

Com a prova de conceito construída e o *blid reader*, foi feita a etapa de testes com usuários com deficiência visual. Essa etapa teve como objetivo validar se os requisitos e hipóteses levantadas na etapa inicial de entrevistas estavam corretos, e além disso se os problemas foram resolvidos de forma total ou parcial pelo *blid reader*. Essa etapa também consistiu de uma entrevista com 3 novos usuários com experiência avançada, conforme a tabela 5

Tabela 5 – Perfil dos entrevistas - Entrevistas Finais

#	Sexo	Idade	Tempo de uso	Usuários *Talkback*	Classificação de acuidade visual
Usuário 01	Masculino	50	> 15 anos	Sim	Cegueira total
Usuário 02	Feminino	60	> 15 anos	Sim	Baixa visão profunda
Usuário 03	Masculino	45	> 5 anos	Sim	Cegueira total

Fonte: Elaborada pelo autor

Por fim, a última etapa foi a avaliação dos resultados encontrados. Os resultados foram avaliados de forma geral, ou seja, não apenas resultados da etapa anterior, mas sim dados coletados durante toda pesquisa.

4.2 Entrevistas Iniciais

As entrevistas iniciais com pessoas com deficiência visual serviram para identificar as dificuldade encontradas por eles. Elas foram realizadas de forma remota. Quanto aos temas abordados nas entrevista, esses por sua vez, foram extraídos dos problemas levantados por Rodrigues *et al.* (2015b), ou seja, a maioria das perguntas ou são problemas em aberto, segundo o artigo, ou foram adaptadas a partir de um desses problemas. A Tabela 6 possui os temas relacionados com a primeira entrevista.

Tabela 6 – Entrevista de levantamento de requisitos

#	Descrição
Tema 1	**Verbosidade da ferramenta assistiva do *Android*:** Esse tema busca identificar se os usuário acreditam que a leitura possuem informações desnecessárias, ou é cansativa
Tema 2	**Orientação dentro da aplicação:** O objetivo desse tema é identificar se o usuário fica perdido dentro da aplicação, ou seja, se ele se perde e precisa fechar e voltar para o início.
Tema 3	**Exploração no uso de uma aplicação nova:** O objetivo desse tema é analisar o comportamento dos usuários mediante ao uso de uma aplicação, ou seja, no processo de primeiro uso das aplicações.
Tema 4	**Funções de ações rápidas como copiar e colar:** Esse tema busca identificar se os usuários deficientes acreditam ser útil a ferramenta possuir ações rápidas como copiar, colar e compartilhar conteúdo através de um gesto simples como arrastar para baixo.

Fonte: Elaborada pelo autor

A Tabela 6 descreve apenas os temas, tendo em vistas que algumas perguntas foram agrupadas por serem semelhantes. As peguntas originais podem ser encontrados no anexo A.1, com a separação entre perguntas do trabalho Rodrigues *et al.* (2015b) e do presente trabalho.

Essa entrevista foi realizada com três usuários deficientes visuais, em que todos possuem nível avançado de experiência com ferramentas leitoras de tela, conforme a tabela 4. Essas entrevistas possuem como função principal a extração dos requisitos para implementação do *blid reader* e validar os pontos que o trabalho de Rodrigues *et al.* (2015b) levantou.

Vale ressaltar que o usuário ficou livre para expressar suas opiniões de modo geral e não apenas para um aplicativo ou plataforma.

A Tabela 7 descreve as resposta subjetivas de cada usuário com relação aos temas.

Tabela 7 – Resultado das entrevistas

Entrevistado 1	
#	Resposta
Tema 1	**Verbosidade da ferramenta assistiva do *Android*:** O usuário respondeu que a verbosidade incomoda um pouco, mas que não é um aspecto tão importante, pois "Já estou acostumado, já usei muito...". Ainda falou que preferia aplicações acessíveis ao invés de menos verbosas, mas também disse que se fosse para melhorar a aplicação, ele aceitaria.
Tema 2	**Orientação dentro da aplicação:** O usuário disse que não tem problemas com isso, pois monta uma espécie de mapa mental da aplicação, a menos que a acessibilidade seja pouco explorada.
Tema 3	**Exploração no uso de uma aplicação nova:** O usuário informou que muitas das vezes precisa conversar com outros usuários que usam aquela aplicação para aprender a usar.
Tema 4	**Funções de ações rápidas como copiar e colar:** Usuário falou que acha muito válido, pois facilitaria o uso no cotidiano.
Entrevistado 2	
#	Resposta
Tema 1	**Verbosidade da ferramenta assistiva do *Android*:** O usuário respondeu que às vezes é ruim e citou o exemplo do botão de gravar áudio da aplicação *Whatsapp*, que possui conteúdo alternativo extenso e, segundo o usuário falou, que já sabe o que é o botão e incomoda ter que escutar o texto.
Tema 2	**Orientação dentro da aplicação:** O usuário informou que não tem problemas, assim como o Entrevistado 1.
Tema 3	**Exploração no uso de uma aplicação nova:** O usuário disse que não tem problema, pois quando estava no início do uso sempre fazia exercícios mentais de exploração em aplicações desconhecidas, mas confessou que quando a aplicação não possui acessibilidade fica muito complicado explorar.
Tema 4	**Funções de ações rápidas como copiar e colar:** Usuário disse que seria de grande ajuda, pois seria como um atalho, e finalizou falando que em outras plataformas como *iOS* tem essas funcionalidades.
Entrevistado 3	
#	Resposta
Tema 1	**Verbosidade da ferramenta assistiva do *Android*:** O usuário informou que existe caso que a verbosidade atrapalha, falando por exemplo "Botão" ou "Caixa de Texto", mas que muitas das vezes é a aplicação que não possui elementos com conteúdos alternativos adequados.
Tema 2	**Orientação dentro da aplicação:** O usuário informou que não tem problemas, assim como os entrevistados 1 e 2.
Tema 3	**Exploração no uso de uma aplicação nova:** O usuário disse que gosta de explorar mesmo quando a aplicação possui uma acessibilidade ruim e acredita que com o tempo ele se habitua com o uso.
Tema 4	**Funções de ações rápidas como copiar e colar:** Assim como os entrevistados 1 e 2, ele gostou dessa ideia e completou falando: "Também seria legal comandos de voz que fazem ações rápidas e para digitar textos grandes".

Fonte: Elaborada pelo autor

Como conclusão dessa primeira etapa de entrevistas, foram definidos três requisitos que os entrevistados acreditam serem úteis para o uso no cotidiano, sendo eles:

Tabela 8 – Requisitos Levantados

#	Requisito	Justificativa
#1	Melhorar a verbosidade dos textos	Embora as resposta demonstrem pouca importância, mas os usuários também acreditam ser útil, só não é prioritário.
#2	Menu de ações rápidas	Todos os usuários gostaram da ideia.
#3	Preencher campos com comando de voz	Existem campos de texto que eles possuem dificuldade de preencher por exigirem uma quantidade de caracteres grande.

Fonte: Elaborada pelo autor

4.3 *Talkback*: Problemas e Limitações Encontradas

Como mostrado na Tabela 7, o *Talkback* possui alguns problemas e limitações. No entanto, os entrevistados também comentaram sobre outros empecilhos. Nesse sentido, os principais problemas enfrentados pelos usuários são:

- **Verbosidade**: O *Talkback* possui um nível de verbosidade que pode incomodar os usuários de forma nula ou parcial, ou seja, existem usuários que estão habituados e outros não. Dessa forma, o mais adequado é o *Talkback* prover um mecanismo para o usuário configurar o nível de verbosidade;
- **Consumo de Bateria**: Todos os entrevistados relataram que a bateria é drenada muito rápido quando comparado com o uso sem o *Talkback* habilitado. Esse problema é pelo fato da arquitetura possuir uma camada adicional (ver Figura 12) para captação e inscrição em todos os eventos de toques do usuário. Para resolver esse problema seria necessário refazer sua arquitetura;
- **Lentidão**: Os entrevistados também relataram que o *Talkback* é muito lento quando comparado com outras ferramentas, como o *VoiceOver*. Para melhorar a performance também seria necessário refazer a maior parte do leitor de tela e identificar detalhes de implementação que tornam o uso dele lento;
- **Menu de Ações**: O menu de ações foi uma das propostas que usuários mais gostaram. No *Talkback* também existe um menu para ações, porém a combinação, sequência e execução dos gestos dificultam muito o uso. Segundo os entrevistados, os gestos precisam ser

simples e citaram o exemplo do *VoiceOver* que usa uma roleta imaginária onde cada item é uma ação de utilidade; e

- **Comando de Voz**: Preencher campos com comando de voz é suportado pelo *Android* e *Talkback*, porém o usuário precisar abrir o teclado e navegar até o botão de gravação no teclado do telefone. Segundo os três entrevistados, eles não usam pela dificuldade, mas usariam caso o preenchimento por comando de voz fosse ativado pelo menu de ações rápidas.

No *blid reader* foi implementado o menu de ações e comando de voz. A verbosidade foi reduzida de modo a não prejudicar usuários habituados ao *Talkback* e melhorar o uso para quem se sente incomodado. Consumo de bateria e lentidão não foram abordados no presente trabalho. A seguir, será detalhado o projeto da solução, assim como cada requisito irá funcionar na ferramenta, a arquitetura e detalhes da implementação.

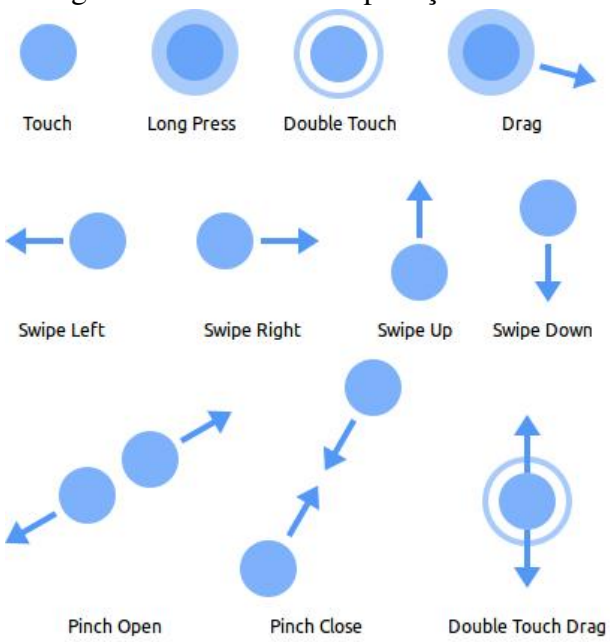
4.4 Projeto da Solução

Como já dito em seções anteriores, o trabalho proposto desenvolveu o *blid reader*, que é um leitor de tela para deficientes visuais. Além do *blid reader*, foi desenvolvido uma prova de conceito para validação (ver Seção 5.1). O *blid reader* usa os serviços de accessibilidades do *Android*. Os serviços de acessibilidades fornecem um mecanismo para criar gestos customizáveis, configurar a verbosidade da leitura e também criar novos eventos de acessibilidade para melhorar o *feedback* para o usuário. É importantes destacar que o *blid reader* não visa substituir o *Talkback*, mas sim validar se os problemas levantados são pertinentes e se ele resolve tais problemas.

4.4.1 Requisitos

Além dos requisitos iniciais extraídos nas entrevistas, foram elencadas mais alguns requisitos baseados em estudos das funcionalidades mais comuns entres outros leitores. Para o entendimento dos requisitos é necessário o conhecimento dos gestos (ver Figura 14).

Figura 14 – Gestos em aplicações móveis.



Fonte: <https://wireframesketcher.com/mockups/touch-gestures.html>

Os requisitos do leitor de tela são os seguintes:

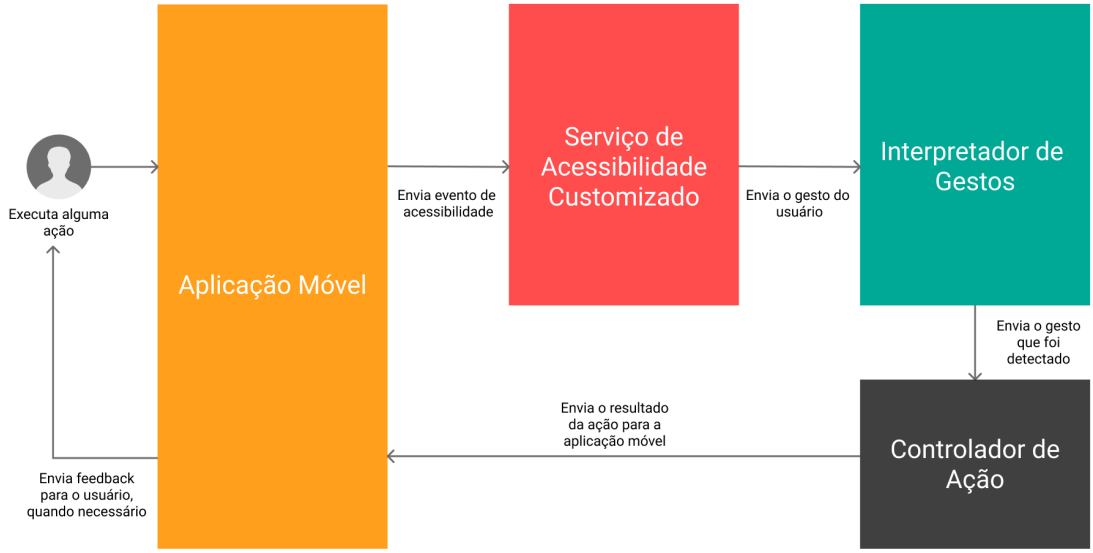
- **Navegação:** Usando os gestos de *Swipe Right* e *Swipe Left*, o usuário deve ser capaz de avançar e voltar o foco respectivamente em elementos na aplicação;
- **Scroll:** O usuário também será capaz de fazer *scroll* do conteúdo com os dois dedos pressionados;
- **Duplo clique:** O usuário ao executar um duplo clique escutará a real ação do elemento, ou seja, a ação que seria executada com um clique simples no uso sem acessibilidade;
- **Funções de utilidade:** Usando os gestos de *Swipe Up* e *Swipe Down*, o usuário poderá escolher uma função de utilidade para usar, as funções de utilidade são: Copiar o texto em foco, colar o texto no campo de texto em foco e abrir o reconhecedor de voz para preencher o campo de texto em foco;
- **Verbosidade:** Remover textos como *TextView*, *Button* ou *TextField* da leitura assistida de modo a deixar a leitura e entendimento melhor; e
- **Feedback em caixa de texto:** Uma vez que o usuário mover o foco para uma caixa de texto, o leitor deve deixar ele ciente se já há valor preenchido, qual esse valor e caso esteja vazio, informar para o usuário.

Vale ressaltar que o leitor desenvolvido nesse trabalho não possui requisito visual como já era esperado. O leitor consiste apenas em gestos e funcionalidades que não são visuais.

4.4.2 Arquitetura da Solução

A Figura 15 mostra a arquitetura da solução. Dentre os principais elementos do fluxo, estão:

Figura 15 – Arquitetura da Solução



Fonte: Elaborada pelo autor

- **Usuário**: O usuário é o ator principal, é ele que inicia o fluxo. Para desencadear o fluxo, o usuário pode executar qualquer um dos gestos descrito pela Figura 14, ou através de outras ações como comando de voz, teclado, chacoalhar ou usando os botões externos do *smartphone*. O usuário é também quem recebe o *feedback* ao final do fluxo, como mudança na interface, vibração ou som;
- **Aplicação Móvel**: A aplicação móvel (ao lado do sistema operacional) desempenha papel fundamental, pois é ela que recebe o gesto feito pelo usuário e encaminha para o serviço de acessibilidade adequado. Podem existir inúmeros serviços de acessibilidades voltados para propósitos distintos, assim a aplicação juntamente com o sistema operacional são encarregados de fazer esse roteamento;
- **Serviço de Acessibilidade**: Uma vez que a aplicação recebe o evento de acessibilidade, ele é delegado para o serviço de acessibilidade. No serviço de acessibilidade é onde as configurações de acessibilidade são feitas, como os gestos que serão responsabilidade do serviço em questão;
- **Interpretados de Gestos**: O serviço de acessibilidade é responsável por receber o gesto,

mas não deve ter a responsabilidade de interpretá-lo e retornar qualquer tipo de *feedback* ao usuário. Para isso, foi implementado um interpretador de gestos. Ele é responsável por receber um gesto qualquer e classificá-lo. Alguns gestos o sistema operacional já classifica com antecedência, como cliques, no entanto gestos complexos precisam de uma análise, como um gesto em forma Z; e

- **Controlador de Ação**: Uma vez que o gesto foi mapeado pelo interpretador de gesto, o controlador de ação é o componente responsável por executar a ação referente ao gesto detectado. A partir do gesto o controlador executa a ação correspondente e solicita a camada de aplicação móvel que transmita algum *feedback* ao usuário, caso necessário.

Para um maior entendimento da solução é apresentado o seguinte fluxo: quando o usuário executar alguma ação, essa ação é capturada pela aplicação, que por sua vez emite um evento de acessibilidade. Este evento será enviado para a solução, o qual um **Serviço de Acessibilidade Customizado** irá receber tais eventos e delegará para o **Interpretador de Gestos** a tarefa de identificar qual gesto o usuário acabou de executar. Quando o gesto for detectado, o **Controlador de Ação** vai recebê-lo como entrada e executar a tarefa relacionada aquele gesto, por exemplo, se o gesto interpretado for o *shake*, ou seja, chacoalhar o telefone, o controlador vai receber essa informação e vai emitir um resultado baseado no mapeamento do evento. Se o resultado é informar em que tela o usuário está, e estando ele na tela inicial, a aplicação móvel pode emitir esse resultado em forma de voz: “Você está na tela inicial”, para que usuários com deficiência visual recebam o *feedback* de forma adequada.

4.4.3 Ferramentas Utilizadas

O *blid reader*, como já mencionado, foi feito utilizando os serviços de accessibilidade do *Android* [1]. Além disso, outras tecnologias utilizadas foram:

- *Android Studio*: IDE oficial para desenvolvimento *Android*;
- Linguagem *Kotlin*: linguagem que possui suporte a parte majoritárias das novas *API* do *Android*; e
- *JUnit*: *Framework* para testes unitários automatizados para verificar se a implementação está correta e verificar a cobertura de código. O funcionamento do código não é garantido pelo *lint* do *Android Studios*, a parte funcional é assegurada pela existência de testes automatizados.

[1] <https://developer.android.com/guide/topics/ui/accessibility/service>

O anexo B contém um link para código fonte do *blid reader*.

4.5 Considerações Finais

Esse Capítulo apresentou o trabalho proposto, que é a construção do *blid reader*, um leitor de tela para deficientes visuais na plataforma *Android*. Foi apresentado como foram realizadas as entrevistas iniciais, bem como seus resultados. Também foi apresentado o projeto e desenvolvimento da solução. O próximo capítulo apresenta a prova de conceito desenvolvida e os resultados das entrevistas realizadas com a solução proposta.

5 RESULTADOS

Neste Capítulo serão apresentados os resultados da pesquisa. A Seção 5.1 apresenta a prova de conceito desenvolvida. A Seção 5.2 apresenta os resultados encontrados.

5.1 Prova de Conceito

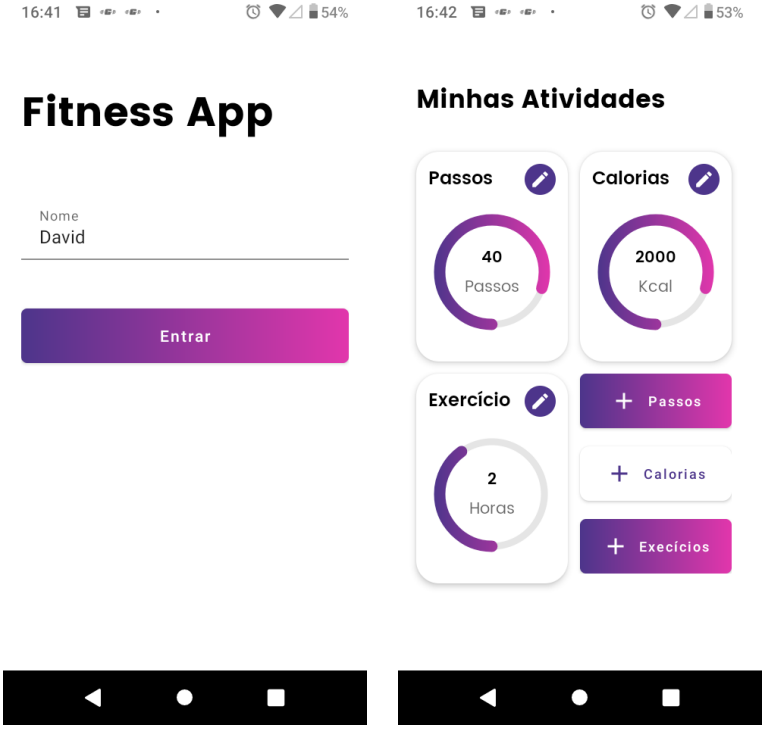
A segunda etapa da fase de desenvolvimento foi a construção da prova de conceito. Na prova de conceito foram utilizadas as mesmas tecnologias que a etapa de desenvolvimento da ferramenta. Além disso, as funcionalidades da prova de conceito foram modeladas com intuito de explorar os problemas na ferramenta padrão *Talkback*, segundo os requisitos.

A prova de conceito é uma aplicação para organizar as atividades físicas, ou seja, uma aplicação *fitness*. Essa aplicação foi baseada em outra prova de conceito que busca avaliar como ferramentas híbridas se comportam mediante a falta de acessibilidade fornecida pelos desenvolvedores, conforme (BEZERRA, 2021). As principais funcionalidades dessa prova de conceito são:

- Adicionar o nome da tela inicial;
- Adicionar um objetivo da semana para quantidades de quilômetros percorridos;
- Verificar quanto falta para o objetivo;
- Alterar o objetivo;
- Copiar o progresso atual em forma de texto; e
- Suporte apenas aos dados de caminhada.

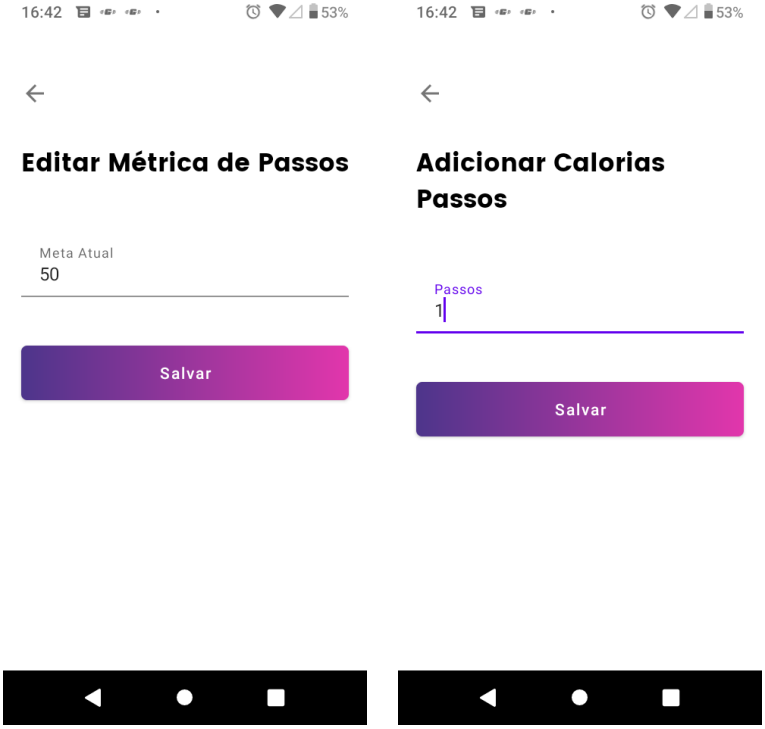
A Figura 16 mostra a tela inicial onde o usuário poderá inserir nome dele e a tela de *dashboard* onde ele gerencia o progresso do treino.

Figura 16 – Prova de conceito



Fonte: Elaborada pelo autor

Figura 17 – Prova de conceito



Fonte: Elaborada pelo autor

De acordo com as Figuras 16 e 17, é possível notar que a aplicação foi projetada exatamente para testar as funcionalidades desenvolvidas no *blid reader*, afinal o objetivo é validar se ele resolve os problemas levantados na Tabela 7 através dos requisitos listados na Seção 4.4.1. Por exemplo, o usuário poderá testar o comando de voz para inserir números e o nome dele, copiar e colar valores entre os exercícios e metas, além de explorar a tela inicial e o *feedback* dos elementos.

Além disso, cabe ressaltar que as aplicações são independentes e completamente distintas, isto é, o *blid reader* pode ser usado em outras aplicações, assim como a prova de conceito pode ser usada sem o *blid reader*. Com a prova de conceito construída, foi realizada a etapa de teste com usuário. O anexo B contém um link para o aplicativo da prova de conceito embutido com o *blid reader*, embora não seja necessário.

5.2 Testes e Análises dos resultados

Essa etapa tem como objetivo validar se os requisitos e hipóteses levantadas na etapa de entrevistas, assim como dos estudos, estavam corretos, e além disso se os problemas foram resolvidos de forma total ou parcial pela solução desenvolvida. Para chegar em tal conclusão, a etapa de testes foi dividida em duas partes, que são:

- Testes com o (*TalkBack*); e
- Teste com o *blid reader*.

Cada usuários executou os testes duas vezes: em uma delas com o leitor de tela *TalkBack* e na outra o *blid reader*. Vale destacar que, o teste usando o *TalkBack* não foi feito com falta de acessibilidade proposital, pelo contrário, usará os recursos de acessibilidade padrão, ou seja, os textos alternativos.

Para garantir a acessibilidade padrão, no teste com o *TalkBack* foi executada uma inspeção de usabilidade usando o aplicativo *Scanner* de acessibilidade, ele verifica o suporte a recursos básicos de acessibilidade conforme a seção 2.4.2. Como resultado da inspeção foi mostrado que todos os elementos possuíam suporte a acessibilidade, isso garante que o teste com *TalkBack* não foi feito sem acessibilidade de maneira proposital.

Uma vez que, o leitor de tela estava habilitado foi solicitado que cada usuário realizasse uma série de tarefas, sendo elas:

- A partir da tela de configuração do leitor, abrir o aplicativo *fitness*;
- Inserir seu nome na tela inicial;

- Explorar a tela "Minhas Atividades", solicitando uma descrição da tela;
- Editar a meta de passos, calorias e exercícios;
- Adicionar nova entrada de dados para passos, calorias e execícios; e
- Copiar valores de passos para uma contato qualquer no aplicativo *whatsapp*.

Para um melhor entendimento, o anexo B contém um link para um vídeo de demostração de uso do *blid reader* como leitor de tela.

Também foi solicitado que, no *blid reader*, cada caixa de texto deveria ser preenchida usando o comando de voz para preenchimento dos dados. Além disso, a sequência de teste durou em média 10 minutos para cada usuário, ele também ficou livre para exploração depois da execução das tarefas listadas acima.

Conforme a tabela 6 das primeiras entrevistas, os testes possuem as seguintes finalidades:

- Avaliar verbosidade das informações;
- Dificuldade para inserir ou modificar campos de texto;
- Dificuldade para copiar e colar as metas e valores; e
- *Feedback* nas caixas de textos.

A ordem dos testes foi aleatória, ou seja, alternando entre o *TalkBack* e o *blid reader* como primeiro teste, pois caso o *TalkBack* sempre seja utilizada primeiro, o usuário provavelmente ficaria mais familiarizado quando for utilizar o *blid reader*, pois o usuário já conhece as funcionalidade do primeiro teste com o *TalkBack*. Uma vez que os testes foram executados, o próximo passo é a avaliação dos resultados. Os resultados foram avaliados de forma geral, ou seja, não apenas os resultados do testes com o *TalkBack* e *blid reader*, mas sim dos dados coletados durante toda a pesquisa.

5.2.1 Testes e Resultados

Essa Seção apresenta os resultados das entrevistas referente ao uso e testes do *TalkBack* e *blid reader*. Na Seção 4.2 foi mostrado o resultado das entrevistas iniciais, ou seja, as entrevistas para levantamento de requisitos. Elas foram feitas antes da concepção da ferramenta e da aplicação da prova de conceito mostrado anteriormente. Os resultados apresentados, a seguir, são com novos usuários, assim é possível comprovar se os problemas são compartilhados entre os demais usuários de ferramentas assistidas foi mitigado.

As entrevistas foram feitas de maneira presencial com um total de 3 participantes.

Todos os participantes, assim como nas entrevistas da Seção 4.2, possuíam nível de experiência com ferramentas de acessibilidade avançado e já tiveram experiência nas plataformas *Android*, *iOS* e *Windows*, conforme o perfil descrito na tabela 5. Assim como nas entrevistas iniciais, as perguntas e respostas foram enquadradas em temas baseada nas suas semelhanças entre si. As perguntas originais podem ser encontradas no Anexo A.2.

As tabelas 9, 10 e 11 apresentam os resultados finais. Cada tabela é um comparativo entre o sentimento do usuário usando o leitor *blid reader* e o *TalkBack*. Além disso, também há observações gerais sobre que o usuário fez durante os testes.

Tabela 9 – Resultado finais - Entrevistado 1

Entrevistado 1		
#	*Blid Reader*	*TalkBack*
Verbosidade das informações	O usuário gostou por não repetir textos como *TextView* e *Button*.	O usuário informou que não se incomoda tanto pelo tempo de uso com o *TalkBack*, mas gostou mais da ferramenta desenvolvida.
Inserção de textos	O usuário gostou muito da abordagem de comando de voz, principalmente para preencher respostas longas quando não há a opção de gravar áudio disponível.	O usuário por ter conhecimento no teclado *Braille* não teve tanta dificuldade, mas confessou ser bem mais prático usando comando de voz.
Copiar e Colar	O usuário achou bem simples copiar e colar textos usando os gestos de *Swipe Left* e *Swipe Right* mostrado na Figura 14.	O usuário sentiu dificuldade, pois o *TalkBack* mostrava um menu na tela e ele tinha que fazer vários gestos seguidos com chance alta de erro.
Feedback em caixas de texto	O usuário gostou muito, pois fornece uma informação precisa do que está preenchido e depois que ele fala no comando de voz, o leitor também informa que a caixa de texto teve seu valor alterado para o valor em questão	O usuário teve uma certa dificuldade para saber o valor preenchido e algumas vezes teve que apagar e inserir novamente porque não sabia o valor que havia no campo de texto.
Observações: Como consideração geral, o usuário preferiu o *blid reader*, no caso para a prova de conceito. Se fosse uma aplicação bancária, utilizando a funcionalidade de comando de voz para preencher a senha, não seria adequado.		

Fonte: Elaborada pelo autor

Tabela 10 – Resultado finais - Entrevistado 2

Entrevistado 2		
#	*Blid Reader*	*TalkBack*
Verbosidade das informações	O usuário achou a leitura dos elementos bem mais limpa.	O usuário, assim como o anterior, informou que não incomoda os textos do *TalkBack*, mas prefere a ferramenta desenvolvida.
Inserção de textos	Assim como o usuário anterior achou a funcionalidade de comando de voz bem útil, principalmente por que depois do preenchimento ele fornece esse *feedback*.	O usuário não possuía conhecimento no teclado *Braille*, pois usava o computador com mais frequência, sendo assim teve um pouco de dificuldade e achou bem mais prático a ferramenta desenvolvida.
Copiar e Colar	O usuário também gostou das ações de copiar e colar textos usando os gestos de *Swipe Left* e *Swipe Right*.	Assim como o usuário anterior, também sentiu dificuldade, pois é uma sequência de gestos para copiar e colar.
Feedback em caixas de texto	O usuário gostou bastante, assim como o anterior, ele gostou do *feedback* do que havia preenchido na caixa de texto.	O usuário não teve tanta dificuldade, segundo ele, é comum ter que memorizar o que preencheu em formulários, porque o *TalkBack* não diz o que está no campo de texto.
Observações: Como consideração geral, o usuário preferiu o *blid reader*, apontou o mesmo problema de reconhecimento de voz com informações sensíveis e em lugares com muito ruído que pode prejudicar o reconhecimento.		

Fonte: Elaborada pelo autor

Tabela 11 – Resultado finais - Entrevistado 3

Entrevistado 3		
#	*Blid Reader*	*TalkBack*
Verbosidade das informações	O usuário achou a leitura dos elementos bem limpa.	O usuário assim como os anteriores, informou que não incomoda os textos do *TalkBack*, mas prefere a ferramenta desenvolvida.
Inserção de textos	O usuário gostou muito do comando de voz.	O usuário não sentiu dificuldade pois possuía conhecimento no teclado *Braille*, mas prefere comando de voz para textos longos.
Copiar e Colar	O usuário também gostou das ações de copiar e colar textos.	Assim como os usuários anteriores, sentiu dificuldade, pois é uma sequência de gestos para copiar e colar.
Feedback em caixas de texto	O usuário gostou bastante do *feedback*, sendo o diferencial.	O usuário teve um pouco dificuldade, segundo ele, porque a aplicação de prova de conceito tem algumas caixas de texto com valor já preenchido. Para ele, o *feedback* é essencial.
Observações: Como consideração geral, o usuário preferiu o *blid reader*, apontou o mesmo problema de reconhecimento de voz com informações sensíveis e em lugares com muito ruído que pode prejudicar o reconhecimento. Além disso, também mencionou que a aplicação falava um texto “Avance para continuar”, segundo ele, era um texto estranho.		

Fonte: Elaborada pelo autor

5.3 Discussões Gerais

A análise do resultado foi feita de maneira subjetiva e sempre buscando entender o sentimento do usuário ao usar os dois leitores. De maneira geral, todos os usuários concordaram que o *blid reader* possui funcionalidades mais interessantes do que o *talkBack*, isso era um resultado esperado, pois os requisitos foram levantados visando pontos fracos do *talkBack*.

Com relação ao tema "Verbosidade das Informações", os três entrevistados gostaram da forma como os textos estavam sendo pronunciados pelo *blid reader*. É importante lembrar que o *blid reader* não criou uma nova síntese de voz. A remoção de textos como *button* e *textview* e adição de conectivos visando tornar a descrição mais clara como "Botão para adicionar calorias" foram aspectos determinantes na visão dos entrevistados. Como mencionado anteriormente, alguns usuários não consideram um problema a leitura verbosa e repetitiva do *TalkBack*, assim como usuários inciantes precisam de tal verbosidade para fixar o entendimento da ferramenta. Em ambos os casos, o ideal é existir a possibilidade de configurar o nível de verbosidade de acordo com o gosto de cada usuário e nível de experiência.

A "Inserção de Textos" é um tema bastante recorrente na acessibilidade para deficientes visuais. Existem várias alternativas, como teclado *braile* e o próprio teclado digital do dispositivo móvel, no entanto, a funcionalidade de inserção de textos usando comando de voz foi considerada pelos entrevistados pertinente e útil no dia-a-dia. Por exemplo, no momento de responder pesquisas subjetivas com perguntas abertas é mais simples gravar um áudio em vez de escrever, também existem locais que suportam apenas textos e a funcionalidade desenvolvida na ferramenta foca nesses casos, onde o usuário deseja falar algo rápido via comando de voz e converter em texto posteriormente. Vale ressaltar que existem situações que inviabilizam o seu uso. Por exemplo, na inserção de senhas bancárias ou informações sensíveis que não devem ser pronunciadas. No entanto, essa funcionalidade é opcional, ou seja, o usuário pode ativar ou não de acordo com a necessidade, reforçando a importância da customização e autonomia no uso das aplicações.

O "Copiar e Colar" foi o que os entrevistados mais gostaram, pois é um problema cotidiano deles, sempre há uma necessidade de transitar dados de uma aplicação para outra e isso é feito usando a funcionalidade de copiar e colar do sistema operacional. No *TalkBack*, existe um atalho para copiar, porém segundo os entrevistados, esse gesto é complexo e quando comparado com o *blid reader*, os entrevistados preferiam o *blid reader* em razão da praticidade nos gestos e também na possibilidade de copiar textos que estão fora de caixa de texto, como títulos e

parágrafos. Dois dos entrevistas mencionaram que gostaria de uma função para copiar usando seleção de texto, ou seja, escolher partes de um parágrafo em vez do parágrafo por completo.

Por fim, o "*feedback* em Caixas de Texto", que foi um dos problemas levantados nas primeiras entrevistas, mostrou que informar o estado das caixas de textos é importante para os usuários, por exemplo, se tivesse um campo para receber a idade do usuário, o leitor do *TalkBack* pronunciava apenas a descrição do elemento "Caixa de texto idade", enquanto que o *blid reader* informava a descrição e o valor preenchido "Caixa de texto para idade com o valor 22" ou "Caixa de texto para idade sem valor preenchido". Para campos simples, os usuários conseguem lembrar dos valores inseridos, porém existe a chance de ocorrer falhas na inserção dos valores e depois não há como, no *TalkBack*, verificar o que foi inserido sendo necessário preencher novamente.

De acordo com os quatro temas discutidos acima, é possível concluir que qualquer funcionalidade deve ser adicionada com o máximo de customização e o mais configurável possível, além de não atrapalhar na autonomia e usabilidade da aplicação, ou seja, não adicionar opções obrigatórias, sempre deixando usuário escolher o que ele deseja usar e quando. Além disso, a praticidade e facilidade para execução de atalhos via gestos é desejada entre os usuários, no entanto, por possuir uma variedade de opções, o *TalkBack* possui gestos complexos que dificultam o aprendizado e sua execução.

5.4 Considerações Finais

Este Capítulo apresentou a prova de conceito e a implementação proposta do *blid reader*, além disso também descreveu a metodologia e execução dos testes finais com usuários. Os testes buscavam verificar se os requisitos foram contemplados. O próximo capítulo apresenta a conclusão do trabalho, assim como as limitações encontradas e os trabalhos futuros.

6 CONCLUSÃO E TRABALHOS FUTUROS

Este Capítulo resume o que foi debatido ao longo do presente trabalho de conclusão de curso. Na Seção 6.1 é apresentado um resumo dos resultados obtidos. A Seção 6.2 apresenta as limitações para desenvolver e avaliar o *blid reader*. Por fim, a última Seção apresenta os trabalhos futuros.

6.1 Resultados Alcançados

De acordo com os resultados obtidos em 5.2, o *blid reader* atende aos requisitos levantados na Seção 4.4.1. Além disso, também foi comprovado que tais problemas são compartilhados entre vários usuários que usam ferramentas leitoras de tela, pois os usuários das entrevistas feitas na Seção 4.2 são diferentes das entrevistas da Seção 5.2. Alguns problemas foram oriundos do trabalho de Rodrigues *et al.* (2015b) e complementaram o trabalho proposto.

Apesar dos usuários apontarem alguns problemas, a maioria deles é possível resolver de forma pontual, por exemplo, o comando de voz pode ser uma opção e não obrigação, assim como os *feedbacks*. Os usuários também informaram a utilidade de customização, ou seja, o usuário escolher quais funcionalidade ele quer ou não habilitadas no leitor de tela.

6.2 Limitações

Ao longo do presente trabalho foram encontradas algumas limitações, as quais impediram o andamento desta pesquisa.

A primeira limitação é a documentação do sistema operacional *Android* para criação de ferramentas de acessibilidade. Na documentação oficial, existe apenas o artigo *Criando seu próprio serviço de acessibilidade*[1]. Vale a pena destacar que a documentação é completa quando referente a como criar aplicação acessíveis, mas não em como criar leitores de telas ou ferramentas de acessibilidade que possam ser usadas em aplicações externas.

Outra limitação constatada foi o perfil dos entrevistados e a quantidade. Foram apenas seis entrevistados no total das duas entrevistas e todos com perfil intermediário para avançado. Essa limitação também é decorrente da localidade em que foi feita a pesquisa, bem como da dificuldade para executar os testes. As primeiras entrevistas, dos resultados iniciais, foram realizadas remotamente, o que prejudica a avaliação como um todo sobre as percepções

[1] <https://developer.android.com/guide/topics/ui/accessibility/service?hl=pt-br>

dos usuários. As entrevistas para os resultados finais, com o *blid reader*, foram realizadas presencialmente, mas a quantidade de pessoas dispostas a participar foi reduzida.

6.3 Trabalhos Futuros

Para os trabalhos futuros forma escolhidos temas que buscam aperfeiçoar a pesquisa feita no presente trabalho, tais como:

- Implementar a versão da ferramenta para *Desktop* e *iOS*;
- Refazer as entrevistas com um grupo maior e mais diversificado quanto ao perfil do usuário;
- Estudo comparativo da ferramenta com outros leitores como o *VoiceOver* no *iOS*;
- Implementar a versão da ferramenta voltada para usuários com deficiência motora; e
- Realizar análise comparativa de consumo energético da ferramenta e de outros leitores.

REFERÊNCIAS

ALAJARMEH, N. fnonvisual access to mobile devices: A survey of touchscreen accessibility for users who are visually impaired. **Displays**, Elsevier, p. 102081, 2021.

ALSHAYBAN, A.; AHMED, I.; MALEK, S. Accessibility issues in android apps: State of affairs, sentiments, and ways forward. In: **2020 IEEE/ACM 42nd International Conference on Software Engineering (ICSE)**. [S.l.: s.n.], 2020. p. 1323–1334.

ALSHAYBAN, A.; AHMED, I.; MALEK, S. Accessibility issues in android apps: State of affairs, sentiments, and ways forward. In: **2020 IEEE/ACM 42nd International Conference on Software Engineering (ICSE)**. [S.l.: s.n.], 2020. p. 1323–1334.

ARRUDA, K. S. d. G. Victor Leal Porto de A. Um estudo sobre a relação dos desenvolvedores mobile com as diretrizes de acessibilidade para deficientes visuais. 2020.

ASSOCIAçãO BRASILEIRA DE NORMAS TéCNICAS. **Acessibilidade em aplicativos de dispositivos móveis — Requisitos**: Acessibilidade em aplicativos de dispositivos móveis — requisitos. Av. das Nações Unidas, 18801 - Cj. 1501 - São Paulo - SP | CEP 04795-000 - Brasil, 2022. 35 p.

BALANSIN, J. B. C. F. Especificação e implementação de um leitor de tela. 2011.

BEZERRA, W. V. F. S. R. Desenvolvimento nativo vs ionic vs react native: uma analise comparativa do suporte a acessibilidade em android. 2021.

BICKENBACH, J. The world report on disability. **Disability & Society**, Routledge, v. 26, n. 5, p. 655–658, 2011. Disponível em: <https://doi.org/10.1080/09687599.2011.589198>.

BRASIL. **Decreto Nº 5.296 de 2 de dezembro de 2004**. 2004. Decreto que regulariza os tipo de deficiência no Brasil. Disponível em: <http://www.planalto.gov.br/ccivil_03/_ato2004-2006/2004/decreto/d5296.htm>. Acesso em: 02 Out. 2021.

BRIEDE-WESTERMEYER, J. C.; PACHECO-BLANCO, B.; LUZARDO-BRICEÑO, M.; PÉREZ-VILLALOBOS, C. Mobile phone use by the elderly: Relationship between usability, social activity, and the environment. **Sustainability**, Multidisciplinary Digital Publishing Institute, v. 12, n. 7, p. 2690, 2020.

CASILARI, E.; SANTOYO-RAMÓN, J. A.; CANO-GARCÍA, J. M. Analysis of a smartphone-based architecture with multiple mobility sensors for fall detection. **PLoS one**, Public Library of Science San Francisco, CA USA, v. 11, n. 12, p. e0168069, 2016.

CISCO, C. V. N. I. Global mobile data traffic forecast update, 2016–2021. **white paper**, 2017.

COUNCIL, A. B. *et al.* Accessibility. **Retrieved March**, v. 10, 2005.

DÍAZ-BOSSINI, J.-M.; MORENO, L. Accessibility to mobile interfaces for older people. **Procedia Computer Science**, Elsevier, v. 27, p. 57–66, 2014.

FAÇANHA, A. R.; DARIN, T.; VIANA, W.; SÁNCHEZ, J. O&m indoor virtual environments for people who are blind: A systematic literature review. **ACM Transactions on Accessible Computing (TACCESS)**, ACM New York, NY, USA, v. 13, n. 2, p. 1–42, 2020.

FERATI, L. S. M. Automatic adaptation techniques to increase the web accessibility for blind users. 2016.

GOV.BR. **Acessibilidade Digital - Governo Digital**. 2022. Disponível em: <https://www.gov.br/governodigital/pt-br/acessibilidade-digital>. Acesso em: 22 Fev. 2023.

HENRY, S. L. Understanding web accessibility. 2006.

IBGE, I. B. de Geografia e E. **Censo Demográfico - Característica Gerais da População, Religião e Pessoas com Deficiência.** [S.l.: s.n.], 2010. ISSN 0104-3145.

JONES, S. Accessibility measures: a literature review. **Publication of: Transport and Road Research Laboratory**, n. TRRL LR 967 Monograph, 1981.

KIM, B. Responsive web design, discoverability, and mobile challenge. **Library technology reports**, v. 49, n. 6, p. 29–39, 2013.

LEAL, D. **Classes de Acuidades Visual - Classificação ICD-9-CM (WHO/ICO)**. 2017. Sociedade Brasileira de Visão Subnormal. São Paulo. Disponível em: <https://www.cbo.com.br/subnorma/conceito.htm>. Acesso em: 27 Ago. 2021.

MARTÍNEZ-PÉREZ, B.; TORRE-DÍEZ, I. D. L.; LÓPEZ-CORONADO, M. Privacy and security in mobile health apps: a review and recommendations. **Journal of medical systems**, Springer, v. 39, n. 1, p. 1–8, 2015.

MULLINS, C. Responsive, mobile app, mobile first: untangling the ux design web in practical experience. In: **Proceedings of the 33rd Annual International Conference on the Design of Communication**. [S.l.: s.n.], 2015. p. 1–6.

NAYEBI, F.; DESHARNAIS, J.-M.; ABRAN, A. The state of the art of mobile application usability evaluation. In: IEEE. **2012 25th IEEE Canadian Conference on Electrical and Computer Engineering (CCECE)**. [S.l.], 2012. p. 1–4.

RODRIGUES, A.; MONTAGUE, K.; NICOLAU, H.; GUERREIRO, T. Getting smartphones to talkback: Understanding the smartphone adoption process of blind users. In: **Proceedings of the 17th international acm sigaccess conference on computers & accessibility**. [S.l.: s.n.], 2015. p. 23–32.

RODRIGUES, A.; NICOLAU, H.; MONTAGUE, K.; GUERREIRO, J.; GUERREIRO, T. J. Open challenges of blind people using smartphones. **CoRR**, abs/1909.09078, 2015. Disponível em: <http://arxiv.org/abs/1909.09078>.

RUIZ, J.; LI, Y.; LANK, E. User-defined motion gestures for mobile interaction. In: **Proceedings of the SIGCHI Conference on Human Factors in Computing Systems**. New York, NY, USA: Association for Computing Machinery, 2011. (CHI '11), p. 197–206. ISBN 9781450302289. Disponível em: <https://doi.org/10.1145/1978942.1978971>.

SCHNELLE-WALKA, D.; ALAVI, A.; OSTIE, P.; MÜHLHÄUSER, M.; KUNZ, A. A mind map for brainstorming sessions with blind and sighted persons. In: SPRINGER. **International Conference on Computers for Handicapped Persons**. [S.l.], 2014. p. 214–219.

SIDI. **SIDI - Guia para o desenvolvimento de Aplicações Móveis Acessíveis**. 2015. Disponível em: <https://www.sidi.org.br/guiadeacessibilidade/index.html#sobre>. Acesso em: 02 Out. 2021.

SIEBRA, C.; ANJOS, M.; FLORENTIN, F.; GOUVEIA, T.; FILHO, A.; CORREIA, W.; PENHA, M.; SILVA, F. Q.; SANTOS, A. L. Accessibility devices for mobile interfaces extensions: A survey. In: **Proceedings of the 17th International Conference on Human-Computer Interaction with Mobile Devices and Services Adjunct**. [S.l.: s.n.], 2015. p. 644–651.

SILVA, A. D. A. P. D. Design responsivo: técnicas, frameworks e ferramentas. 2014.

SILVA, C. A.; OLIVEIRA, A. F. B. A. de; MATEUS, D. A.; COSTA, H. A. X.; FREIRE, A. P. Types of problems encountered by automated tool accessibility assessments, expert inspections and user testing: A systematic literature mapping. In: **Proceedings of the 18th Brazilian Symposium on Human Factors in Computing Systems**. New York, NY, USA: Association for Computing Machinery, 2019. (IHC '19). ISBN 9781450369718. Disponível em: <https://doi.org/10.1145/3357155.3358479>.

TAIVALSAARI, A.; MIKKONEN, T. On the development of iot systems. In: IEEE. **2018 Third International Conference on Fog and Mobile Edge Computing (FMEC)**. [S.l.], 2018. p. 13–19.

TAKAGI, H.; SAITO, S.; FUKUDA, K.; ASAKAWA, C. Analysis of navigability of web applications for improving blind usability. 2007.

TAM, C.; SANTOS, D.; OLIVEIRA, T. Exploring the influential factors of continuance intention to use mobile apps: Extending the expectation confirmation model. **Information Systems Frontiers**, Springer, v. 22, n. 1, p. 243–257, 2020.

TECHTRENDS. Four types of disabilities: Their impact on online learning. 2008.

TURNER-MCGRIEVY, G. M.; HALES, S. B.; SCHOFFMAN, D. E.; VALAFAR, H.; BRAZENDALE, K.; WEAVER, R. G.; BEETS, M. W.; WIRTH, M. D.; SHIVAPPA, N.; MANDES, T. *et al.* Choosing between responsive-design websites versus mobile apps for your mobile behavioral intervention: presenting four case studies. **Translational behavioral medicine**, Oxford University Press, v. 7, n. 2, p. 224–232, 2017.

VENDOME, C.; SOLANO, D.; LIñáN, S.; LINARES-VáSQUEZ, M. Can everyone use my app? an empirical study on accessibility in android apps. In: **2019 IEEE International Conference on Software Maintenance and Evolution (ICSME)**. [S.l.: s.n.], 2019. p. 41–52.

VIDAL, P. V. C. Dependência mobile: a relação da nova geração com os gadgets móveis digitais. 2015.

WCAG. **Distinguishable: Understanding Guideline 1.4**. 2016. Disponível em: <https://www.w3.org/TR/UNDERSTANDING-WCAG20/visual-audio-contrast.html>. Acesso em: 02 Out. 2021.

WCAG. **Introduction to Understanding WCAG 2.1**. 2016. Disponível em: <https://www.w3.org/WAI/WCAG21/Understanding/intro#understanding-the-four-principles-of-accessibility>. Acesso em: 07 Out. 2021.

WCAG. **Understanding Success Criterion 1.4.2: Audio Control**. 2016. Disponível em: <https://www.w3.org/WAI/WCAG21/Understanding/audio-control.html>. Acesso em: 02 Out. 2021.

WCAG. **Understanding Success Criterion 1.4.6: Contrast (Enhanced)**. 2016. Disponível em: <https://www.w3.org/WAI/WCAG21/Understanding/contrast-enhanced.html>. Acesso em: 02 Out. 2021.

WCAG. **Understanding Success Criterion 2.4.1: Bypass Blocks**. 2016. Disponível em: <https://www.w3.org/WAI/WCAG21/Understanding/bypass-blocks.html>. Acesso em: 02 Out. 2021.

WCAG. **Understanding Success Criterion 2.4.8: Location**. 2016. Disponível em: <https://www.w3.org/WAI/WCAG21/Understanding/location.html>. Acesso em: 02 Out. 2021.

ZEMEL, T. **Web Design Responsivo: páginas adaptáveis para todos os dispositivos**. [S.l.]: Editora Casa do Código, 2015.

ZHU, H.; XIONG, H.; GE, Y.; CHEN, E. Mobile app recommendations with security and privacy awareness. In: **Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining**. [S.l.: s.n.], 2014. p. 951–960.

APÊNDICE A – FORMULÁRIOS

Os formulários serão apresentados em forma de tabela com a descrição da pergunta e com seu respectivo tipo: resposta livre, múltipla escolha, avaliação e sim ou não. As peguntas da avaliação são perguntas de nível de satisfação do usuário com o valor variando entre 0 e 5, onde 5 é o nível satisfação máxima.

A.1 Formulário das entrevistas iniciais

Essa Seção apresenta o formulários com as peguntas feitas na entrevistas iniciais para levantamento de requisitos. As pergunta extraídas do trabalho de Rodrigues *et al.* (2015b) estão marcadas com *****.

Tabela 12 – Formulário das perguntas iniciais

Descrição	Tipo de questão
Quando está usando uma aplicação qualquer e surge a necessidade de buscar uma opção que deseja, você precisa navegar e passar por todos os elementos para chegar nela, mesmo que sejam elementos que não possuam relação com o que está procurando? *	Múltipla escolha: 1. Sim e considero um incômodo ter que passar por todos os elementos para encontrar algo; 2. Sim, mas não considero um incomodo; 3. Não; e 4. Outro (resposta livre).
Ainda sobre a primeira questão, caso acredite ser um incômodo, que outros aspectos você gostaria de compartilhar sobre encontrar uma opção em determinada parte da aplicação?	Resposta livre
Ao acessar uma aplicação nova, para descobrir sobre os recursos principais da aplicação, você navega por todos os elementos ou o leitor fornece uma forma de você ter um resumo da tela que você está em questão? *	Múltipla escolha: 1. Sim e acho ruim não ter um resumo da tela. Elementos para encontrar algo; 2. Sim, mas não me importo em ter que ir procurando e descobrindo os elementos; 3. Não; e 4. Outro (resposta livre).
Ao navegar em uma aplicação, você fica perdido e não consegue mais voltar para um local conhecido? Por exemplo, foi navegando e não sabe em que tela está. *	Sim ou Não
O leitor de tela fornece uma forma de dizer em qual tela você está?	Sim ou Não (com comentário)

O leitor de tela fornece uma forma de voltar para a tela inicial, com o intuito de começar de novo a navegação a partir de um ponto conhecido?	Sim ou Não (com comentários)
Sobre a verbosidade do leitor de tela, ou seja, quando ele fala "Clique no botão login". Como você avalia ela? *	Múltipla escolha: 1. Muito verboso e me incomoda; 2. Muito verboso, mas não me importo; 3. Pouca verbosidade; e 4. Nenhuma verbosidade.
Sobre o *feedback* do leitor de tela ao clicar em botão, por exemplo, como você considera o *feedback*?	Múltipla escolha: 1. Ele fornece o *feedback* sobre uma descrição do botão, o que vai acontecer se ele clicar e um *feedback* quando o botão é pressionado; 2. Ele fornece o *feedback* sobre uma descrição do botão, mas não diz o que vai acontecer e nem informa quando eu clico nele; e 3. Não fornece nenhum *feedback*.
Caso tenha alguma consideração extra, pode está descrevendo aqui.	Resposta livre

Fonte: Elaborada pelo autor

A.2 Formulário das entrevistas finais

Essa Seção apresenta o formulário com as entrevistas finais que avaliaram a ferramentas desenvolvida no presente trabalho, a comparação com o *TalkBack* e o cumprimento dos requisitos levantas na Seção 4.4.1. Vale destacar que algumas perguntas foram feitas mais de uma vez em áreas diferentes da aplicação (marcadas com **(+)**).

Tabela 13 – Formulário das Entrevistas Finais

Descrição	Tipo de questão
Como foram as interações com os gestos na tela?	Resposta livre
Quais as primeiras impressões ao usar a ferramenta?	Resposta livre
Como estão os *feedbacks* do usuário?	Resposta livre
Facilidade para preencher o campo de texto referente ao nome.	Resposta avaliação
Feedback do campo de texto quando ele possui ou não valor.	Resposta avaliação
Feedback quando a tela abre.*(+)*	Resposta avaliação
Facilidade para copiar valor atual das calorias.	Resposta avaliação
Feedback dos botões.*(+)*	Resposta avaliação
Facilidade para inserir o valor no campo de texto.*(+)*	Resposta avaliação
Feedback para colar um valor no campo de texto.*(+)*	Resposta avaliação
Verbosidade da descrição dos elementos.*(+)*	Resposta avaliação
Descrição dos elementos.*(+)*	Resposta avaliação
Baseado na sensação do uso, qual foi mais rápido?	Múltipla escolha • Ferramenta proposta; ou • *Talkback*.
Baseado na prova de conceito, qual foi melhor?	Múltipla escolha • Ferramenta proposta; ou • *Talkback*.
Funcionalidade de comando de voz é útil?	Resposta avaliação
Atalho para copiar e colocar é útil?	Resposta avaliação
Considerando a prova de conceito e os gestos, qual foi melhor?	Múltipla escolha • Ferramenta proposta; ou • *Talkback*.
Considerações finais.	Resposta livre

Fonte: Elaborada pelo autor

APÊNDICE B – APLICATIVO E DEMOSTRAÇÃO

O código fonte do *blid reader* pode ser encontrado nesse link: <https://github.com/david32145/TCCII>.

Para *download* do aplicativo que contém tanto o leitor *blid reader* como a prova de conceito, acesse o seguinte link: <https://github.com/david32145/TCCII/raw/master/Fitness_App_base.apk>.

Para acesso do vídeo de uso demostrativo do *blid reader*, acesse o seguinte link: <https://drive.google.com/file/d/1d-bEZ3KnGCTRpoGWBk7Fwfe1QCSuTfWy/view?usp=sharing>