Universidade de Brasília – UnB
Faculdade UnB Gama – FGA
Engenharia de Software

Catálogo de Práticas de Acessibilidade: Um Apoio *Online* Voltado à Acessibilidade da Web

Autor: Felipe de Oliveira Hargreaves
Orientadora: Profa. Dra. Milene Serrano

Brasília, DF
2021

Felipe de Oliveira Hargreaves

Catálogo de Práticas de Acessibilidade: Um Apoio *Online* Voltado à Acessibilidade da Web

Monografia submetida ao curso de graduação em Engenharia de Software da Universidade de Brasília, como requisito parcial para obtenção do Título de Bacharel em Engenharia de Software.

Universidade de Brasília – UnB

Faculdade UnB Gama – FGA

Orientador: Profa. Dra. Milene Serrano

Coorientador: Prof. Dr. Maurício Serrano

Brasília, DF

2021

Felipe de Oliveira Hargreaves
Catálogo de Práticas de Acessibilidade:
Um Apoio *Online* Voltado à Acessibilidade da Web/ Felipe de Oliveira Hargreaves.
– Brasília, DF, 2021-
121 p. : il. (algumas color.) ; 30 cm.

Orientador: Profa. Dra. Milene Serrano

Trabalho de Conclusão de Curso – Universidade de Brasília – UnB
Faculdade UnB Gama – FGA , 2021.

1. Acessibilidade. 2. Qualidade de software. I. Profa. Dra. Milene Serrano. II. Universidade de Brasília. III. Faculdade UnB Gama. IV. Catálogo de Práticas de Acessibilidade:
Um Apoio *Online* Voltado à Acessibilidade da Web

CDU 02:141:005.6

Felipe de Oliveira Hargreaves

Catálogo de Práticas de Acessibilidade: Um Apoio *Online* Voltado à Acessibilidade da Web

Monografia submetida ao curso de graduação em Engenharia de Software da Universidade de Brasília, como requisito parcial para obtenção do Título de Bacharel em Engenharia de Software.

Trabalho aprovado. Brasília, DF, 19 de Maio de 2021:

Profa. Dra. Milene Serrano
Orientadora

Prof. Dr. Maurício Serrano
Co-orientador

Prof. Dr. Renato Coral Sampaio
Convidado

Brasília, DF
2021

Agradecimentos

Agradeço aos meus pais, Wilson e Belarmina, por sempre terem incentivado e instigado a minha curiosidade e vontade de aprender. Obrigado por todos os conselhos, pelas longas e proveitosas conversas e pelo apoio incondicional.

Agradeço aos meus orientadores, Milene Serrano e Maurício Serrano, por tornarem a experiência do TCC tão motivadora e enriquecedora. Agradeço por todas as reuniões, conversas, revisões e pelo altíssimo nível de empenho, dedicação e compreensão. Agradeço também por toda a ajuda e orientação ao longo da graduação.

Agradeço ainda a todos os amigos que fiz durante a universidade, por todos os momentos e experiências compartilhadas durante o curso. Obrigado pela ajuda e pelo companheirismo.

Resumo

A acessibilidade é um critério de qualidade de software que tem crescido em importância nos últimos anos. Em um mundo cada vez mais dependente de tecnologia para inúmeros aspectos do cotidiano, torna-se essencial que o acesso à essa tecnologia seja transparente e igualitário, para usuários das mais diferentes origens e capacidades. A normatização de práticas e técnicas, bem como a formulação de legislação promovendo a obrigatoriedade de construção de interfaces acessíveis para *sites* de empresas e serviços governamentais são exemplos de avanços importantes na área. No entanto, grande parte dos *sites* e aplicativos modernos ainda apresenta problemas de acessibilidade. Um motivo para esse cenário é a baixa consciência sobre questões de acessibilidade por parte dos desenvolvedores de software. Este trabalho buscou contribuir para a criação de interfaces mais acessíveis, ao oferecer um apoio *online* que reúne e resume aspectos relevantes da literatura especializada sobre o tema, com ênfase na acessibilidade da Web. Esse apoio, livre e de código aberto, apresenta recursos informativos sobre práticas e tecnologias relacionadas à acessibilidade, bem como provê exemplos de interfaces acessíveis e não-acessíveis. O apoio foi desenvolvido e aperfeiçoado ao longo de três ciclos de análise, e disponibilizado publicamente ao final do projeto.

Palavras-chave: acessibilidade, qualidade de software, web, WCAG.

Abstract

Accessibility is a software quality criteria of increasing importance in recent years. In a world more and more dependent of technology for innumerous aspects of daily life, it becomes essential for the access to that technology to be transparent and equalitarian. The standardization of practices and techniques, as well as the passing of legislation promoting mandatory accessibility in websites of enterprises and government services are examples of relevant improvements in the area. Nonetheless, a large part of modern websites and applications still present accessibility issues. A reason for this situation is the lack of awareness by software developers about accessibility questions. The present work intends to contribute to the creation of more accessible user interfaces, by providing an online support, where relevant aspects of the specialized literature concerning software accessibility are collected and summarized, with a special focus on Web accessibility. This support, free and open source, presents informative resources about accessibility-related practices and technologies, as well as providing examples of accessible and not accessible interfaces. The support was developed and improved during three cycles of analysis, and made publicly available at the end of the project.

Key-words: accessibility, software quality, web, WCAG.

Lista de ilustrações

Lista de tabelas

Lista de abreviaturas e siglas

WAI	*Web Accessibility Initiative*
W3C	*World Wide Web Consortium*
WCAG	*Web Content Accessibility Guidelines*
ARIA	*Accessible Rich Internet Applications*
eMAG	Modelo de Acessibilidade em Governo Eletrônico

Sumário

1 Introdução

1.1 Contextualização

O mundo moderno é, cada vez mais, dominado pela presença constante de *software* nos mais variados domínios - comunicação, trabalho, entretenimento, serviços públicos, comércio, saúde, entre outros. De acordo com (CGI, 2018), em 2018, cerca de 70% da população brasileira possuía acesso à Internet. Entre esses usuários, 97% utilizava a rede através de um telefone celular. Dentro desse contexto, é importante que o desenvolvimento de aplicações seja realizado de forma a gerar um produto utilizável em sua totalidade por pessoas de diferentes origens e capacidades.

Para isso, a preocupação com critérios de acessibilidade na construção de interfaces tem se tornado de crescente relevância. Exemplos do progresso realizado na área incluem a criação e evolução de diretrizes como a *Web Content Accessibility Guidelines* (WCAG, atualmente na versão 2.1) (W3C, 2018c), tecnologias como a *Accessible Rich Internet Applications Suite* (WAI-ARIA) (W3C, 2017a), e regulamentações governamentais ao redor do mundo, que criam uma base jurídica para a necessidade de desenvolvimento de aplicações acessíveis. No Brasil, existe o decreto presidencial 5.296, de 02/12/2004 (BRASIL, 2004), que regulamenta leis de acessibilidade e obriga que sites governamentais sejam acessíveis, e o eMAG (Modelo de Acessibilidade em Governo Eletrônico) (BRASIL, 2014), documento baseado nas recomendações do WCAG e utilizado como base para a criação de conteúdos digitais do governo.

Embora essas medidas tenham como objetivo principal garantir o uso pleno de *software* por pessoas com algum tipo de necessidade especial, autores como (BARBOSA; SILVA, 2010) destacam que a acessibilidade não visa apenas atender a um grupo específico de usuários, mas sim permitir o uso de um sistema sem que sua interface apresente barreiras, para qualquer tipo de pessoa.

1.2 Questão de pesquisa

Ao longo desse trabalho, pretende-se responder primariamente a seguinte questão de pesquisa:

- Como idealizar e construir um suporte de apoio *online*, informativo e aderente à literatura especializada sobre acessibilidade digital, e que promova o conhecimento acerca desse critério de qualidade, tendo como público alvo desenvolvedores de software?

Desta forma, podem ainda ser destacados os seguintes desafios correlacionados com a questão de pesquisa, que também exigem um levantamento bibliográfico adequado:

- Há literatura especializada contemplando acessibilidade de software?
- Por que questões de acessibilidade não costumam ser tratadas no desenvolvimento de software?
- Como prover um suporte informativo sobre acessibilidade que apoie os desenvolvedores de software?

1.3 Justificativa

Apesar dos avanços mencionados, a falta de acessibilidade ainda é um problema que atinge uma grande quantidade dos programas desenvolvidos atualmente. Em (YAN; RAMACHANDRAN, 2019), os autores mostram que quase 95% dos apps mais baixados para Android apresentam violações de diretrizes de acessibilidade. Em (G3ICT, 2016), tem-se que, apesar de haver legislação promovendo políticas e regulamentações relacionadas à acessibilidade, o nível real de implementação de serviços digitais acessíveis ao redor do mundo ainda é pequeno.

Entre os motivos existentes para esse cenário, a falta de conhecimento e treinamento por parte dos desenvolvedores é um fator crucial, como relatado por (YAN; RAMACHANDRAN, 2019). Antonelli et al. (2018); Freire, Russo e Fortes (2008); Pichiliani e Pizzolato (2019) são exemplos de pesquisas visando avaliar os níveis de conhecimento e interesse de pessoas envolvidas com o desenvolvimento Web, em relação à acessibilidade, no Brasil e em outros países. Embora não seja possível fazer uma comparação direta entre seus resultados, devido a diferenças metodológicas, a falta de pessoas capacitadas na equipe e a falta de consciência sobre o assunto são motivos comumente citados como impedimento para a acessibilidade (PICHILIANI; PIZZOLATO, 2019; ANTONELLI et al., 2018; FREIRE; RUSSO; FORTES, 2008).

A necessidade de proporcionar melhores fontes de conhecimento e capacitação para que desenvolvedores de software construam aplicações mais acessíveis é reconhecida. Silva et al. (2018); Shinohara et al. (2018); Gay, Djafarova e Zefi (2017); El-Glaly (2020); Cohen et al. (2005) discutem propostas e experiências em como introduzir o assunto de acessibilidade, em sua maioria com foco em programas de ensino superior em Engenharia de Software ou Ciência da Computação. O presente trabalho visa contribuir com essas iniciativas, buscando construir um suporte *online* que proporcione conhecimento e apoio aos desenvolvedores de software na criação de produtos de software orientados às boas práticas de acessibilidade digital.

1.4 Objetivos

1.4.1 Objetivo geral

Prover um suporte de apoio *online* aos desenvolvedores de *software*, que promova conceitos básicos sobre o tema; apresentando ferramentas de desenvolvimento e testes, e demonstrando exemplos de interfaces e componentes acessíveis.

1.4.2 Objetivos específicos

- Identificar as diretrizes e recomendações mais reconhecidas e utilizadas para o desenvolvimento de interfaces de software acessíveis;
- Identificar e explorar as principais tecnologias assistivas existentes para interação com software;
- Identificar ferramentas de apoio ao desenvolvimento de aplicações acessíveis;
- Estruturar e priorizar as informações obtidas nos objetivos anteriores;
- Implementar a solução proposta, apresentando de forma objetiva e clara os conhecimentos elicitados, e que deverão ser utilizados para a própria construção da solução;
- Tratar os resultados obtidos, de forma a avaliar se a implementação realizada conseguiu catalogar e apresentar uma quantidade relevante de informações sobre o tema, tendo como base o referencial adquirido nas fases anteriores do projeto;
- Avaliar se a solução foi construída de acordo com as diretrizes de acessibilidade elicitadas.

1.5 Organização do trabalho

O restante desta monografia está organizado em capítulos, apresentados brevemente a seguir. O Capítulo 2 introduz alguns dos conceitos essenciais para a compreensão do tema de acessibilidade de software, além de contextualizar o estado atual da área em relação a tecnologias, normas e métodos de avaliação. O Capítulo 3 descreve as ferramentas e tecnologias de apoio à construção do trabalho, justificando suas escolhas. O Capítulo 4 define os elementos metodológicos que orientaram a elaboração do trabalho, ao longo de todas as suas etapas. O Capítulo 5 apresenta o apoio em si, detalhando o seu conteúdo, os pontos de destaque da sua implementação e as várias etapas de seu desenvolvimento. O Capítulo 6 mostra os procedimentos de análise dos resultados obtidos. Por fim, o Capítulo 7 traz o desfecho do trabalho e comenta perspectivas de evoluções e trabalhos futuros.

2 Referencial teórico

Este capítulo apresentará a fundamentação teórica necessária para o desenvolvimento do projeto, visando facilitar a compreensão sobre tópicos essenciais ao tema. São apresentadas definições de termos importantes relacionados à acessibilidade, assim como algumas das ferramentas, tecnologias e técnicas mais utilizadas dentro desse contexto. Também são expostos e discutidos os principais padrões técnicos e legais referentes à acessibilidade digital, bem como alguns dos métodos mais comumente utilizados para se avaliar a acessibilidade de um *software*.

2.1 Acessibilidade

O conceito de acessibilidade é aplicável a vários contextos, como arquitetura, transportes, comunicação e informação. Essa variedade leva a diferentes compreensões sobre o termo, que embora sutis podem resultar em abordagens diferentes de *design*, como menciona (MELO; CECíLIA; BARANAUSKAS, 2005). Este trabalho trata de acessibilidade sob a ótica da Interação Humano-Computador, com foco especial na acessibilidade da Web.

Preece, Sharp e Rogers (2019) definem acessibilidade como o nível em que um produto interativo é acessível para o maior número possível de pessoas, dando foco para pessoas com deficiências. Preece, Sharp e Rogers (2019) ainda relacionam o termo com os conceitos de inclusividade e de *design* inclusivo, que envolvem o esforço para a criação de produtos e serviços que acomodem o maior número possível de pessoas, independentemente de fatores como deficiência e diferenças de idade, nível de educação ou renda.

Barbosa e Silva (2010) definem acessibilidade como um critério de qualidade relacionado à capacidade de interação de um usuário com um sistema, sem que sua interface apresente obstáculos para seu uso ou acesso de suas informações. Barbosa e Silva (2010) destacam ainda que a acessibilidade visa permitir que *mais pessoas* possam usufruir de um sistema, não que *apenas* um grupo específico seja atendido.

Shneiderman et al. (2018) abordam o tema de acessibilidade através de um conceito mais amplo, o de Usabilidade Universal. Essa ideia defende que o *design* de interfaces de usuário deve levar em conta fatores como:

- Diferenças de espaço físico, considerando fatores de ambiente como ruídos, iluminação, vibrações e movimento;
- Diversidade de habilidades cognitivas e de percepção;

- Diverenças de personalidades entre os usuários;
- Diversidade cultural e linguística;
- Usuários com deficiência, e
- Usuários idosos e crianças.

Em todas as definições, é possível identificar elementos em comum - a ideia de que as interfaces de software devem ser projetadas e implementadas de forma a serem usadas plenamente pelo maior número possível de pessoas, independente de suas diferentes origens, habilidades ou capacidades.

Dentro desse contexto, é importante entender os diferentes tipos de impedimentos que devem ser levados em conta para a acessibilidade de um sistema. Preece, Sharp e Rogers (2019) os categorizam de duas formas: quanto ao tipo de limitação, e quanto a sua duração. Em relação aos tipos de limitações, distinguem:

- Limitações sensoriais, como perda de visão ou audição;
- Limitações físicas, como perda de função motora em decorrência de um acidente, e
- Limitações cognitivas, como perda de memória ou dificuldades de aprendizado.

Preece, Sharp e Rogers (2019) destacam ainda que cada tipo inclui uma variedade ampla e complexa de pessoas e necessidades, cada uma exigindo abordagens específicas de *design*. Por fim, separam as limitações em função de sua duração, podendo ser permanentes, temporárias ou situacionais.

2.1.1 Acessibilidade e Usabilidade

Outro ponto a se destacar é a relação entre o critério de acessibilidade e o de usabilidade. Segundo (NIELSEN, 1993), a usabilidade é um critério composto de um conjunto de fatores que determinam a qualidade de interação com um sistema. Esses fatores são:

- *Aprendizagem* - a facilidade de se aprender a utilizar o sistema;
- *Memorização* - a facilidade de se lembrar de como interagir com o sistema;
- *Eficiência de uso* - o suporte que a interface oferece para que os usuários consigam conduzir suas atividades com um alto nível de produtividade;
- *Erros/Segurança de uso* - a interface deve apresentar uma taxa baixa de erros, e proporcionar mecanismos para que os usuários se recuperem de falhas com facilidade, e

- *Satisfação* - a avaliação subjetiva do quão agradável é utilizar o sistema.

Desde o passado, Nielsen (1993) já destacava que a usabilidade deve ser avaliada com base em um grupo específico de usuários, para um conjunto específico de tarefas. Dessa forma, as características de usabilidade de um mesmo sistema podem variar, de acordo com o usuário a utilizá-lo. Adicionalmente, um sistema pode ser considerado de boa usabilidade para um grupo considerável de pessoas, mas ainda assim não ser considerado acessível. De acordo com (W3C, 2016), acessibilidade e usabilidade encontram-se quando se incluem pessoas com uma variedade de deficiências no grupo de usuários, e quando o contexto de uso considera elementos como tecnologias assistivas.

Por outro lado, um sistema pode ser projetado com acessibilidade em mente, e ainda assim não atender aos fatores mencionados para definir sua usabilidade. Esse ponto é reforçado por (NIELSEN, 2005), ao considerar que a acessibilidade não pode ser tratada em um vácuo, sem considerar as ações que os usuários desejam realizar através do sistema.

Barbosa e Silva (2010) resumem essas considerações, ao considerar que o *design* de interfaces deve permitir que os objetivos de interação com um sistema sejam alcançados por um amplo espectro de usuários (contemplando desta forma a acessibilidade), e ao mesmo tempo garantir que esses objetivos sejam alcançados com eficácia, eficiência e satisfação para os usuários (atendendo desta forma a usabilidade).

2.1.2 Acessibilidade da Web

O desenvolvimento da Web é um dos fatores a impulsionar a consciência sobre critérios de acessibilidade em software na atualidade. Shneiderman et al. (2018) destacam que grande parte das leis e regulamentos sobre acessibilidade nos meios digitais é baseada nos padrões definidos pela *Web Accessibility Initiative* (WAI), até mesmo para sistemas que não fazem parte da Web.

Para (W3C, 2019), a acessibilidade da Web diz respeito a *websites*, ferramentas e tecnologias projetados de forma que pessoas com deficiências possam perceber, compreender, navegar, interagir e contribuir com a Web. Além disso, W3C (2019) enfatiza que a acessibilidade também beneficia outros tipos de usuários, como pessoas com limitações temporárias ou situacionais, idosos e pessoas com uma conexão de Internet lenta ou limitada.

Autores como (HENRY; ABOU-ZAHRA; BREWER, 2014) alertam, no entanto, que apesar de existir uma sobreposição significativa entre projetar para a acessibilidade e projetar para limitações situacionais, deve ser mantida a separação de acessibilidade como uma disciplina dedicada para as necessidades específicas de usuários com deficiências. Dessa forma, garante-se que essas necessidades não sejam perdidas em meio a uma categoria mais ampla de problemas. Henry, Abou-Zahra e Brewer (2014) argumentam

ainda que as considerações de acessibilidade devem ser integradas à pesquisa, ao desenvolvimento e ao *design* de interfaces, promovendo a colaboração com áreas relacionadas que tratam de assuntos similares, mas de forma mais ampla.

A acessibilidade na Web é dependente de vários componentes - não apenas da implementação dos *websites* em si, mas também da forma com que as tecnologias relacionadas à Web são desenvolvidas, como o suporte à acessibilidade nos *browsers* e nas ferramentas de autoria (ex. interfaces de desenvolvimento, editores de texto) (W3C, 2019). A WAI é responsável por criar diretrizes, especificações técnicas e demais recursos relacionados à acessibilidade para esses componentes.

2.2 Tecnologias assistivas

O grau de acessibilidade em um sistema de software é dependente de dois fatores - o *design* inclusivo das interfaces e o *design* de tecnologias assistivas (PREECE; SHARP; ROGERS, 2019). Certos critérios de acessibilidade podem ser alcançados apenas com uma abordagem inclusiva de *design*; por exemplo, uma escolha de cores e contraste em uma interface que não prejudique usuários com daltonismo. Para outras situações, no entanto, se torna necessário o apoio de ferramentas externas à interface.

A definição formal e legal para Tecnologia Assistiva no Brasil, elaborada pelo Comitê de Ajudas Técnicas (CAT), é a seguinte:

> Tecnologia Assistiva é uma área do conhecimento, de característica interdisciplinar, que engloba produtos, recursos, metodologias, estratégias, práticas e serviços que objetivam promover a funcionalidade, relacionada à atividade e participação, de pessoas com deficiência, incapacidades ou mobilidade reduzida, visando sua autonomia, independência, qualidade de vida e inclusão social. (BRASIL, 2009)

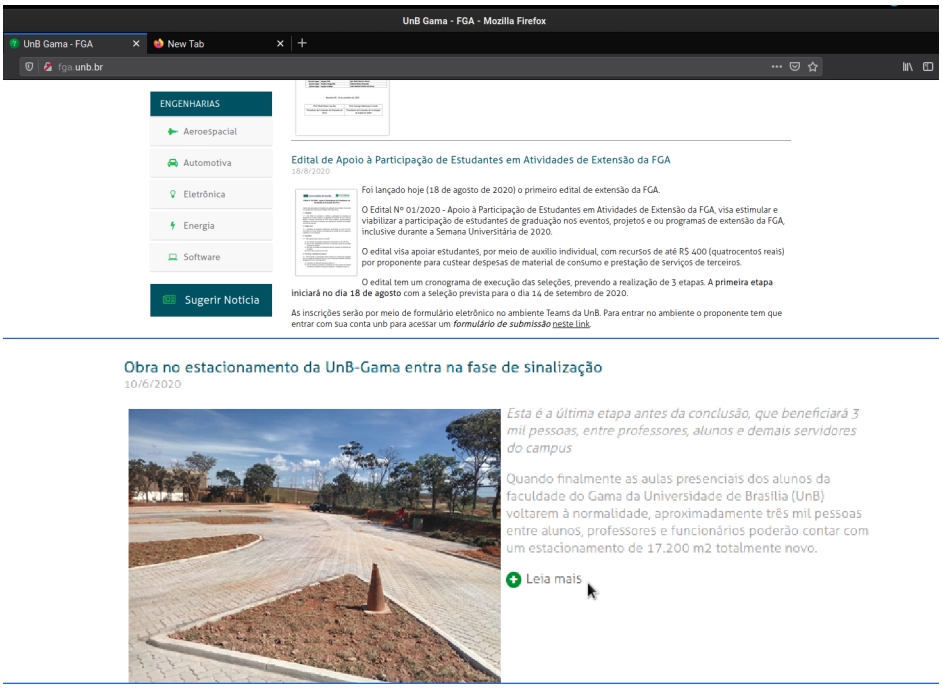
Para este trabalho, são consideradas as tecnologias assistivas relacionadas ao contexto da Interação Humano-Computador, isto é, tecnologias que auxiliam usuários com deficiência no uso de interfaces de software. De acordo com (W3C, 2017b), para a acessibilidade da Web, tecnologias assistivas são *software* ou *hardware* utilizados por pessoas com deficiências para melhorar a interação com a Web. O conjunto de técnicas utilizadas por esses usuários para melhorar a interação é definido como *estratégias adaptivas*.

A seguir serão apresentadas algumas das tecnologias assistivas mais comumente utilizadas para a interação com *software*.

Ampliadores de tela - Software utilizado em sua maioria por pessoas com baixa visão, para ampliar o conteúdo de uma interface, parcialmente ou em sua totalidade. É instalado por padrão em sistemas como Windows e MacOS e ambientes de desktop Linux

como GNOME (Figura 1) e KDE. Também é uma ferramenta do sistema em dispositivos Android (Figura 2) e iOS.

Figura 1 – Exemplo de uso do ampliador de tela do GNOME, com a interface ampliada na parte inferior da tela.



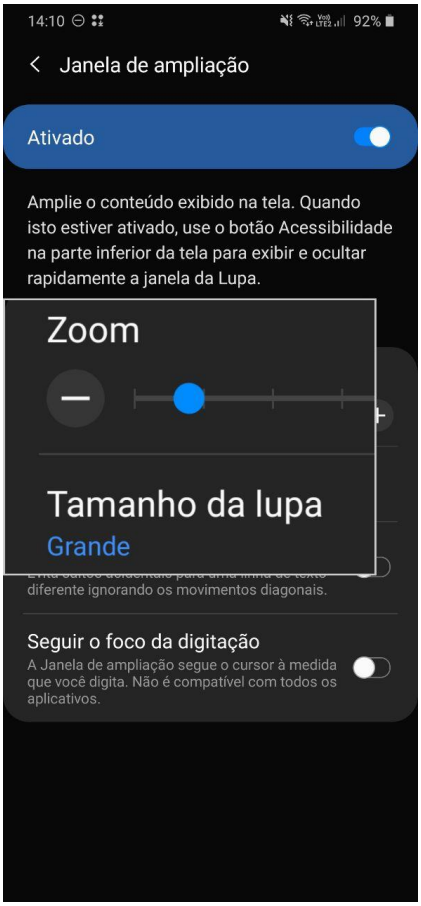
Fonte: Autor.

Leitores de tela - Software que processa o conteúdo de uma interface e o converte, em voz sintetizada ou Braille (W3C, 2017b). Usado, principalmente, por pessoas com deficiências visuais, mas também pode auxiliar pessoas com dificuldades de leitura (MELO; CECíLIA; BARANAUSKAS, 2005). É importante ressaltar que o uso do leitor de tela não torna uma aplicação imediatamente acessível para esses usuários; é necessário que o software utilizado tenha sido implementado com o suporte a leitores de tela em mente. Por exemplo, elementos não textuais, como imagens e ícones, devem possuir uma descrição alternativa textual. Exemplos de leitores de tela incluem: Windows Narrator [1], JAWS [2] e NVDA [3] para Windows; Orca para Linux [4]; VoiceOver para dispositivos da Apple [5]; e TalkBack para dispositivos Android [6].

Display Braille - Um dispositivo mecânico que transforma o conteúdo textual de

[1] https://support.microsoft.com/en-us/windows/complete-guide-to-narrator-e4397a0d-ef4f-b386-d8ae-c172f109bdb1. Acessado pela última vez em 03/05/2021.
[2] https://www.freedomscientific.com/products/software/jaws/. Acessado pela última vez em 03/05/2021.
[3] https://www.nvaccess.org/about-nvda/. Acessado pela última vez em 03/05/2021.
[4] https://help.gnome.org/users/orca/stable/introduction.html.en. Acessado pela última vez em 03/05/2021.
[5] https://www.apple.com/accessibility/mac/vision/. Acessado pela última vez em 03/05/2021.
[6] https://support.google.com/accessibility/android/answer/6283677?hl=en. Acessado pela última vez em 03/05/2021.

Figura 2 – Exemplo de uso do ampliador de tela do Android.



Fonte: Autor.

uma interface através de uma linha de células braille, que são atualizadas dinamicamente ao abaixar e levantar os pinos do dispositivo.

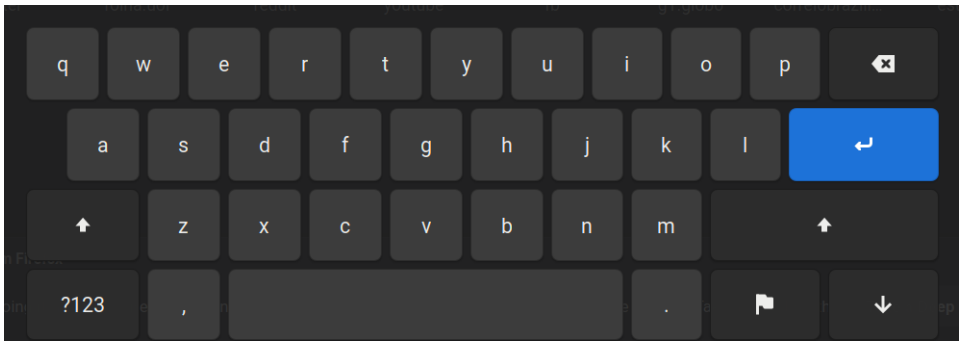
Teclados alternativos - *Software* ou *hardware* utilizados, principalmente, por pessoas com deficiências físicas ou cognitivas (W3C, 2017b), que funcionam como alternativa ao uso de teclados convencionais (MELO; CECíLIA; BARANAUSKAS, 2005). Alguns desses dispositivos são:

- Teclados com teclas maiores;
- Teclas com cores em alto contraste;
- *Keyguards* (Figura 3), placas rígidas colocadas em cima de teclados, que ajudam a reduzir o pressionamento acidental de teclas e permitem o repouso das mãos em cima do teclado;
- Teclados virtuais, como o da Figura 4, e
- Teclados em Braille, como os existentes em Android (ver Figura 5) e iOS, integrados aos seus respectivos leitores de tela. O formato tradicional de teclado é substituído

Figura 3 – Teclado com *Keyguard*.

Fonte: (ABILITYNET, 2019).

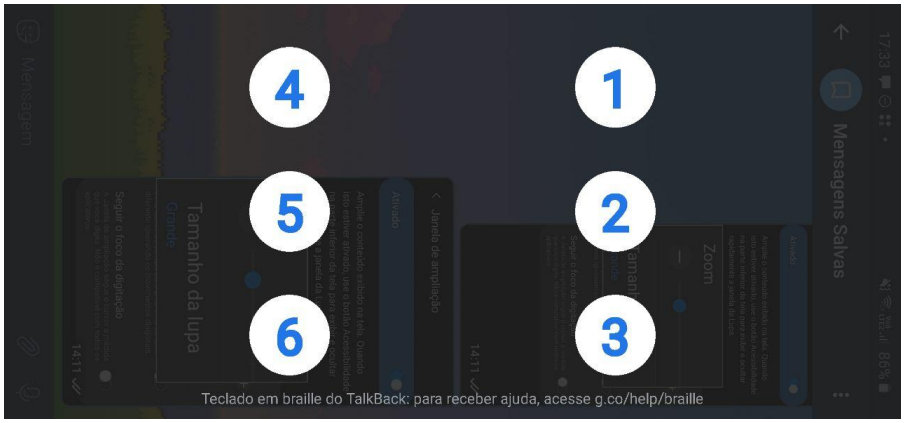
Figura 4 – Exemplo de teclado virtual no Linux, com ambiente de Desktop GNOME.



Fonte: Autor.

por 6 botões, correspondentes aos 6 pontos em Braille. O usuário recebe *feedback* sonoro de cada letra digitada através do leitor de tela, e pode usar uma sequência de gestos na tela para realizar ações como pular de linha ou apagar caracteres.

Figura 5 – Teclado em Braille utilizado em conjunto com o Android TalkBack.



Fonte: Autor.

Dispositivos apontadores alternativos - Dispositivos que funcionam como uma alternativa ao *design* comum de *mouses*, proporcionando o acesso a usuários com limitações motoras. Alguns exemplos são: dispositivos que controlam o movimento do cursor através da movimentação da cabeça ou dos olhos; acionamento de elementos da in-

terface através de ações como pressão em botões, tração, piscar de olhos, sopro e contração muscular (BERSCH; PELOSI, 2007).

Software de reconhecimento de fala - Software que permite a interação com o computador através da voz, para ditar entradas de texto ou executar ações do sistema (W3C, 2017b), possibilitando o acesso por usuários com dificuldades de movimento nos membros superiores. Também beneficia usuários de forma geral em vários contextos situacionais, como o controle de um GPS enquanto se dirige, por exemplo. Esse mesmo tipo de tecnologia pode ser utilizado para a geração automática de legendas em vídeos, beneficiando usuários com problemas de audição.

2.3 Diretrizes e legislação sobre acessibilidade

Muitas das diretrizes reconhecidas internacionalmente sobre acessibilidade em software são criadas pela W3C/WAI, ou baseadas em suas recomendações (SHNEIDERMAN et al., 2018). Embora seja direcionada especificamente para a acessibilidade na Web, grande parte de suas técnicas e recursos podem ser aplicados e adaptados à outros ambientes de software, como aplicações *desktop* e *mobile* (W3C, 2015b; W3C, 2013).

Os padrões da WAI fazem parte de um grupo mais amplo de padrões da Web como um todo, conhecidos como *W3C Recommendations*. As principais recomendações com diretrizes relacionadas à acessibilidade são:

- *Web Content Accessibility Guidelines* - define as diretrizes para tornar o contéudo da Web acessível. Está atualmente na versão 2.1, com a versão 2.2 planejada para 2021. A versão 2.0 é considerada um padrão ISO (ISO, 2012).
- *Authoring Tools Accessibility Guidelines* - define as diretrizes de acessibilidade para os criadores de ferramentas de autoria de conteúdo da Web. Exemplos de ferramentas de autoria incluem editores de HTML, software de criação ou conversão de páginas Web e sites que permitam a criação de conteúdo por seus usuários (W3C, 2015a).
- *User Agent Accessibility Guidelines* - define as diretrizes de acessibilidade para os criadores de *user agents*. Se entende por *user agent*, o software utilizado para renderizar o conteúdo da Web, como *browsers* e suas extensões (W3C, 2002).

Além dessas diretrizes, outra recomendação relevante é o *Accessibile Rich Internet Applications* (ARIA), uma especificação técnica para melhorar a acessibilidade de aplicações Web com conteúdo dinâmico e interfaces avançadas com uso de Ajax, HTML e JavaScript (W3C, 2017a).

Entre os padrões mencionados, o WCAG é o de propósito mais geral para desenvolvedores de websites e aplicações. Em sua versão 2.1, o WCAG é composto de 13

diretrizes, organizadas dentro de quatro princípios, que determinam que o conteúdo na Web deve ser:

- Percebível - As informações e os componentes de uma interface devem ser apresentados de forma que seus usuários consigam percebê-los;
- Operável - Os usuários precisam ser capazes de operar a interface, isto é, a interface não pode exigir interação que não possa ser realizada;
- Compreensível - As informações e as operações da interface de usuário devem ser compreendidas por seus usuários, e
- Robusto - O conteúdo precisa ser interpretado de forma confiável através de vários *user agents*, incluindo tecnologias assistivas, e deve se manter acessível ao longo de evoluções nessas tecnologias e nos *user agents*.

Para cada diretriz contida nesses princípios, existem critérios de sucesso, testáveis, que abordam problemas específicos de acesso para usuários com deficiências. Os critérios são classificados em três níveis, A, AA e AAA, e correspondem aos níveis de *conformidade* com a norma WCAG. Isto é, se uma página da Web cumpre todos os critérios de sucesso do nível A, pode se dizer que essa página tem nível A de conformidade com o WCAG. O teste dos critérios deve ser realizado com um misto de técnicas automatizadas e manuais.

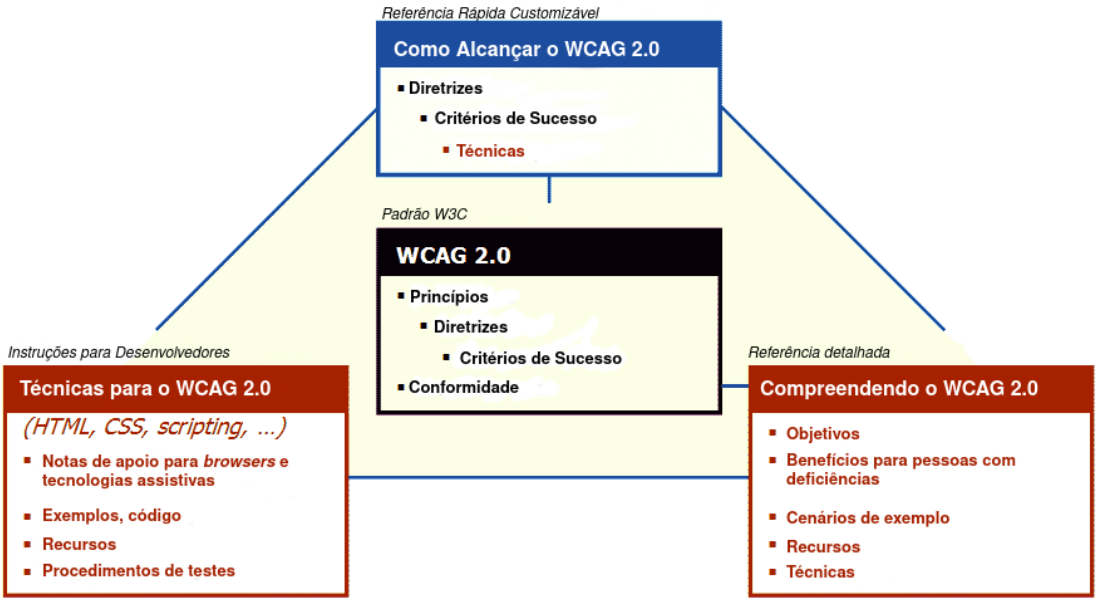
Em complemento ao padrão técnico, a WAI também provê documentos complementares, com recursos de referência às diretrizes e detalhando técnicas e exemplos que auxiliem nos critérios de sucesso. Uma visão geral desses documentos e suas relações pode ser vista na Figura 6.

A acessibilidade *online* é prevista por lei no Brasil, através do Decreto nº5.296 de 2004 (BRASIL, 2004) e da Lei 13.146 (BRASIL, 2015), que tornam obrigatória a acessibilidade em *sites* governamentais e em *sites* de empresas com sede ou representação comercial no Brasil, respectivamente. Para os *sites* e portais do governo, a implementação deve ser guiada pelo Modelo de Acessibilidade em Governo Eletrônico (eMAG), uma versão especializada do WCAG e adaptada para a realidade brasileira (BRASIL, 2014). Define elementos padronizados a serem utilizados por todos os *sites* do Governo Federal, além de recomendações para a escrita de HTML, comportamento das páginas, apresentação do conteúdo e dos elementos, componentes de multimídia e formulários.

2.4 Métodos de avaliação de acessibilidade

O nível de acessibilidade de um sistema pode ser avaliado através de diversos métodos, com diferentes procedimentos, propósitos e resultados. A seguir serão apresentados alguns dos métodos mais comumente utilizados.

Figura 6 – Relação de Documentos do WCAG 2.0.



Fonte: (W3C, 2018b), traduzido pelo autor.

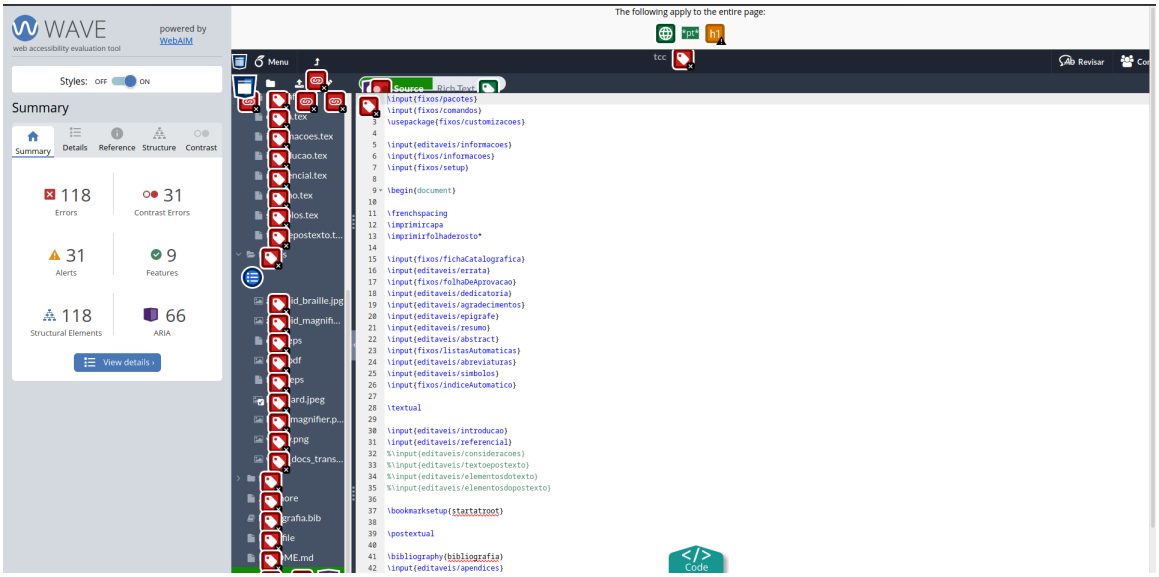
Métodos de inspeção - consistem em um avaliador inspecionando o sistema em relação à sua acessibilidade. A forma mais comum de inspeção é através de Avaliações de Conformidade, que visam aferir o grau de conformidade de um sistema a um conjunto definido de diretrizes (BRAJNIK; YESILADA; HARPER, 2010). Essas diretrizes podem ser padrões internacionais como o WCAG, diretrizes nacionais ou regionais, ou um conjunto de regras definidas internamente por uma organização (BRAJNIK, 2008).

Testes automatizados - envolve o uso de ferramentas automatizadas de avaliação de acessibilidade, que averiguam a conformidade de uma interface em relação ao conjunto de diretrizes programadas na ferramenta (BRAJNIK; YESILADA; HARPER, 2010). Um exemplo é o WAVE (*Web Accessibility Evaluation Tool*), uma suíte de ferramentas que usa as diretrizes do WCAG para encontrar erros e informar pontos passíveis de avaliação manual em uma página na Web (Figura 7).

Screening techniques - um conjunto de atividades que geralmente envolvem a interação com um sistema enquanto o usuário (neste caso, geralmente da própria equipe de desenvolvimento) tem uma ou mais de suas habilidades físicas ou sensoriais reduzidas artificialmente, com o objetivo de identificar potenciais problemas de *design* da interface, fazendo uso de tecnologias assistivas e estratégias adaptivas (HENRY, 2007).

Testes de usabilidade - testes realizados com usuários reais, onde os avaliadores observam o comportamento dos usuários ao realizar tarefas com o software. Dentro do contexto da acessibilidade, devem ser planejados e executados com algumas adaptações em relação a testes de usabilidade mais gerais, de forma a considerar a realidade específica dos usuários participantes (HENRY, 2007).

Figura 7 – Execução do WAVE sobre uma página Web.



Fonte: Autor.

Cada um desses métodos possui vantagens e limitações em comparação aos outros. Conformidade com diretrizes é um requisito legal para alguns sistemas, e demonstra um nível determinado de esforço em relação a questões técnicas de acessibilidade. Com um conjunto de diretrizes adequado, é possível identificar um número razoável de problemas em contextos variados (BRAJNIK, 2008). No entanto, pode não considerar problemas na usabilidade do sistema por usuários com deficiências. Esses problemas são mais simples de serem identificados através de testes com usuários reais (HENRY, 2007; NIELSEN, 2005). Além disso, inspeções de conformidade variam consideravelmente de acordo com o grau de expertise dos avaliadores. Mesmo especialistas com conhecimentos e experiências similares podem chegar a avaliações consideravelmente diferentes sobre um mesmo sistema, e falhar em reconhecer não-conformidades (BRAJNIK; YESILADA; HARPER, 2010). Padrões como o WCAG são reconhecidamente complexos e podem apresentar barreiras para avaliadores com pouca experiência (ALONSO et al., 2010; BRAJNIK; YESILADA; HARPER, 2010).

O uso de ferramentas automatizadas pode auxiliar a encontrar problemas, mas não é suficiente para analisar elementos que exigem uma interpretação subjetiva (HENRY, 2007). Por exemplo, o relatório de uma análise automatizada pode identificar componentes como imagens que não possuem uma descrição textual alternativa para usuários que utilizem leitores de tela; entretanto, não consegue avaliar se a descrição é de fato adequada, em elementos que possuem texto alternativo.

Testes de usabilidade são uma ferramenta importante para aumentar a compreensão sobre problemas comuns que os usuários encontram ao utilizar o sistema, mas não permitem encontrar todos os problemas de acessibilidade do sistema, ou garantir

sua conformidade com padrões (HENRY, 2007). Por exigir cuidados com planejamento e preparação, além da disponibilidade de usuários reais, é recomendado aplicar outras avaliações anteriormente, de forma a eliminar erros e impedimentos que podem interferir ou até mesmo inviabilizar o andamento dos testes.

O uso de *Screening Techniques* pode auxiliar a identificar problemas óbvios com o *design* do sistema de forma rápida e simples, mas não deve ser considerado como uma simulação confiável do uso por usuários com deficiências, devido à diferença de experiência entre os membros da equipe de desenvolvimento ao executar as atividades e os usuários reais (HENRY, 2007).

Uma avaliação abrangente da acessibilidade de um sistema deve considerar essas limitações e combinar essas práticas ao longo de todo o processo de desenvolvimento.

2.5 Considerações finais do capítulo

Este capítulo abordou alguns dos principais elementos necessários para se compreender o campo da acessibilidade, dentro do contexto da Interação Humano-Computador, com foco especial para a acessibilidade da Web. Foram apresentadas e comentadas definições para o próprio conceito de acessibilidade, assim como sua relação com outros conceitos de *design*, como Usabilidade e *design* inclusivo. Em seguida, foram abordadas algumas das tecnologias mais utilizadas por usuários com deficiências para viabilizar a interação com computadores. Também foram levantadas as principais normas técnicas e legais relacionadas à acessibilidade de software, no Brasil e internacionalmente. Por fim, foram apresentados métodos tradicionalmente utilizados para se avaliar a acessibilidade de um sistema de *software*, com seus respectivos benefícios e limitações.

3 Suporte tecnológico

Neste capítulo, serão apresentadas as principais ferramentas e tecnologias utilizadas para a construção da solução proposta por este trabalho, de forma a descrever e justificar as escolhas realizadas. Serão abordadas, inicialmente, as tecnologias utilizadas para a implementação da solução em si, e em seguida serão apresentadas ferramentas de apoio ao trabalho como um todo. O Quadro 1, disponibilizado ao final do capítulo, resume essas escolhas, detalhando ainda as versões utilizadas de cada ferramenta.

3.1 Tecnologias Web

Tratando-se de um apoio *online*, a construção da solução fez uso principalmente de tecnologias Web, como HTML, CSS e JavaScript. Dois fatores importantes para a escolha das ferramentas utilizadas, dentro do vasto ecossistema de *frameworks* e bibliotecas para a Web, foram:

1. O foco principal do *site* é no conteúdo informativo apresentado ao usuário, e
2. Esse conteúdo é, em sua maior parte, estático.

Esses fatores levantam algumas considerações. A natureza estática das páginas remove a necessidade de se programar código do lado do servidor - basta publicar o conjunto de páginas HTML referente ao *site*. Pelo mesmo motivo, não é necessário fazer uso de um banco de dados. É interessante, no entanto, manter o conteúdo de cada página separado das definições de como esse conteúdo deve ser apresentado. Dessa forma, pode se definir um *layout* básico para um conjunto de páginas uma única vez, e apenas "popular"esse *layout* com conteúdo, quantas vezes for necessário. Para atender a esses requisitos, optou-se por fazer uso de um *Static Site Generator*, mais especificamente o Eleventy (11ty), apresentado a seguir.

3.1.1 Eleventy

Eleventy [1] (também conhecido como 11ty) é um *static site generator* construído em JavaScript, que preza por simplicidade e flexibilidade na construção de páginas Web. Permite que o conteúdo das páginas seja escrito em uma linguagem simples de marcação, como Markdown. Também permite que o *layout* dessas páginas seja definido em arquivos de *template*, reutilizáveis. Ao se gerar uma *build* do *website*, o conteúdo é adicionado aos

[1] https://www.11ty.dev/docs/. Acessado pela última vez em 03/05/2021.

templates, que são transformados em páginas HTML, prontas para a publicação em um servidor.

Desta forma, o Eleventy atende às necessidades mencionadas. Outros fatores para sua escolha são a facilidade de integração contínua com plataformas de publicação como Netlify e Github Pages, e a preocupação explícita dos desenvolvedores com questões de Acessibilidade [2]. O 11ty foi utilizado para a construção do A11Y Project [3], um *site* com recursos relacionados à Acessibilidade Digital e uma *checklist* de conformidade com o WCAG.

3.1.2 Sass

Sass [4] (sigla para *Syntactically Awesome Style Sheets*) é um pré-processador para CSS, que amplia suas funcionalidades através de recursos como variáveis e aninhamento de seletores. Foi utilizado nesse projeto para facilitar a manutenção e melhorar a modularidade das folhas de estilo.

3.1.3 Netlify

O Netlify [5] é uma plataforma de hospedagem de sites. Utilizando de um fluxo de trabalho vinculado ao Git, permite a publicação de forma rápida e simples através do código armazenado em um repositório. Foi utilizado para hospedar o apoio *online*.

3.1.4 Figma

O Figma [6] é uma aplicação Web para *design* e prototipagem de interfaces. Possui *plugins* que podem auxiliar com questões de acessibilidade, como escolha adequada de cores para usuários daltônicos e uso correto de contraste entre elementos da interface. Foi utilizado para o esboço do *layout* geral do *website*.

3.2 Tecnologias de apoio

3.2.1 Sistemas Operacionais

3.2.1.1 Linux

O Linux foi o Sistema Operacional principal para o desenvolvimento do trabalho. Nele, foi realizada a maior parte da prototipação e codificação da solução, bem como parte

[2] https://www.11ty.dev/docs/accessibility/. Acessado pela última vez em 03/05/2021.
[3] https://www.a11yproject.com/. Acessado pela última vez em 03/05/2021.
[4] https://sass-lang.com/. Acessado pela última vez em 03/05/2021.
[5] https://www.netlify.com/. Acessado pela última vez em 03/05/2021.
[6] https://www.figma.com/. Acessado pela última vez em 03/05/2021.

das atividades de avaliação.

3.2.1.2 Windows 10/Android/iOS

Adicionalmente, foram utilizados o Windows 10, o Android e o iOS, de forma complementar. O objetivo inicial foi explorar e se familiarizar com ferramentas de acessibilidade exclusivas de cada SO. Durante o desenvolvimento do apoio, também foram utilizados para avaliar o funcionamento do *site* nesses ambientes distintos, de forma a prezar pela acessibilidade do produto final em mais de um sistema.

3.2.2 Leitores de tela

Como forma de se familiarizar com o uso de tecnologias assistivas, mencionadas na Seção 2.2, fez-se uso de leitores de tela em várias plataformas, para navegação na Web e utilização de programas Desktop e mobile. No *Windows*, utilizou-se o *Windows Narrator*, instalado por padrão no sistema, e o *NVDA*, um leitor de tela gratuito e *open source*. No Linux, utilizou-se o *Orca*, que é o leitor de tela padrão na maior parte das distribuições Linux. Por fim, foram utilizados o *Android Talkback* e o *Apple VoiceOver*, em *smartphones* Android e iOS, respectivamente.

3.2.3 WAVE

Mencionado na Seção 2.4, o WAVE é uma ferramenta automatizada para avaliação de acessibilidade em páginas da Web. Foi utilizado como apoio à construção da solução, para averiguar a conformidade da interface com o WCAG ao longo do desenvolvimento.

Quadro 1 – Relação de tecnologias utilizadas.

Ferramenta	**Categoria**	**Versão**
11ty	*Static Website Generator*	v0.11.0
Sass	Pré-processador CSS	v1.27.0
GithubPages	Hospedagem	-
Figma	Design e prototipagem	-
Arch Linux	Sistema Operacional	*rolling release* [7]
Windows	Sistema Operacional	10
Android	Sistema Operacional	10
iOS	Sistema Operacional	10.3.3
Windows Narrator	Leitor de Tela	-
NVDA	Leitor de Tela	2020.3
Orca	Leitor de Tela	3.38.3
Android TalkBack	Leitor de Tela	8.2.0
Apple VoiceOver	Leitor de Tela	-
WAVE	Avaliação de Acessibilidade	3.1.2

3.3 Considerações finais do capítulo

Esse capítulo apresentou algumas das principais questões a serem consideradas para a escolha das tecnologias relacionadas ao projeto. Essas escolhas contemplam tanto tecnologias utilizadas para a implementação concreta da solução, quanto tecnologias e ferramentas utilizadas para se alcançar melhor familiarização com o tema e auxiliar na avaliação do sistema implementado.

Sempre que possível, buscou-se o uso de ferramentas que levem em conta questões de acessibilidade. Desta forma, foi possível ter auxílio tecnológico adequado mesmo antes da codificação do apoio, desde a prototipação da interface. Contemplar esses fatores durante todo o processo de desenvolvimento contribui para a qualidade do produto final no que diz respeito à acessibilidade.

Complementarmente, optou-se por fazer uso de uma variedade de sistemas operacionais, para fins de experimentação. O objetivo foi obter melhor compreensão sobre o uso e as diferenças entre recursos e tecnologias assistivas específicas de cada plataforma. Assim, pôde-se obter conhecimento útil para a construção do *site*, considerando limitações e funcionalidades de ambientes diversos.

[7] O *Arch Linux* não segue padrão convencional de versões, realizando atualização constante de seus pacotes.

4 Metodologia

Este capítulo pretende abordar as escolhas metodológicas realizadas para a execução deste trabalho, ao longo de todas as suas etapas. Primeiramente, a pesquisa será classificada em relação a vários aspectos, sendo: abordagem, natureza, objetivos e procedimentos. Esses formatos de classificação serão explicados e suas escolhas justificadas. Em seguida, será apresentado o fluxo de atividades realizadas para a execução do projeto. Esse fluxo ilustrará uma visão geral dos processos necessários para a concepção e implementação do trabalho. Alguns subprocessos importantes, como os procedimentos de pesquisa bibliográfica, serão tratados em maiores detalhes. Mais ao final, tem-se a exposição do cronograma de realização das atividades e, por fim, as considerações finais do capítulo.

4.1 Classificação da pesquisa

Os elementos de pesquisa deste trabalho serão classificados através das categorias descritas por (GERHARDT; SILVEIRA, 2009): quanto à abordagem da pesquisa, quanto à natureza, quanto aos objetivos e quanto aos procedimentos. Um resumo das escolhas realizadas pode ser encontrado no Quadro 2.

4.1.1 Abordagem da pesquisa

No que diz respeito à abordagem, o presente trabalho pode ser caracterizado como uma **pesquisa qualitativa**. Nessa abordagem, a pesquisa preocupa-se com aspectos da realidade não quantificáveis, trabalhando com relações, processos e fenômenos que não são reduzíveis à operacionalização de variáveis.(GERHARDT; SILVEIRA, 2009). Procura-se compreender o contexto em que o objeto de pesquisa está inserido, suas origens e relações (OLIVEIRA, 2011).

Ao se falar de acessibilidade, a abordagem qualitativa é de extrema importância, por se tratar de um critério de qualidade, subjetivo. Embora a existência de padrões e normas técnicass gere a noção de que a Acessibilidade seja quantificável em termos de

Quadro 2 – Classificação da pesquisa

Categoria	**Classificação**
Abordagem da Pesquisa	Pesquisa Qualitativa
Natureza da Pesquisa	Pesquisa Aplicada
Objetivos da Pesquisa	Pesquisa Exploratória
Procedimentos	Pesquisa Bibliográfica, Pesquisa-Ação

conformidade com um conjunto de regras, não é possível reduzir a compreensão e avaliação do tema a apenas esses fatores (KELLY et al., 2007).

4.1.2 Natureza da pesquisa

Em relação à natureza, o trabalho pode ser definido como **pesquisa aplicada**. Essa modalidade de pesquisa direciona os conhecimentos gerados para uma aplicação prática, visando solucionar um problema definido (GERHARDT; SILVEIRA, 2009). Essa definição vai de encontro ao objetivo principal do trabalho, onde os conhecimentos adquiridos servem de insumo para a produção do apoio *online*.

4.1.3 Objetivos da pesquisa

Quanto aos objetivos da pesquisa, o trabalho é caracterizado como uma **pesquisa exploratória**. Segundo Gil (2008), esse tipo de pesquisa tem por objetivo aumentar a familiaridade do pesquisador com o tema, facilitando a construção de hipóteses e formulação de perguntas mais precisas. É flexível e comumente associada à abordagem qualitativa (OLIVEIRA, 2011).

Seu uso neste trabalho justifica-se, novamente, pelo caráter subjetivo do tema, sendo importante obter uma ampla quantidade de informações, sob diferentes pontos de vista e contextos.

4.1.4 Procedimentos

São definidos dois procedimentos principais de pesquisa, para fases distintas do trabalho: a **pesquisa bibliográfica** e a **pesquisa-ação**. A pesquisa bibliográfica pode ser vista como o ponto de partida para qualquer trabalho científico, permitindo a obtenção de conhecimento sobre o assunto, fazendo uso de referências já analisadas e publicadas (OLIVEIRA, 2011). Foi utilizada principalmente, mas não exclusivamente, durante as fases iniciais do trabalho, com o objetivo de construir um forte embasamento teórico sobre o tema. Os resultados dessa etapa de pesquisa servem como base para todo o restante do desenvolvimento do trabalho.

Por sua vez, a pesquisa-ação é um formato de pesquisa que pressupõe uma ação prática para a resolução do problema investigado, com o pesquisador adquirindo um papel ativo, visando intervir e transformar a realidade observada (PRODANOV; FREITAS, 2013; GERHARDT; SILVEIRA, 2009). Também é caracterizada por sua natureza cíclica, com a avaliação constante dos resultados sendo aplicada para aprimorar atividades anteriores do processo (ENGEL, 2000). Dessa forma, foi utilizada para a análise dos resultados da pesquisa, fornecendo mecanismos para corrigir e aperfeiçoar a solução proposta, ainda durante a implementação do trabalho.

4.1.5 Metodologia de desenvolvimento

Para o desenvolvimento do apoio *online*, vários elementos de metodologias ágeis de desenvolvimento de software foram utilizados, mais especificamente o ***Kanban*** e o ***Scrum***. Essas metodologias foram utilizadas com adaptações, para melhor atenderem à realidade de um projeto desenvolvido individualmente.

Do *Scrum*, fez-se uso principalmente do *Product Backlog*, uma lista priorizada dos requisitos do *software*, sempre sujeita à alterações de acordo com a evolução do projeto (SCHWABER; BEEDLE, 2002). No entanto, optou-se por uma abordagem mais leve em processos em relação ao *Scrum* tradicional, sem fazer uso dos papéis e rituais como reuniões diárias. Em complemento ao *backlog*, foram utilizados quadros *Kanban*, por permitir uma visualização constante do estado e da evolução do desenvolvimento, facilitando a compreensão, gerenciamento e fluidez do fluxo de trabalho (ANDERSON, 2010).

4.2 Fluxos de atividades

O modelo de processos referente ao fluxo completo de atividades da presente monografia foi dividido em duas etapas. Essas etapas são referentes às duas disciplinas de Trabalho de Conclusão de Curso (TCC1 e TCC2), e podem ser visualizadas nas Figuras 8 e 9, respectivamente. A seguir serão detalhadas as atividades definidas em cada etapa.

4.2.1 TCC1

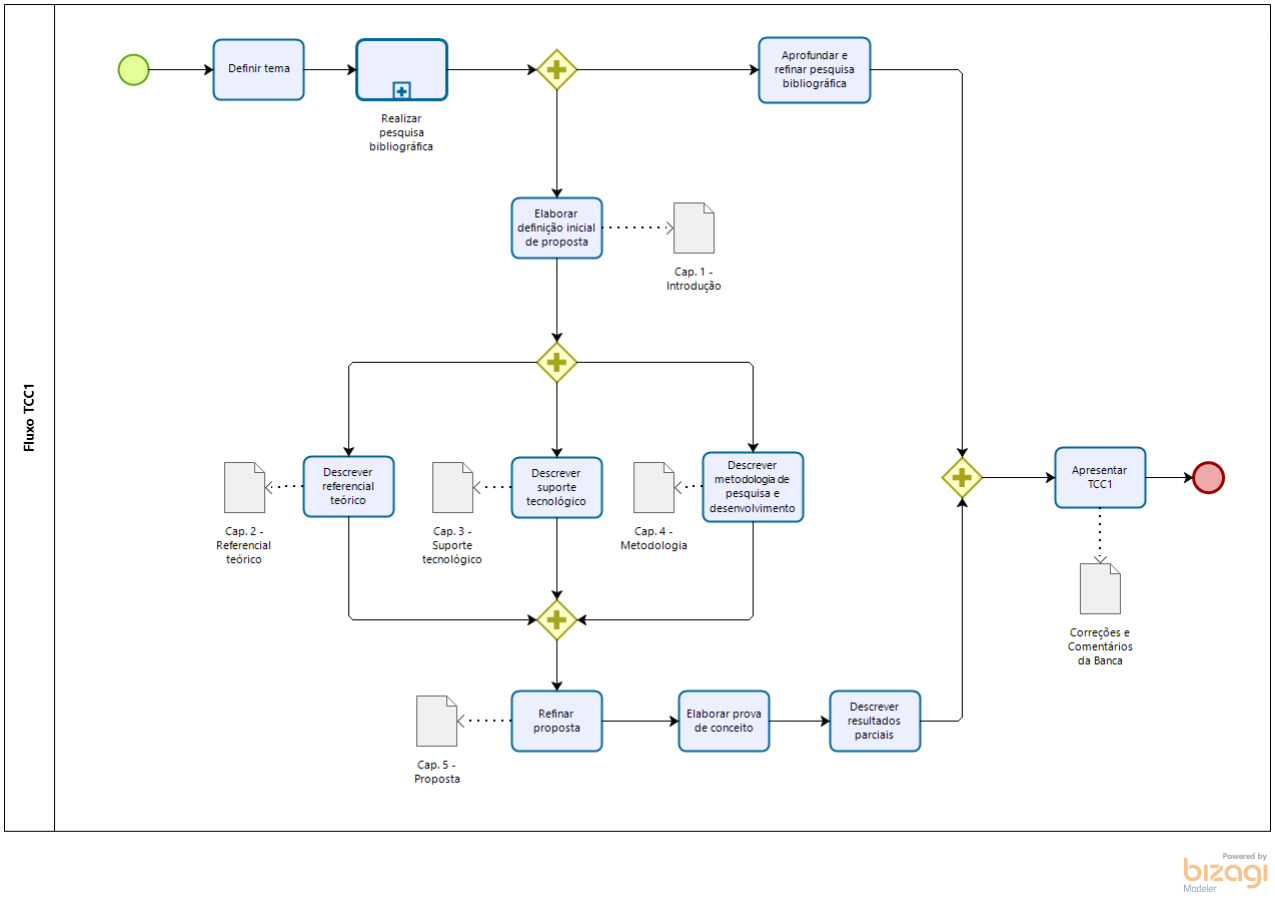
- **Definir tema**: Envolve a escolha da área geral em que se deseja realizar o trabalho, com definições iniciais de escopo e avaliação de possíveis trabalhos a serem desenvolvidos;

- **Realizar pesquisa bibliográfica**: A partir da definição de tema, inicia-se o levantamento de fontes iniciais, para refinar a visão sobre o contexto e permitir uma melhor adequação de escopo. Subprocesso descrito em maiores detalhes na Seção 4.2.3;

- **Elaborar definição inicial de proposta**: Contextualizar e delimitar o escopo do trabalho, estabelecendo e justificando questões de pesquisa e objetivos. O resultado dessa atividade é documentado no capítulo 1;

- **Aprofundar e refinar pesquisa bibliográfica**: Apresentada em mais detalhes na Seção 4.2.3, essa atividade é realizada ao longo de todo o trabalho. Evolui o levantamento inicial de fontes, com critérios mais rigorosos, de forma a estabelecer uma base teórica consistente e robusta para atender aos objetivos do trabalho;

Figura 8 – Fluxo de atividades do TCC1.

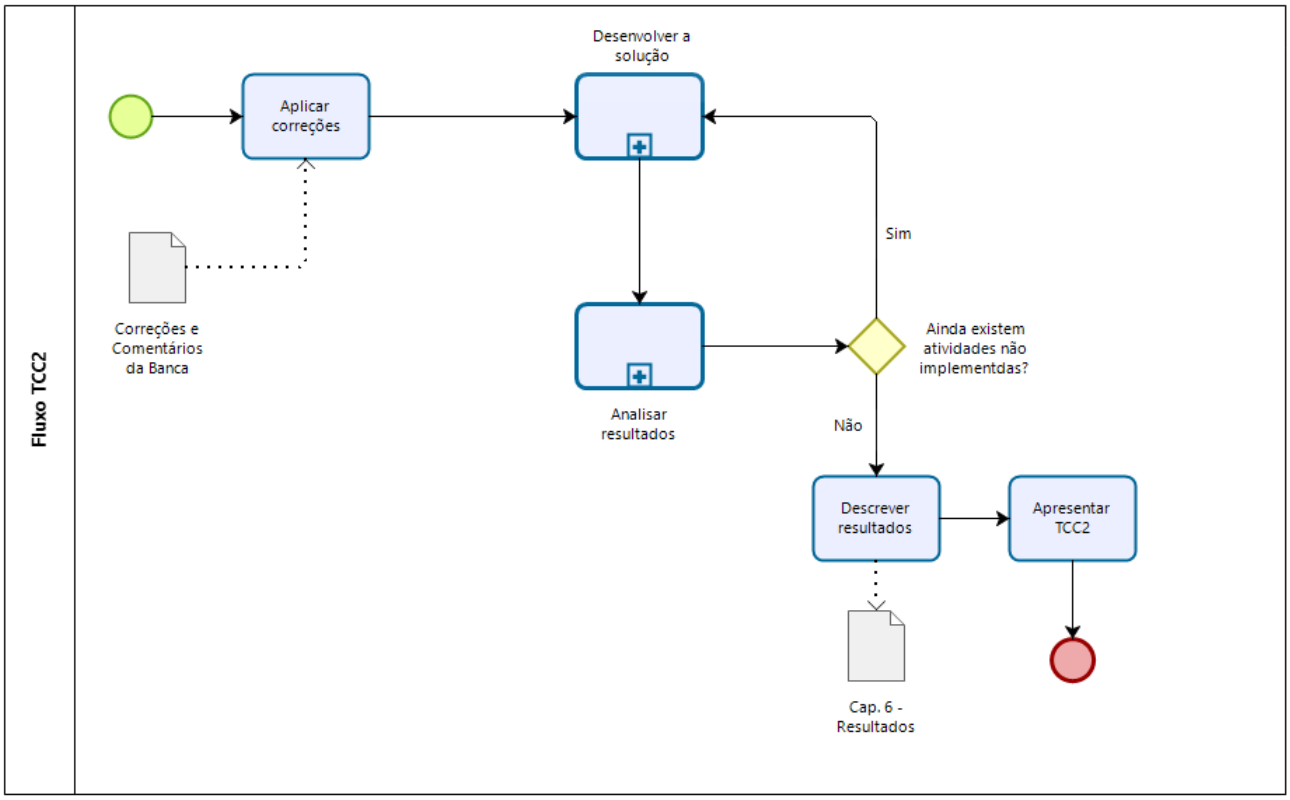


- **Descrever referencial teórico**: Utiliza os insumos obtidos na atividade anterior para descrever o referencial teórico necessário para a compreensão do trabalho. Resulta no Capítulo 2;
- **Descrever suporte tecnológico**: Define e descreve as ferramentas e apoios tecnológicos necessários para a execução do trabalho. Corresponde ao Capítulo 3;
- **Descrever metodologia de pesquisa e desenvolvimento**: Categoriza e descreve detalhadamente os procedimentos metodólogicos realizados durante a execução do trabalho. Tem seus resultados apresentados neste capítulo;
- **Refinar proposta**: A partir da melhor compreensão sobre o assunto, adquirida ao longo da elaboração das atividades anteriores, avalia o escopo previamente definido e realiza alterações, caso necessário;
- **Elaborar prova de conceito**: Averiguar a viabilidade de realização da proposta definida, implementando uma prova de conceito. Isso permite identificar possíveis riscos e estabelecer bases para a implementação concreta da solução, como protótipos de interface e configuração de ambiente;

- **Descrever resultados parciais**: Resume o estado do trabalho antes da defesa do TCC1, descrevendo o que foi alcançado até o momento e as perspectivas para o TCC2, e
- **Apresentar TCC1**: Apresentação dos resultados parciais à banca examinadora.

4.2.2 TCC2

Figura 9 – Fluxo de atividades do TCC2.

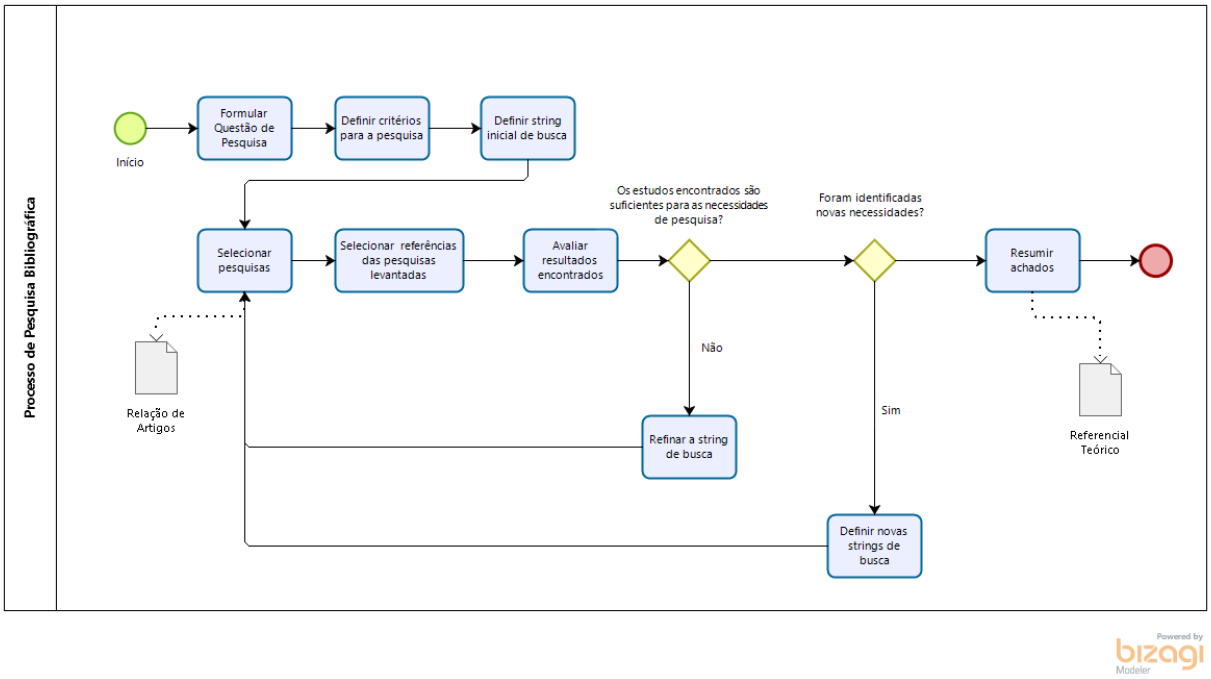


- **Aplicar correções**: De acordo com os comentários transmitidos pela banca durante a defesa do TCC1, adequar e corrigir a monografia;
- **Desenvolver a solução**: Implementar o apoio *online*, seguindo os requisitos elicitados na proposta e a metodologia de desenvolvimento definida. As atividades de desenvolvimento são descritas em maiores detalhes na Seção 4.2.4;
- **Analisar resultados**: Avaliar se a solução implementada atende aos objetivos propostos para o trabalho, de acordo com os procedimentos de pesquisa-ação. Esses procedimentos são descritos em maiores detalhes na Seção 4.2.5;

- **Descrever conclusões**: Resumir o estado final do trabalho, refletindo sobre os objetivos alcançados e propondo pontos passíveis de trabalhos futuros, e
- **Apresentar TCC2**: Apresentação dos resultados finais do trabalho à banca examinadora.

4.2.3 Fluxo de atividades da pesquisa bibliográfica

Figura 10 – Fluxo de atividades da pesquisa bibliográfica.



Adicionalmente, foi modelado o processo detalhando as atividades necessárias para a condução da pesquisa bibliográfica, de forma a esclarecer e organizar seus procedimentos. Esse fluxo pode ser visualizado na Figura 10. Uma relação dos critérios estabelecidos para a pesquisa, bem como o conjunto de *strings* de buscas utilizadas pode ser encontrado no Apêndice A. A seguir serão descritas as atividades que compõem o fluxo.

- **Formular questão de pesquisa**: Definir a questão norteadora do trabalho, o ponto de partida para a definição do que deve ser pesquisado;
- **Definir critérios para a pesquisa**: Definir critérios a serem utilizados para guiar os procedimentos de pesquisa, como a escolha de bases adequadas para o tema e fatores para determinar a inclusão ou não de uma pesquisa nos resultados;
- **Definir *string* inicial de busca**: Definir uma *string* de busca a ser pesquisada nas bases escolhidas, buscando obter resultados referentes a questão de pesquisa;

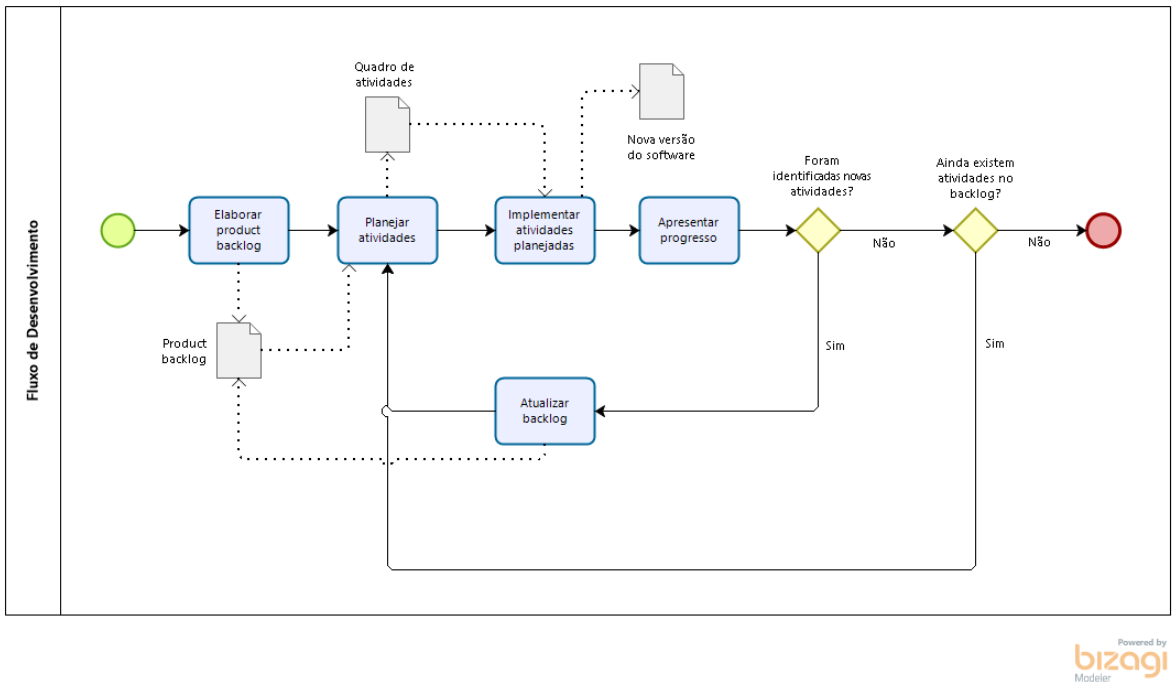
- **Selecionar pesquisas**: De acordo com os critérios estabelecidos, selecionar e armazenar as pesquisas avaliadas como mais relevantes para o trabalho;
- **Selecionar referências das pesquisas levantadas**: Procurar, dentre as pesquisas selecionadas, referências a outras pesquisas que possam ser relevantes ao trabalho. Caso existam, selecionar e armazená-las;
- **Avaliar resultados encontrados**: Avaliar se os resultados encontrados até o momento são suficientes para definir o arcabouço teórico necessário para a realização do trabalho. Caso ainda não sejam, realizar novas pesquisas;
- **Refinar a *string* de busca**: Caso os resultados de pesquisa para uma determinada *string de busca* não sejam suficientes para atender a uma ou mais necessidades estabelecidas, refinar a *string*, com o objetivo de obter novos resultados;
- **Definir novas *strings* de busca**: Caso sejam identificadas novas necessidades de pesquisa, que não possam ser atendidas por uma das *strings* de busca já existentes, elaborar uma ou mais *strings* de busca novas, e
- **Resumir achados**: Descrever as informações obtidas, que correspondem ao referencial teórico do trabalho.

4.2.4 Fluxo de atividades de desenvolvimento

A seguir são detalhadas as atividades que compõem o subprocesso de "Desenvolver a solução", apresentado na visão macro do fluxo do TCC2. As atividades foram definidas de acordo com a metodologia descrita na Seção 4.1.5, e podem ser visualizadas na Figura 11.

- **Elaborar *product backlog***: Envolve o levantamento e a priorização dos requisitos do *software*, organizados em uma lista, que é utilizada como base para a condução do desenvolvimento;
- **Planejar atividades**: Envolve a seleção de um conjunto de atividades do *backlog*, o qual é adicionado ao quadro *Kanban* daquela iteração do desenvolvimento;
- **Implementar atividades planejadas**: De acordo com o planejamento realizado, implementar as atividades selecionadas. Ao final dessa atividade, tem-se uma nova versão do sistema publicada;
- **Apresentar progresso**: Apresentar os resultados da última iteração, durante a reunião de orientação do TCC. Avaliar a necessidade de se incluir novas atividades ao *backlog*, e

Figura 11 – Fluxo de desenvolvimento do *software*.



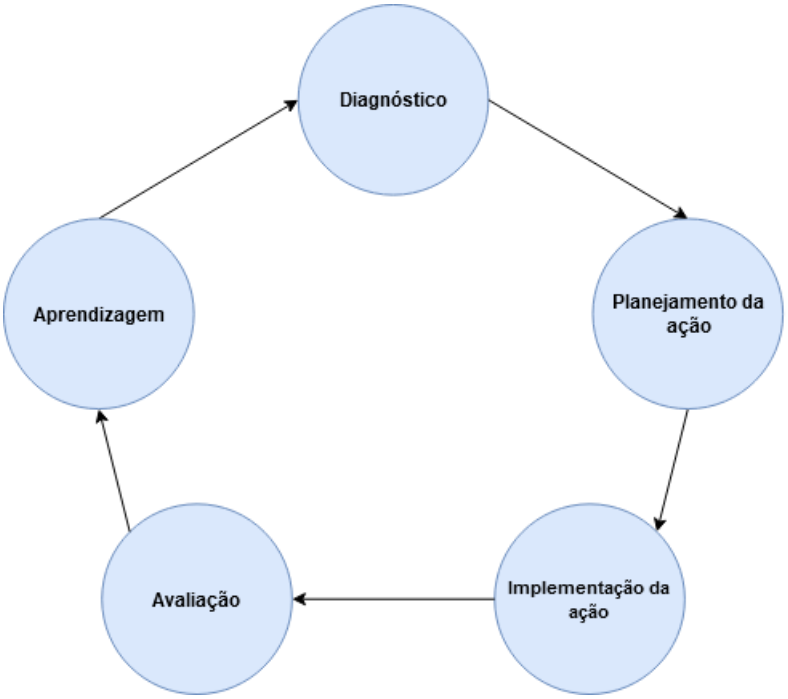
- **Atualizar *backlog***: De acordo com o *feedback* obtido, atualizar o *backlog* com as novas tarefas identificadas.

4.2.5 Fluxo de atividades de análise de resultados

Como acordado na Seção 4.1.4, a metodologia de pesquisa-ação foi utilizada para o processo de análise de resultados do trabalho. O protocolo definido está de acordo com o ciclo definido por (STARON, 2020), que trata da pesquisa-ação especificamente dentro da Engenharia de Software. Uma visualização desse ciclo pode ser vista na Figura 12, e suas atividades serão descritas a seguir.

- **Diagnóstico**: Envolve a identificação do problema e do contexto em que o mesmo está inserido, bem como a coleta de dados que auxiliem nessa definição;
- **Planejamento de ação**: A partir do diagnóstico estabelecido, planejar a intervenção a ser realizada, visando formas de resolver o problema identificado;
- **Implementação da ação**: Envolve a execução da intervenção planejada na etapa anterior, coletando dados que permitam a observação posterior de seus efeitos;
- **Avaliação**: Analisar os dados coletados, de forma a avaliar se a ação implementada atingiu os objetivos esperados. Essa análise pode servir de insumo para a elaboração de novos ciclos, e

Figura 12 – Ciclo de pesquisa-ação.



Fonte: (STARON, 2020), adaptado pelo autor.

- **Aprendizagem**: Envolve a documentação e a divulgação dos resultados adquiridos.

4.3 Cronograma de atividades

Os Quadros 3 e 4 apresentam o cronograma das Atividades referentes ao TCC1 e TCC2, respectivamente.

Quadro 3 – Cronograma de atividades do TCC1

Atividades	**Agosto**	**Setembro**	**Outubro**	**Novembro**	**Dezembro**
Definir Tema	X				
Levantar Fontes Bibliográficas Iniciais	X				
Elaborar Definição Inicial de Proposta	X				
Aprofundar e Refinar Pesquisa Bibliográfica	X	X	X	X	
Descrever Referencial Teórico		X			
Descrever Suporte Tecnológico			X		
Descrever Metodologia de Pesquisa e Desenvolvimento			X	X	
Refinar Proposta				X	
Elaborar Prova de Conceito				X	
Descrever Resultados Parciais				X	
Apresentar TCC1					X

Quadro 4 – Cronograma de atividades do TCC2

Atividades	**Fevereiro**	**Março**	**Abril**	**Maio**
Aplicar correções	X			
Desenvolver a solução	X	X	X	
Analisar resultados		X	X	
Descrever Resultados			X	X
Apresentar TCC2				X

4.4 Considerações finais do capítulo

Esse capítulo tratou das principais tomadas de decisão referentes à organização metodológica deste trabalho. Os elementos de pesquisa do projeto foram caracterizados de acordo com critérios bem definidos na literatura científica. Foram discutidos os principais procedimentos das etapas de pesquisa, desenvolvimento e análise de resultados do trabalho. Também foram detalhadas as atividades que constitutem a elaboração do trabalho como um todo, e a forma como essas atividades estão relacionadas entre si.

5 O apoio

Esse capítulo apresenta de forma detalhada o projeto implementado neste trabalho, ao longo de suas várias etapas de desenvolvimento. Inicialmente, a Seção 5.1 expõe um panorama de alto nível dos requisitos da solução.

A seguir, nas Seções 5.2 e 5.3, são detalhados os procedimentos e resultados do desenvolvimento do apoio, desde a implementação de provas de conceito iniciais e atividades exploratórias até a versão atual do *website*, já publicada, destacando e comentando pontos de interesse dos exemplos e do catálogo como um todo.

Mais adiante, são acordados alguns trabalhos relacionados, procurando abordar um quadro comparativo e destacando os diferenciais da presente proposta frente às propostas similares.

Por fim, são tecidas as considerações finais do capítulo.

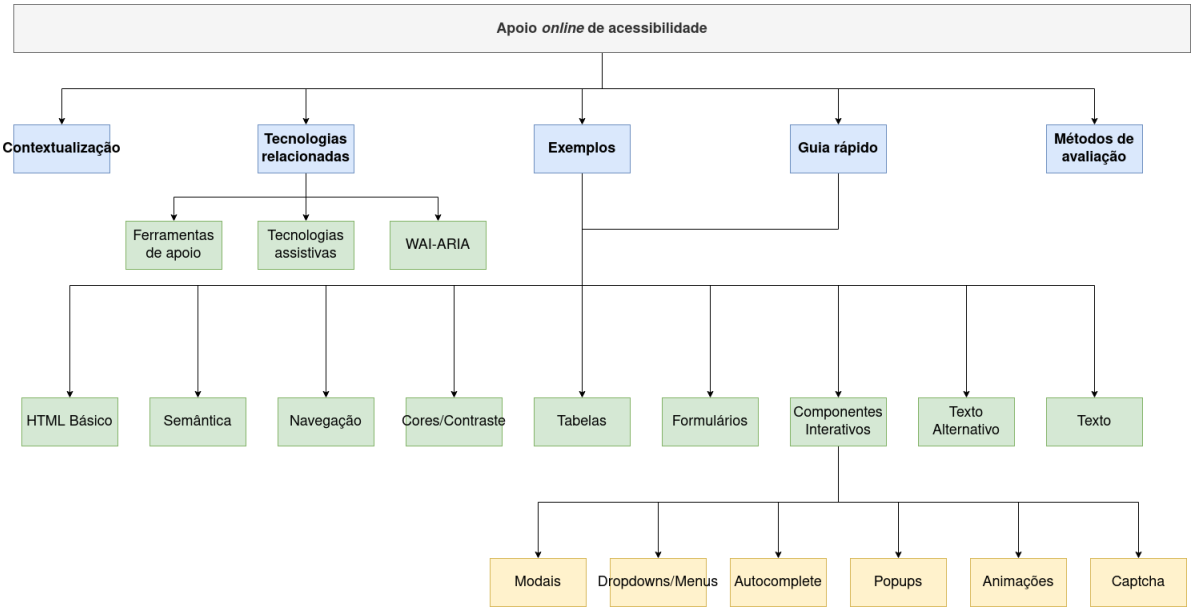
5.1 Requisitos do Apoio

5.1.1 Conteúdo da solução

A solução proposta disponibiliza informações relativas à acessibilidade de software, visando auxiliar desenvolvedores, com foco especial na acessibilidade da web. O apoio pode ser utilizado tanto para se familiarizar com o tema em um nível introdutório, quanto para obter informações específicas sobre o uso de ferramentas ou boas práticas de desenvolvimento. Uma visão geral da estrutura da solução, em relação ao seu conteúdo, pode ser vista na Figura 13.

- **Contextualização**: Envolve a apresentação ao usuário de tópicos básicos sobre o tema, como a própria definição de acessibilidade e a sua importância para o desenvolvimento de software;
- **Tecnologias relacionadas**: Trata do suporte tecnológico existente relacionado à acessibilidade, tanto para desenvolvedores quanto para usuários. Considera a apresentação conceitual das tecnologias, além de sugestões de uso. Foi dividido em três categorias:
  - **Ferramentas de apoio**: Expõe ferramentas que auxiliam o desenvolvedor a considerar certos aspectos de acessibilidade durante o desenvolvimento. Alguns exemplos são: ferramentas de análise de contraste e uso de cores, e avaliadores automáticos de conformidade;

Figura 13 – Estrutura do conteúdo do apoio *online*.



Fonte: Autor.

  - **Tecnologias assistivas**: Apresenta algumas das tecnologias assistivas mais comumente utilizadas, explicando seu uso e introduzindo considerações que devem ser levadas em conta para que uma aplicação seja acessível a essas tecnologias, e
  - **WAI-ARIA**: Introduz o WAI-ARIA, explicando seu propósito e os cuidados que devem ser tomados ao utilizar seus recursos.

- **Exemplos**: Envolve a construção e exibição de provas de conceito de interfaces e componentes, abordando aspectos diversos de acessibilidade. Vários exemplos são apresentados em duas versões: uma com problemas de acessibilidade, seguida de outra, acessível. De forma geral, os exemplos foram desenvolvidos para serem auto-contidos; isto é, um exemplo qualquer pode ser visualizado e compreendido individualmente, sem a necessidade de acompanhar o restante do catálogo inteiro para obter informações sobre um componente específico. Quando necessário, pré-requisitos ou sugestões de leituras anteriores são apresentadas durante a própria explicação do exemplo. Os exemplos foram catalogados nas seguintes categorias:

  - **HTML Básico**: Exemplos relacionados à estrutura básica de um documento HTML, com padrões que devem ser aplicados independente do conteúdo da página. Envolve a definição de atributos como o idioma do documento e regras para permitir redimensionamento da interface;
  - **Semântica**: Exemplos que destacam a importância do uso da semântica correta das *tags* HTML, tanto para a apresentação adequada do conteúdo, quanto para a interação com controles da página;

  - **Navegação**: Exemplos que destacam componentes e práticas essenciais para que os usuários consigam localizar e interagir corretamente com os elementos da interface. Demonstra como tecnologias assistivas tratam blocos específicos de uma página como cabeçalhos e *links*, e como estes devem ser declarados;
  - **Cores e contraste**: Exemplos envolvendo o uso adequado de cores e o contraste entre elementos em uma interface;
  - **Formulários**: Exemplos tratando a exibição e o comportamento de elementos de inserção de dados em uma página. Envolve aspectos como descrição correta de campos, agrupamento de componentes e validação e apresentação de erros;
  - **Tabelas**: Exemplos contemplando a exibição de tabelas em uma interface. Trata questões como a categorização correta das células de uma tabela e seus mecanismos de navegação, e
  - **Componentes interativos**: Exemplos envolvendo a implementação de componentes customizados, que fazem uso de JavaScript para a manipulação dos elementos da interface. Mais especificamente, traz exemplos de implementação de modais, *menus* de navegação com *dropdown* e *inputs* com *autocomplete*. Aprofunda-se ainda sobre práticas indesejadas da Web moderna, como uso excessivo e inadequado de *popups*, uso de animações de forma prejudicial e cuidados com o uso de ferramentas *Captcha*.

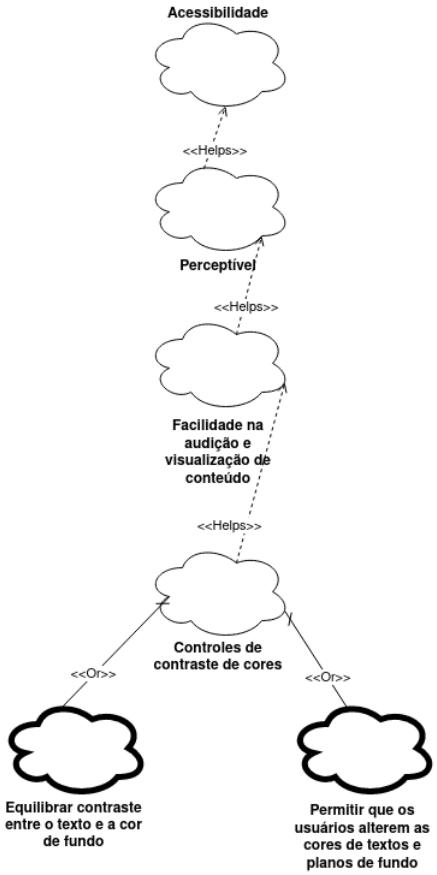
- **Guia Rápido**: Apresenta uma lista resumida dos assuntos tratados detalhadamente na seção de Exemplos. Foi implementado como uma alternativa menos densa em conteúdo aos exemplos, sendo ainda assim informativo. Além disso, fala brevemente sobre dois tópicos não detalhados na seção de exemplos, com dicas mais pontuais:

  - Texto: Apresenta alguns testes simples para verificar a acessibilidade do conteúdo textual de uma página, e
  - Texto alternativo: Considerações sobre a aplicação correta e incorreta de texto alternativo em imagens.

- **Métodos de Avaliação**: Tópico destinado à exposição de métodos de avaliação da acessibilidade de um sistema, manuais e automatizados, bem como formas de combiná-los com o objetivo de se realizar uma avaliação abrangente.

A definição e a implementação dos exemplos, que constituem a maior parte do apoio, foram orientadas pelas fontes acadêmicas e técnicas reunidas ao longo das fases anteriores do trabalho. As recomendações presentes no WCAG serviram como um ponto de partida sólido, visto se tratar de um padrão reconhecido internacionalmente e em

Figura 14 – Recorte do SIG de acessibilidade.



Fonte: Autor, adaptado de (OLIVEIRA et al., 2016) e (OLIVEIRA, 2014).

constante revisão e evolução. O mapeamento das diretrizes do WCAG em exemplos concretos e objetivos foi baseado no catálogo elaborado por (OLIVEIRA, 2014), que modela requisitos de acessibilidade através do *NFR Framework*.

O *NFR Framework*, como definido por (CHUNG et al., 2000), propõe uma estrutura para modelar requisitos não-funcionais através de grafos, conhecidos como *softgoal interdependency graphs* (SIGs). Esse modelo considera os requisitos não-funcionais como *softgoals* (*metas flexíveis*), relacionados entre si. Esses *softgoals* podem ser decompostos e refinados em diversos níveis, com os últimos níveis correspondendo a *operacionalizações*: possíveis operações a serem implementadas no sistema, com o objetivo de alcançar os *softgoals*.

O catálogo mencionado utiliza esse modelo para organizar as diretrizes do WCAG, partindo de seus princípios mais abstratos, até chegar nas técnicas e nos procedimentos que permitem o cumprimento desses princípios (OLIVEIRA et al., 2016). Um recorte desse grafo pode ser visualizado na Figura 14.

5.1.2 Interface da solução

A implementação do apoio *online* foi sujeita à aplicação das próprias técnicas que compõem seu conteúdo. O seu *design* prezou pela acessibilidade e conformidade com as normas estudadas, sempre que possível. Fugiu a esse princípio um caso em específico: os exemplos de componentes e interfaces incorretas, que intencionalmente não contemplaram os princípios de acessibilidade.

Para os demais elementos da solução, foram definidos alguns pontos de atenção especial, destacados a seguir. Esses pontos, elicitados a partir do WCAG, correspondem às considerações de acessibilidade mais relevantes ao contexto do apoio, de acordo com o seu conteúdo.

- **Estrutura textual**: Os elementos textuais da página possuem organização semântica coerente, com marcação adequada de títulos, *links* e parágrafos. Isso facilita a navegação via teclado e/ou leitores de tela, permitindo a identificação rápida de tópicos e seções da página;
- **Contraste**: O contraste entre elementos de texto e seu *background* é suficiente para que usuários, com diferentes capacidades visuais, consigam perceber o texto corretamente. A razão de contraste utilizada foi de 7:1 para textos de até 18pt e 4,5:1 para textos maiores, de acordo com as recomendações presentes em (W3C, 2018a);
- **Proporções e responsividade**: a implementação da solução permite o redimensionamento da interface, sem perda de conteúdo ou comprometimento do *layout*. O tamanho proporcional de cada elemento, bem como seu posicionamento em relação a outros elementos, é mantido ao se aplicar *zoom* na página ou ao acessar o *site* em dispositivos com tamanhos de tela variados, e
- **Foco**: Elementos da interface como botões e *hyperlinks* devem ter sua apresentação destacada quando estiverem em foco. A ausência de indicação de foco inviabiliza a navegação exclusivamente através do teclado. A mudança de foco dos elementos da página deve seguir uma ordem lógica, que seja coerente com a apresentação do conteúdo.

5.2 Provas de Conceito

5.2.1 Atividades exploratórias

Desde a concepção inicial do trabalho, antes da definição concreta da proposta, foram realizadas algumas atividades exploratórias com tecnologias assistivas. Essas atividades podem ser comparadas às *screening techniques* descritas na Seção 2.4, embora com

objetivos distintos. Ao invés de se tentar avaliar a acessibilidade de um sistema em específico, buscou-se familiarizar com ferramentas como leitores de tela e teclados alternativos, utilizando-os para acessar um número diverso de *sites* e aplicativos.

Foi possível identificar problemas na maior parte dos produtos de software explorados, de gravidades variadas. Algumas das falhas mais comuns foram erros no fluxo de navegação e falta de indicação de foco, ao se utilizar a interface via teclado ou com auxílio de leitor de tela. Alguns programas mostraram-se completamente inacessíveis, com nenhum elemento da interface sendo reconhecido por ferramentas assistivas. Outros mostraram-se inconsistentes, como o caso de uma aplicação de comunicação instantânea que eventualmente ignorava o conteúdo de algumas mensagens ao se utilizar a aplicação com leitor de tela, embora funcionasse normalmente na maior parte do tempo. Também houve casos de aplicações com acessibilidade adequada em sua versão *mobile*, mas praticamente inutilizável na versão *desktop*. Ressalta-se que uma parcela das dificuldades pode estar relacionada à falta de experiência com o uso das ferramentas.

Ainda pôde-se investigar os diferentes níveis de maturidade das ferramentas de apoio através de ambientes distintos. Em sistemas *mobile*, o pacote de tecnologias assistivas costuma já vir instalado com o sistema. Isso também acontece no Windows 10, embora o usuário possa instalar outras ferramentas posteriormente, de acordo com suas preferências. No Linux, o cenário varia de acordo com a distribuição e o ambiente de *desktop* utilizado, podendo exigir maiores esforços de configuração e customização. Ferramentas mais simples, como ampliador de tela e teclado virtual, funcionam sem grandes dificuldades independente do sistema.

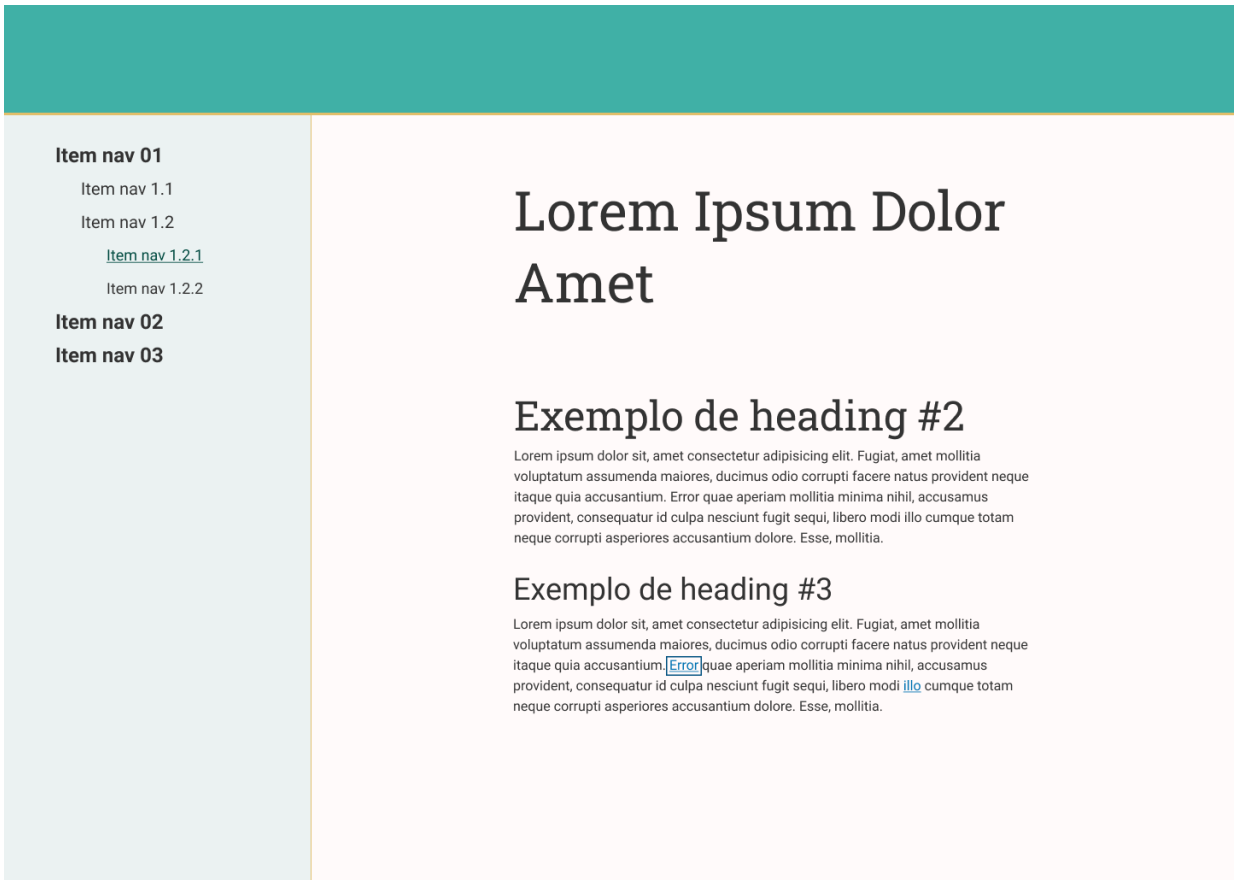
Essas atividades permitiram uma melhor compreensão do assunto estudado, provendo novas perspectivas e possibilitando a identificação de questões de acessibilidade sob uma ótica diferente da obtida ao ler materiais de referência ou utilizar ferramentas de análise automática.

5.2.2 Protótipo

Foi construído um protótipo da interface do apoio, com o objetivo de explorar alternativas de *design* e melhor visualizar formas de alcançar os requisitos definidos na Seção 5.1.2. Para isso, utilizou-se a plataforma Figma, conforme acordado no Capítulo 3. A principal tela desse protótipo está presente na Figura 15.

A escolha de cores foi orientada de forma a manter o contraste entre os elementos textuais e os planos de fundo, visando legibilidade adequada. Procurou-se ainda garantir que a interface seja completamente visível para usuários com diferentes tipos de daltonismo. Para avaliar esses dois pontos, foram utilizados dois *plugins* de acessibilidade: o primeiro gera novas versões de uma tela selecionada, modificando suas cores de forma a

Figura 15 – Protótipo do apoio, versão *desktop*.



Fonte: Autor.

simular os principais tipos de daltonismo; o segundo avalia o contraste entre os elementos da tela, informando o nível de conformidade com as regras de contraste do WCAG. Uma dessas telas é exibida na Figura 16, onde as cores do protótipo simulam a condição de deuteranopia, forma comum de daltonismo que prejudica a visualização da cor verde. O resultado de execução do *plugin* de contraste pode ser visto na Figura 17.

Em relação à estrutura textual, a dimensão dos parágrafos limita a quantidade de caracteres por linha, em torno de 70 a 80. Também não se utiliza texto duplamente justificado, mantendo o alinhamento apenas à esquerda. Essas duas medidas seguem recomendações para aumentar a legibilidade do conteúdo. Optou-se pela escolha de fontes amplamente conhecidas, de forma a proporcionar familiaridade para um maior número de usuários. Portanto, foram escolhidas as fontes **Roboto** e **Roboto Slab**, utilizadas em várias aplicações do Google. A Roboto, fonte padrão do Android, será utilizada para a maior parte do conteúdo textual do apoio. Por sua vez, a Roboto Slab servirá principalmente para o destaque de títulos e seções.

O recorte do protótipo apresentado na Figura 18 mostra o estado de um *hyperlink* sob o foco do teclado. O indicador de foco também é coerente com as normas de contraste, e permite identificar com clareza o elemento focado em um determinado momento, o

Figura 16 – Protótipo com filtro de cores simulando deuteranopia.

Fonte: Autor.

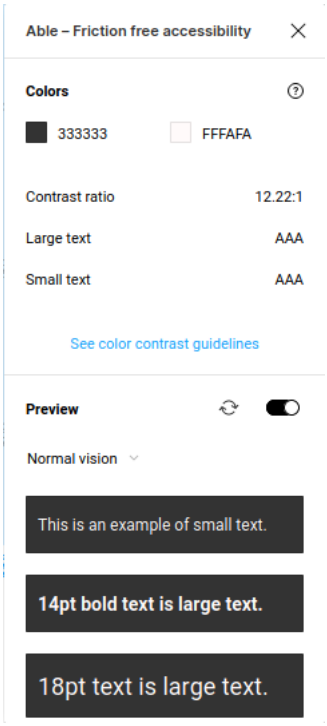
diferenciando do outro *link* ao final do parágrafo.

As Figuras 19a e 19b exemplificam a visualização da interface em uma tela de *smartphone*, demonstrando o redimensionamento dos elementos. O menu de navegação é recolhido por padrão, e pode ser aberto ao pressionar o botão na barra superior. Os elementos em foco em cada imagem ilustram ainda o fluxo lógico de navegação da página. Na primeira imagem, o foco do teclado encontra-se no botão de navegação. Ao ser clicado, se o usuário navegar para o próximo elemento focável (apertando a tecla *Tab*, por exemplo), o indicador de foco passa para o primeiro elemento do menu de navegação, como visto na segunda imagem. Se o menu ainda estivesse fechado, essa ação deveria levar o indicador de foco ao primeiro *link* existente no conteúdo do artigo.

5.2.3 Configuração do ambiente

Foi implementada uma versão mínima da interface desenhada na seção anterior, com o objetivo de testar e configurar as ferramentas de desenvolvimento propostas no Capítulo 3. Uma visão geral da estrutura de pacotes da solução pode ser vista na Figura 20. A mesma será detalhada a seguir.

Figura 17 – Execução do *plugin* de contraste no protótipo.



Fonte: Autor.

Figura 18 – Paragráfo com exemplo de *hyperlink* focado.

Lorem ipsum dolor sit, amet consectetur adipisicing elit. Fugiat, amet mollitia voluptatum assumenda maiores, ducimus odio corrupti facere natus provident neque itaque quia accusantium. Error quae aperiam mollitia minima nihil, accusamus provident, consequatur id culpa nesciunt fugit sequi, libero modi illo cumque totam neque corrupti asperiores accusantium dolore. Esse, mollitia.

Fonte: Autor.

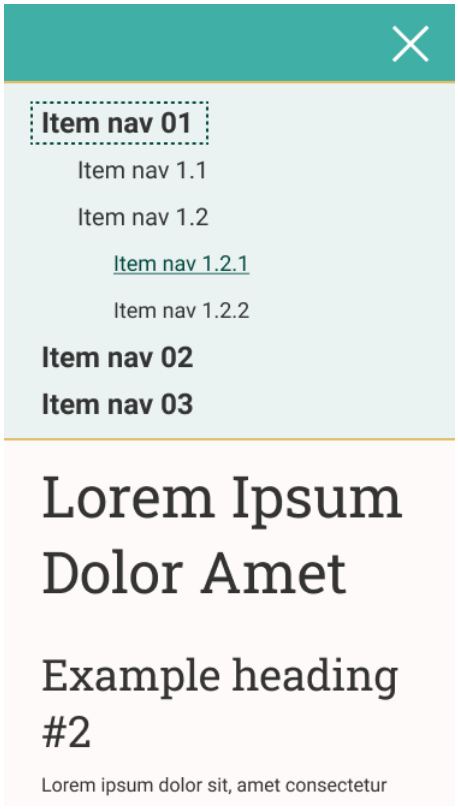
- **src**: Possui os subpacotes de implementação da solução. Em **__includes**, localizam-se os arquivos de *template*, que contém o *layout* geral das páginas do *site*, sem conteúdo definido. Em **sass**, localizam-se as folhas de estilo do projeto, que definem as regras de apresentação dos componentes da interface. Por fim, **pages** contém os arquivos de conteúdo do apoio, escritos em formato *Markdown*.

- **public**: Contém os artefatos de *build* do *website*, utilizados para a publicação do apoio. O subpacote **pages** contém as páginas HTML construídas a partir da combinação do conteúdo dos arquivos *Markdown* com o *layout* dos arquivos de *template*. O subpacote **css** contém as folhas de estilo em CSS3, resultantes do processamento dos arquivos *sass*.

A partir dessa estrutura, construiu-se um *layout* básico, considerando vários dos

Figura 19 – Protótipo do apoio, versão *mobile*.

(a) Navegação contraída.



(b) Navegação expandida.

pontos estabelecidos no protótipo. Também foi criado um exemplo de arquivo de conteúdo. A página resultante pode ser vista na Figura 21.

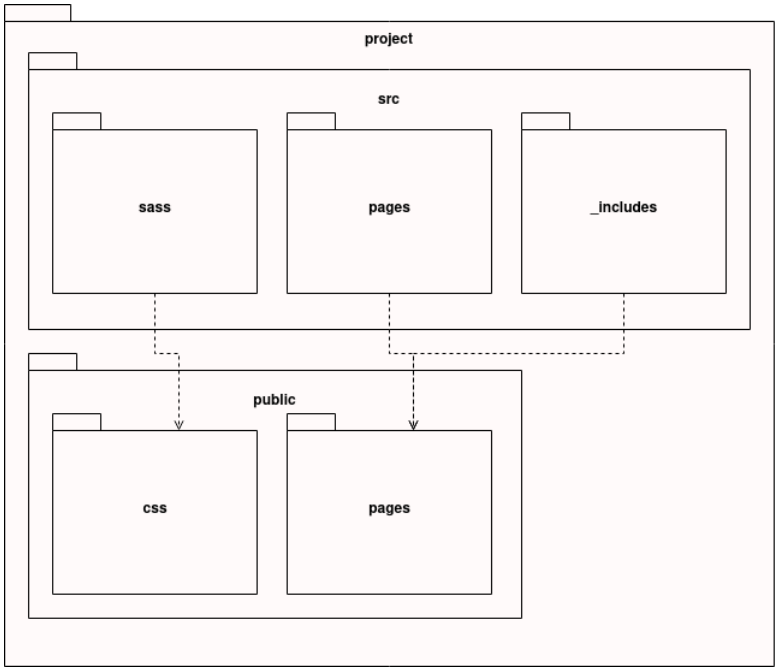
Como uma forma de avaliação preliminar, executou-se o WAVE na página. O relatório gerado, disponível na Figura 22, não apontou erros de conformidade. Esse tipo de análise é útil para avaliar certos aspectos de acessibilidade da página, como possíveis erros de contraste ou da estrutura de tópicos.

5.3 Solução Desenvolvida

Com base em todos os requisitos levantados e os insumos obtidos com as provas de conceito, prosseguiu-se com o desenvolvimento concreto do apoio. O mesmo pode ser acessado através da URL <https://tender-leavitt-ddf88b.netlify.app>. A Figura 23 mostra a versão mais atual da página inicial do projeto.

As atividades de desenvolvimento foram organizadas em um *backlog*, de acordo com a metodologia proposta no Capítulo 4. Esse *backlog*, apresentado na Figura 24, foi separado em três níveis macro de tarefas, representando fases distintas da implementação. No primeiro nível, foram estabelecidas funcionalidades básicas para o funcionamento do *website*, como implementação dos principais componentes reutilizáveis, definindo o for-

Figura 20 – Diagrama de pacotes da solução.



Fonte: Autor.

Figura 21 – Página de teste do ambiente.

Fonte: Autor.

mato de elementos como blocos de código, imagens, *links* e listas. Também se definiu a implementação da barra de navegação do apoio, que é gerada automaticamente a partir das páginas existentes. Por fim, configurou-se o *deploy* automatizado do apoio, integrado ao repositório no Git.

Com a base técnica do projeto implementada, os níveis seguintes de atividades corresponderam à escrita dos artigos que constituem o apoio. No segundo nível,

Figura 22 – Página de teste do ambiente, após execução do WAVE.

Fonte: Autor.

Figura 23 – Página inicial do apoio implementado.

Fonte: Autor.

concentraram-se os tópicos de contextualização, tecnologias e métodos de avaliação, enquanto o terceiro nível foi dedicado exclusivamente aos exemplos.

O formato dos exemplos variou de acordo com cada tópico, mas de forma geral cada artigo foi escrito de forma a:

Figura 24 – *Backlog* de atividades do projeto.

Grupo de Atividades	*Task*
Estrutura básica do *website*	Implementar *layout* básico de páginas de conteúdo
	Definir formato de alguns componentes (trechos de código, imagens, *links*)
	Implementar corretamente *navbar* com *links* para qualquer parte do site
	Configurar *deploy*
Textos informativos	Escrever *home*
	Escrever introdução
	Escrever sobre tecnologias - geral
	Escrever sobre tecnologias - teclado e leitores de tela
	Escrever sobre tecnologias - ARIA
	Escrever sobre tecnologias - Ferramentas de apoio
	Escrever sobre métodos de avaliação
Exemplos	Exemplos - Estrutura básica do HTML
	Exemplos - Semântica do HTML
	Exemplos - Cores e Contraste
	Exemplos - Navegação
	Exemplos - *Forms*
	Exemplos - Tabelas
	Exemplos - Interativos - Modais
	Exemplos - Interativos - *Menus*
	Exemplos - Interativos - *Autocomplete*
	Práticas a se evitar - *Popups*
	Práticas a se evitar - Animações
	Práticas a se evitar - *Captcha*
	Guia rápido - Resumo dos exemplos
	Guia rápido - Elementos textuais
	Guia rápido - Textos alternativos (imagens)

Fonte: Autor.

- Explicar o conceito sendo retratado naquele exemplo;
- Explicar como tecnologias assistivas interagem com o tipo de elemento descrito;
- Apresentar, na própria página, provas de conceito curtas dos conceitos descritos, com trechos de código, imagens e/ou áudio;
- Apresentar, em páginas separadas, provas de conceito mais elaboradas sobre o tópico, com instruções de atividades a serem realizadas para melhor compreensão do assunto, e
- Referenciar recursos adicionais sobre o assunto, especialmente as diretrizes utilizadas para a construção dos exemplos.

A seguir serão explicados em mais detalhes os exemplos apresentados no apoio.

5.3.1 Exemplos

5.3.1.1 HTML Básico

Essa seção partiu de um mesmo bloco básico de HTML (visto na Figura 25) para a apresentação de dois tópicos distintos, mas essenciais para a acessibilidade de uma página:

a definição do idioma do documento e dos metadados de *viewport*.

Figura 25 – Código de exemplo para a seção de HTML Básico.

```
<!DOCTYPE html>
<html lang="pt">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Document</title>
</head>
<body>
  <header>...</header>
  <nav>...</nav>
  <main>...</main>
  <footer>...</footer>
</body>
</html>
```

Fonte: Autor.

A partir desse bloco, foram refinados exemplos específicos. No primeiro caso, utilizou-se de trechos em áudio para demonstrar a diferença de funcionamento de um leitor de tela ao definir correta e incorretamente a *tag* de idioma.

Para o segundo caso, exibiu-se uma versão modificada do próprio apoio para demonstrar a visualização *mobile* prejudicada ao se remover os metadados de *viewport*, como visto na Figura 26a. Para comparação, essa mesma página com os atributos corretos é exibida na Figura 26b.

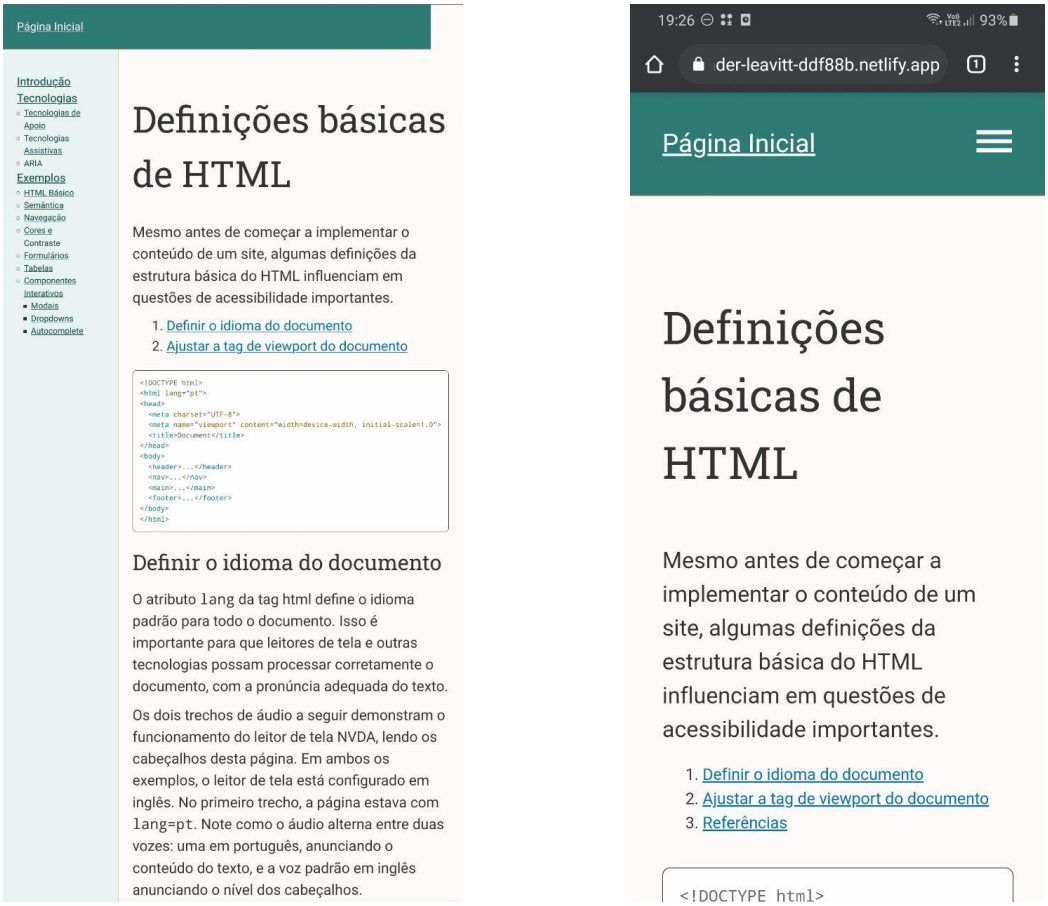
5.3.1.2 Semântica

Para esta seção, foram apresentados pequenos trechos de código ao longo da explicação dos conceitos. Esses trechos foram reunidos e utilizados ao final para a construção de um exemplo maior, implementado à parte. Esse exemplo final apresenta uma versão acessível e inacessível de uma mesma página, e o leitor é convidado a explorar as duas versões com um leitor de tela e navegando através do teclado, investigando as diferenças entre cada uma. Vale destacar que visualmente as páginas são idênticas, apesar de possuírem grandes diferenças estruturais.

O exemplo incorreto, visto na Figura 27, utiliza apenas tags genéricas `div` e `span` para declarar os elementos, utilizando de CSS para estilizá-los como cabeçalhos e botões. Dessa forma, apesar de sua aparência, o botão não é navegável por teclado, e os cabeçalhos não são reconhecidos por leitores de tela.

Já a versão correta, vista na Figura 28, utiliza as *tags* apropriadas para o contexto e é acessível por padrão.

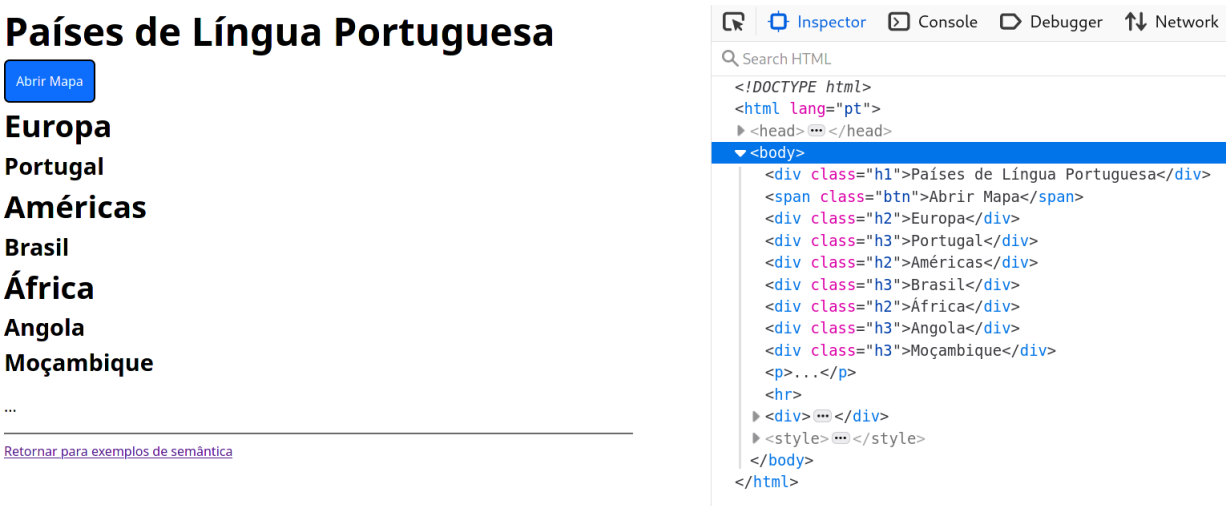
Figura 26 – Comparação de página *mobile* sem e com metadados de *viewport*.



(a) Sem *tag* de *viewport*. (b) Com *tag* de *viewport*.

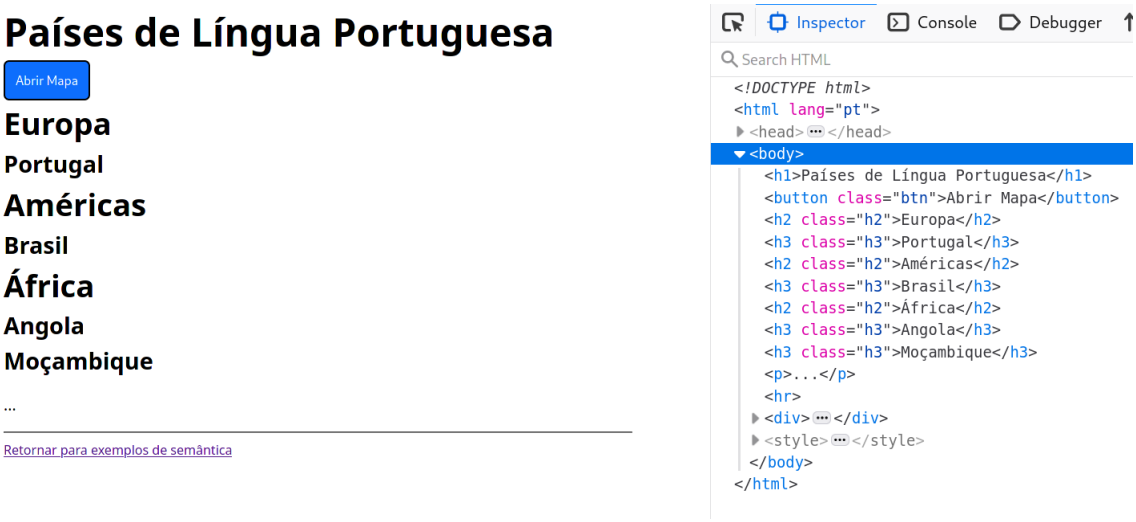
Fonte: Autor.

Figura 27 – Exemplo incorreto de semântica.



Fonte: Autor.

Figura 28 – Exemplo correto de semântica.

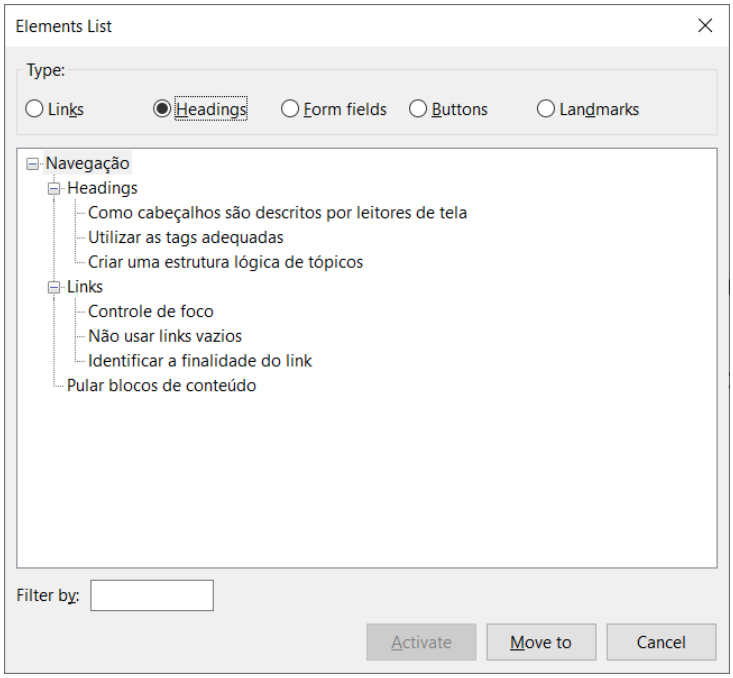


Fonte: Autor.

5.3.1.3 Navegação

Esta seção deu um enfoque maior à compreensão de como usuários com deficiências navegam pelas páginas de um *site*, e que técnicas e ferramentas são proporcionadas a eles pelas tecnologias assistivas. Por exemplo, a Figura 29 mostra como o leitor de tela NVDA processa os cabeçalhos de uma página em uma estrutura de tópicos.

Figura 29 – Estrutura de cabeçalhos do NVDA.

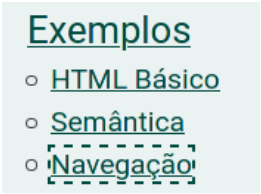


Fonte: Autor.

Para os exemplos, utilizou-se da estrutura do próprio apoio, demonstrando as funcionalidades de acessibilidade implementadas, como o controle de foco para os *links*

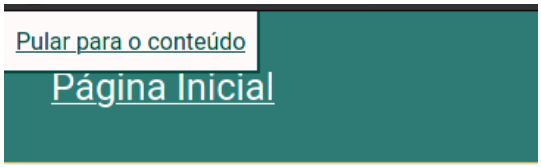
(Figura 30) e o *link* especial de "Pular para o conteúdo"(Figura 31). Esse tipo de *link* reduz o esforço de usuários que dependem do teclado para navegar, fazendo com que não precisem passar por todos os *links* da barra de navegação para acessar o conteúdo principal da página.

Figura 30 – Barra de navegação do apoio, com *link* sob foco do teclado.



Fonte: Autor.

Figura 31 – Exemplo de *link* para pular para o conteúdo principal.

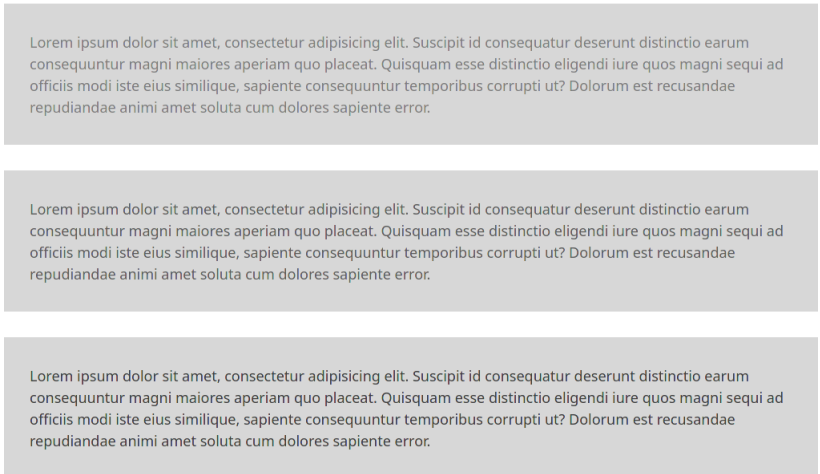


Fonte: Autor.

5.3.1.4 Cores e contraste

Dado o contexto particularmente visual desse tópico, optou-se pelo uso exclusivo de imagens para os exemplos, ao contrário de outras seções, que fizeram maior uso de trechos de código. A Figura 32 foi utilizada para exemplificar diferentes níveis de contraste, com o primeiro trecho abaixo de qualquer padrão de acessibilidade; o segundo no nível AA, e o terceiro no nível AAA do WCAG.

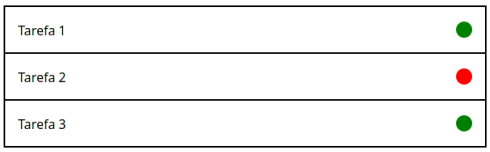
Figura 32 – Exemplo de diferentes razões de contraste.



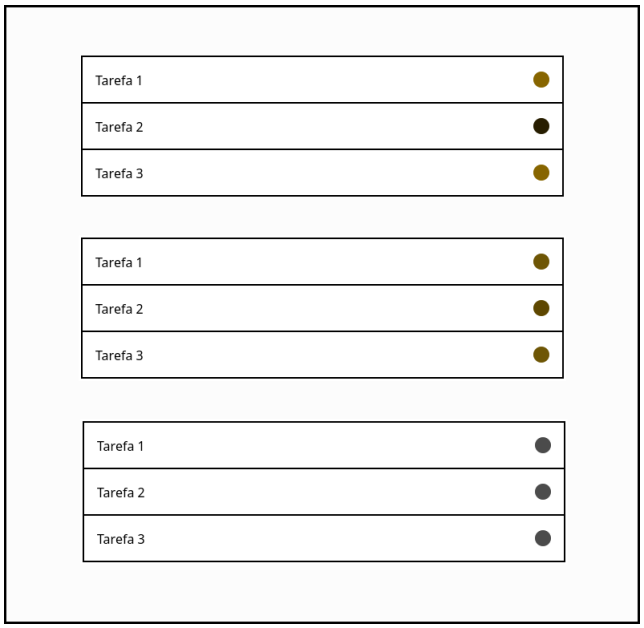
Fonte: Autor.

Já as Figuras 33a e 33b demonstram porque não utilizar apenas cor para transmitir informações em uma interface. Os status de sucesso e falha (representados respectivamente pelas cores verde e vermelha) são perdidos para usuários com certos tipos de daltonismo.

Figura 33 – Exemplo de uso de cores para transmitir informações.



(a) Lista com indicativos de cores.



(b) Lista com indicativos de cores, simulação de tipos de daltonismo.

Fonte: Autor.

5.3.1.5 Formulários - considerações básicas

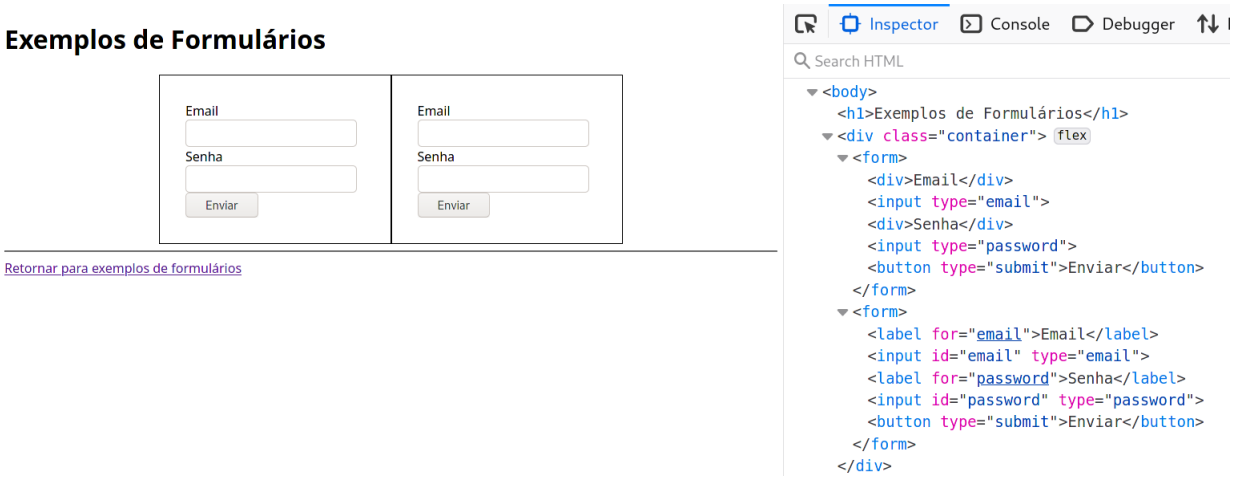
Esta seção fez uso da mesma abordagem do artigo sobre semântica, criando duas interfaces visualmente idênticas, mas com implementações drasticamente diferentes do ponto de vista da acessibilidade, instigando o usuário a investigar as diferenças entre cada uma.

Essa comparação pode ser vista na Figura 34. O primeiro `form` não utiliza `labels` para rotular seus campos, utilizando apenas `divs`. Dessa forma, tecnologias assistivas não conseguem associar uma descrição aos *inputs*, podendo causar dificuldades de compreensão para os usuários.

Em complemento ao exemplo maior, também foram utilizados pequenos trechos de código e controles de formulários na própria página. Por exemplo, a Figura 35 mostra um trecho de código com aplicação incorreta de uma `label`, que é declarada, mas não associada ao respectivo `input`.

Já a Figura 36 mostra a forma correta de se declarar *inputs* agrupados, como conjuntos de `checkboxes`, exibindo tanto o trecho de código quanto a interface em si.

Figura 34 – Comparação entre formulários.



Fonte: Autor.

Figura 35 – Uso incorreto de *label*.

```
<label>Nome</label>
<input id="nome" type="text">
```

Fonte: Autor.

Figura 36 – Exemplo de uso de `fieldset`.

Idiomas

Inglês ☐
Português ☐
Espanhol ☐

Grupos de controles relacionados devem ser contidos dentro de uma tag `fieldset`. A descrição desse fieldset deve ser rotulada pela tag `legend`. Para cada controle dentro do fieldset, as regras de `label` já descritas ainda se aplicam.

```
<fieldset>
<legend>Idiomas</legend>
<div>
  <label for="english">Inglês</label>
  <input type="checkbox" id="english">
</div>
<div>
  <label for="portuguese">Português</label>
  <input type="checkbox" id="portuguese">
</div>
<div>
  <label for="spanish">Espanhol</label>
  <input type="checkbox" id="spanish">
</div>
</fieldset>
```

Fonte: Autor.

5.3.1.6 Formulários - validações e mensagens de erro

Esta seção também contou com vários exemplos menores ao longo da página, culminando em duas interfaces mais elaboradas, para duas abordagens distintas de validação. Essa seção é a primeira a contar com maior uso de JavaScript e de atributos da ARIA.

A Figura 37 mostra um exemplo de validação básica *client-side*, utilizando-se dos atributos `required` e `aria-required="true"` para informar de forma acessível que um campo é obrigatório.

Figura 37 – Exemplo de validação *client-side* com `required` e `aria-required`.

Fonte: Autor.

Já as Figuras 38 e 39 exemplificam situações com mensagens de erro mais complexas, possivelmente após receber uma resposta *server-side*. A Figura 38 demonstra o uso do atributo `aria-describedby`, que permite conectar um *input* à sua mensagem de erro, fazendo com que tecnologias assistivas anunciem a mensagem de erro ao navegar pelo *input*. Já a Figura 39 apresenta um trecho de código em JavaScript, para que, ao clicar em uma mensagem de erro, o foco do teclado seja redirecionado para o *input* associado.

Figura 38 – Exemplo de uso do atributo `aria-describedby`.

```
<fieldset>
<legend>Erros</legend>
<ul>
  <li id="error-password">Senha deve conter pelo menos um caracter especial.</li>
  <li id="error-date">Ano de nascimento deve ser superior a 1900.</li>
</ul>
</fieldset>
<form>
<div>
  <label for="password">Senha</label>
  <input id="password" type="password" aria-describedby="error-password" invalid>
</div>
<div>
  <label for="birthday">Data de nascimento</label>
  <input id="birthday" type="date" aria-describedby="error-date" invalid>
</div>
</form>
```

Fonte: Autor.

Figura 39 – Controle de foco para mensagens de erro, em JavaScript.

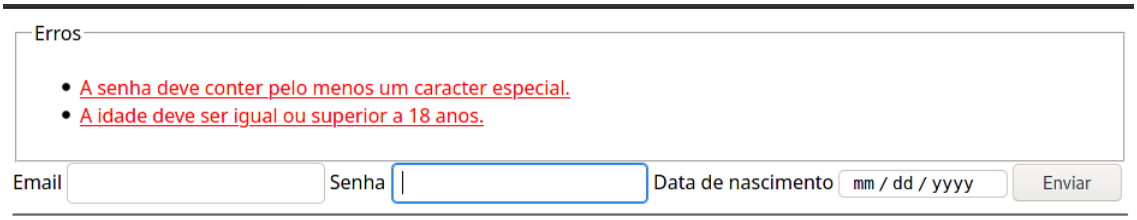
```
const errorLink = document.querySelector('a#error');
const referencedControl = document.querySelector('#some_input');

errorLink.onclick = (event) => {
  referencedControl.focus();
  return event.preventDefault();
}
```

Fonte: Autor.

As duas técnicas descritas acima foram utilizadas na construção do exemplo de interface da Figura 40.

Figura 40 – Exemplo de exibição de erros ao topo da página.



Fonte: Autor.

5.3.1.7 Tabelas

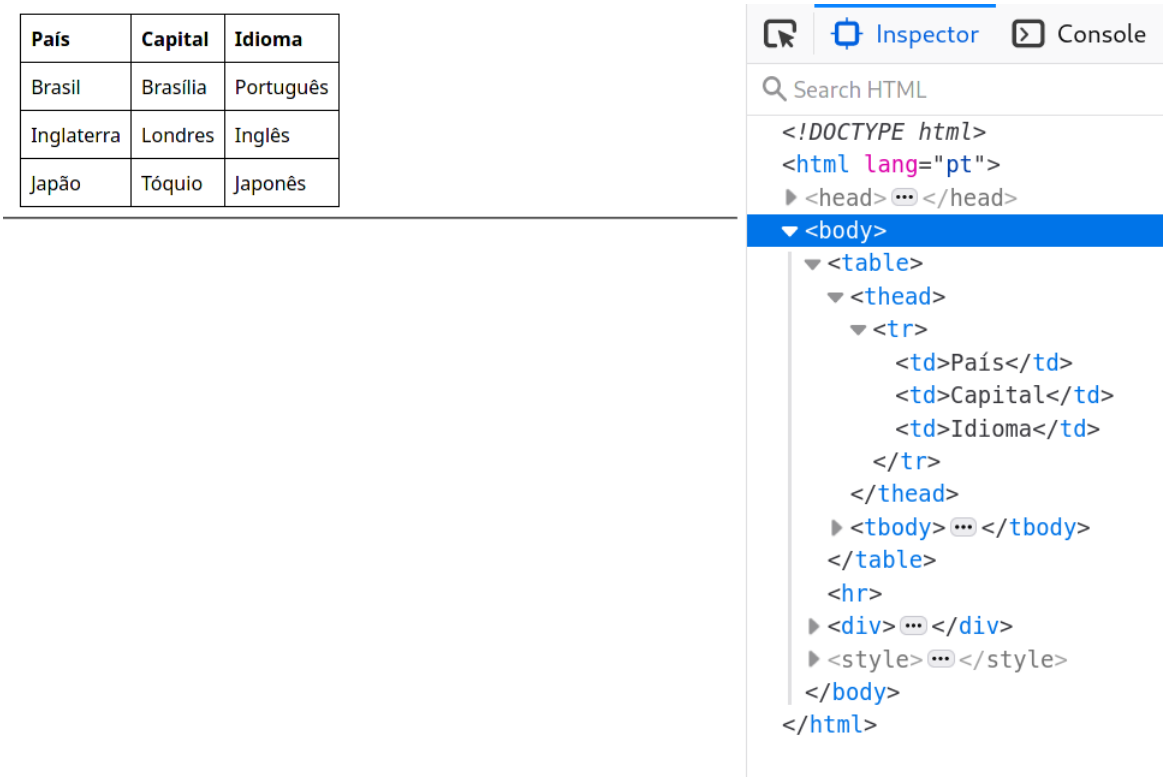
Utilizou-se mais uma vez do padrão de exemplos visualmente idênticos, mas estruturalmente diferentes. A Figura 41 mostra o exemplo com problemas de acessibilidade. Nele, todos os elementos da tabela são declarados com a *tag* `td`, não existindo distinção para os cabeçalhos. Dessa forma, tecnologias assistivas não conseguem anunciar corretamente a descrição de um elemento na tabela.

A Figura 42 mostra o exemplo que faz uso correto das *tags* de cabeçalho `th`. A Figura 43 mostra ainda outra evolução desse exemplo, utilizando o atributo `scope` para identificar se um cabeçalho está relacionado aos elementos de uma linha ou de uma coluna.

5.3.1.8 Componentes Interativos

Os exemplos dessa seção são, no geral, mais complexos que os das seções anteriores, e exigem um maior uso de JavaScript e de ARIA, por representarem funcionalidades que fogem ao básico do HTML. Nos três tópicos, optou-se por apresentar diretamente o exemplo do componente, e em seguida destacar as particularidades da implementação. Embora representem componentes distintos, com contextos específicos, os três exemplos apresentam desafios em comum: mais especificamente, garantir a ordem lógica de navegação entre os elementos, muitas vezes através de controle de foco customizado através

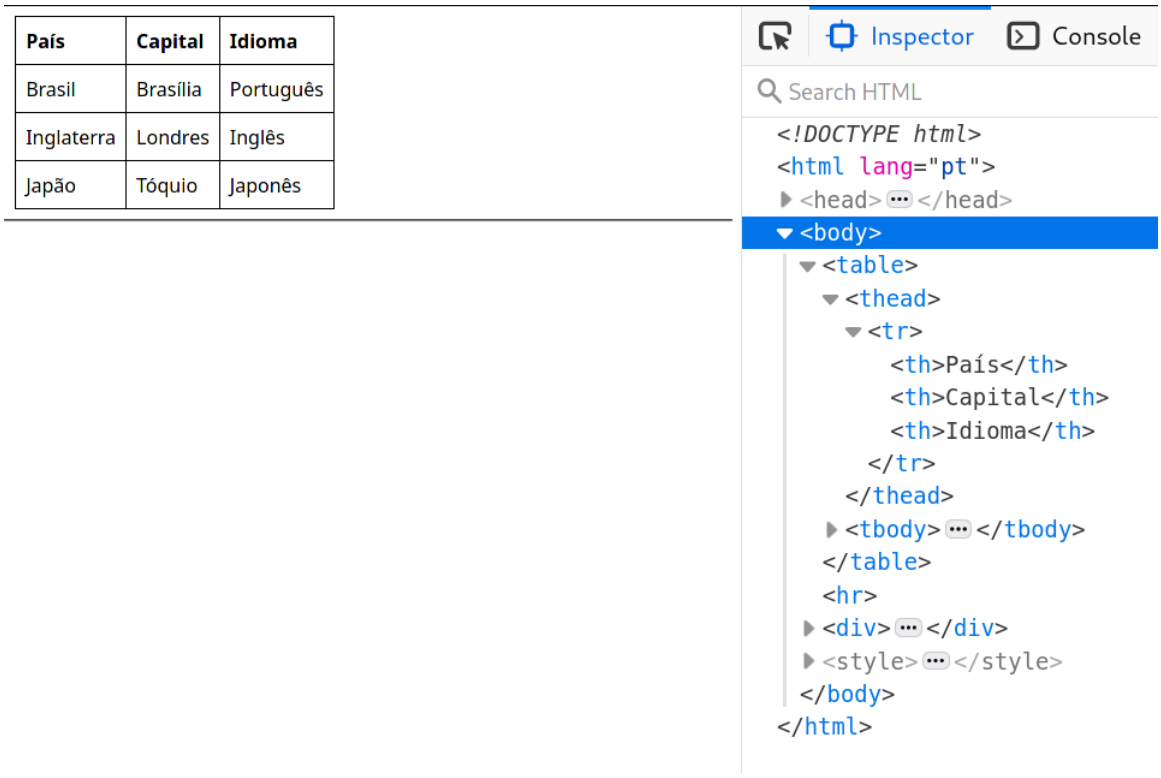
Figura 41 – Exemplo de tabela sem marcação de cabeçalhos.



País	Capital	Idioma
Brasil	Brasília	Português
Inglaterra	Londres	Inglês
Japão	Tóquio	Japonês

Fonte: Autor.

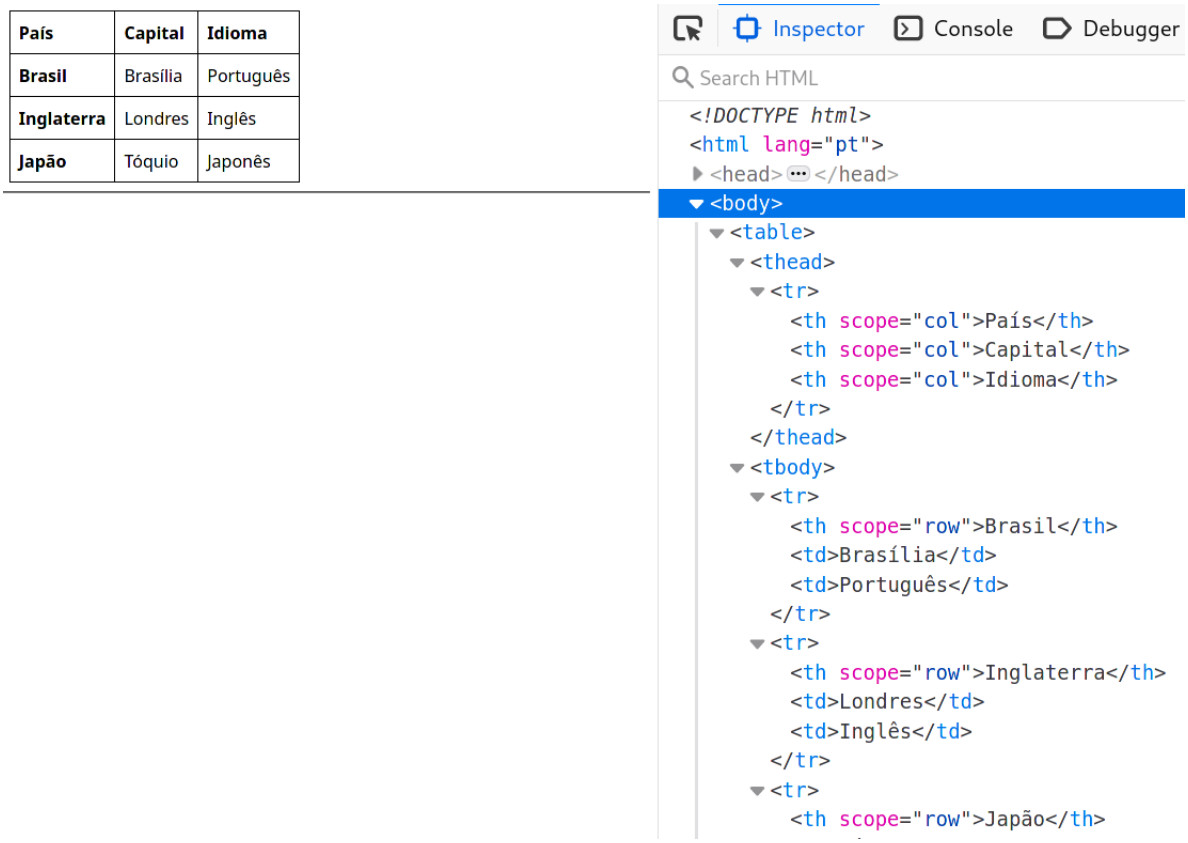
Figura 42 – Exemplo de tabela com marcação de cabeçalhos.



País	Capital	Idioma
Brasil	Brasília	Português
Inglaterra	Londres	Inglês
Japão	Tóquio	Japonês

Fonte: Autor.

Figura 43 – Exemplo de tabela com marcação de cabeçalhos e uso de *scopes*.



País	Capital	Idioma
Brasil	Brasília	Português
Inglaterra	Londres	Inglês
Japão	Tóquio	Japonês

Fonte: Autor.

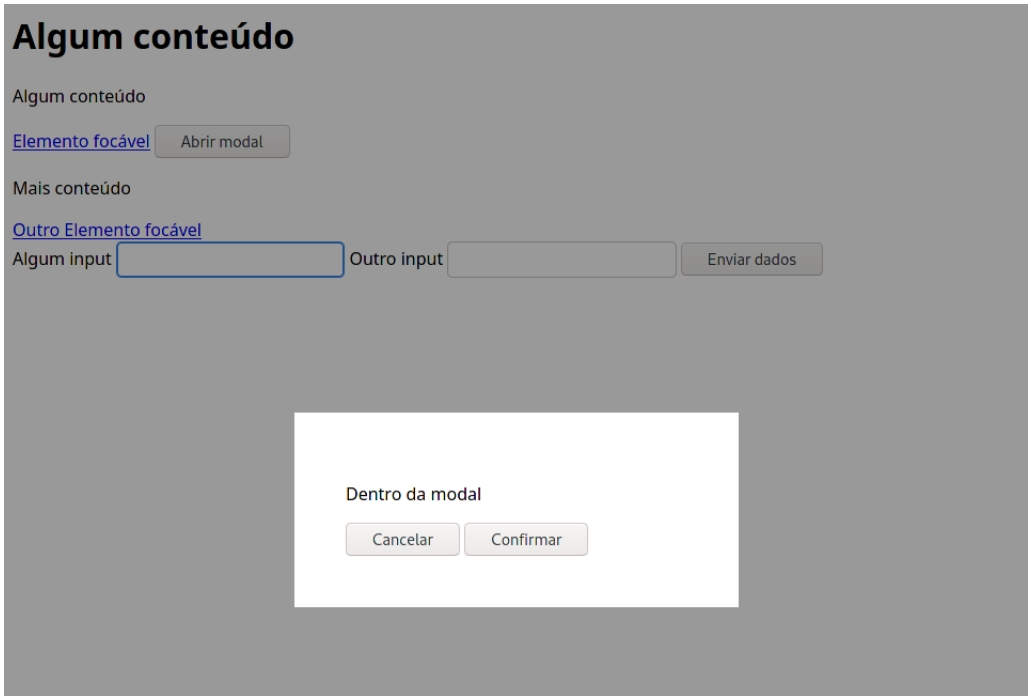
de JavaScript, e garantir que os diferentes estados dos componentes sejam perceptíveis e navegáveis.

O primeiro caso a ser apresentado é o de implementação de modais. A Figura 44 demonstra a versão com problemas de acessibilidade do exemplo. Nela, o foco do teclado não é redirecionado para dentro da modal, e o usuário pode ter que navegar por toda a estrutura restante de *links* da página até alcançar os controles da modal, ou nem mesmo estar ciente da existência da modal.

Já a Figura 45 mostra o exemplo orientado à acessibilidade, onde o foco do teclado é trazido diretamente para dentro da modal, e bloqueando a navegação aos elementos do conteúdo principal da página até que a modal seja fechada. É levantada ainda uma questão geral de usabilidade: o foco inicial do teclado deve ser levado à ação não-destrutiva do contexto (como um botão de cancelar), para evitar que usuários cometam uma ação irreversível por acidente.

As Figuras 46 e 47 destacam o código em JavaScript relevante para que a modal apresente o comportamento desejado. A Figura 46 mostra como manter o foco do teclado preso dentro dos componentes da modal, enquanto a mesma não for fechada, voltando para o primeiro elemento ao apertar `Tab` a partir do último controle, ou indo ao último elemento ao apertar `Shift Tab` a partir do primeiro. Já a Figura 47 mostra como salvar

Figura 44 – Exemplo de modal com problemas de acessibilidade.



Fonte: Autor.

Figura 45 – Exemplo de modal com navegação acessível.

Fonte: Autor.

o estado do elemento que iniciou a ação de abrir a modal, para que o foco do teclado seja retornado a ele ao se fechar o diálogo. Dessa forma, o fluxo de navegação mantém-se consistente.

Por fim, a Figura 48 mostra parte do HTML do componente da modal, com des-

Figura 46 – Código para controle de foco dentro da modal.

```
const TAB = 9;
const primeiroControleModal = document.getElementById('modal-first-el');
const ultimoControleModal = document.getElementById('modal-last-el');

primeiroControleModal.onkeydown = (e) => {
  if (e.shiftKey && e.keyCode === TAB) {
    e.preventDefault();
    ultimoControleModal.focus();
  }
}

ultimoControleModal.onkeydown = (e) => {
  if (!e.shiftKey && e.keyCode === TAB) {
    e.preventDefault();
    primeiroControleModal.focus();
  }
}
```

Fonte: Autor.

Figura 47 – Código para retorno do foco ao fechar a modal.

```
let focoAnterior;

botaoAbreModal.onclick = (e) => {
  focoAnterior = document.activeElement;
  // ...
}

botaoFechaModal.onclick = (e) => {
  // ...
  focoAnterior.focus();
}
```

Fonte: Autor.

taque para os atributos da ARIA utilizados. O uso de `role=dialog` e `aria-modal=true` permitem que tecnologias assistivas anunciem corretamente a existência de um elemento que se sobrepõe ao conteúdo principal da tela, e que é especificamente uma modal. Além disso, utiliza-se do atributo `aria-describedby` para que o propósito da modal seja anunciado assim que a mesma seja aberta.

O artigo sobre *dropdowns* apresenta dois exemplos distintos, com diferenças consideráveis de implementação. O primeiro, visto nas Figuras 49 e 50, mostra uma interface

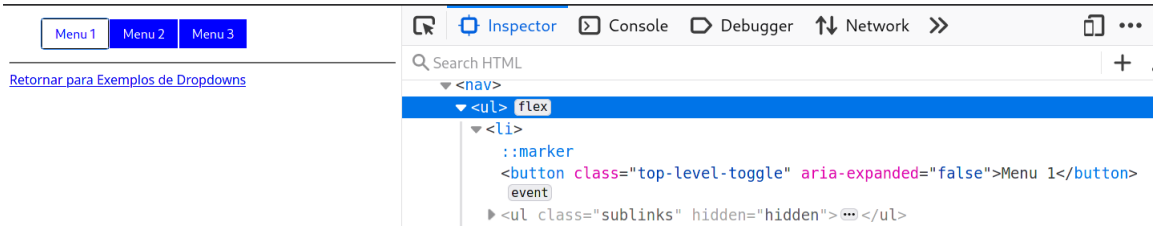
Figura 48 – Estrutura HTML da modal.

```
<div id="modal-container" role="dialog" aria-modal="true" aria-
labelledby="modal__confirm-label"> event
  <div class="overlay"></div>
  <div class="modal">
    <h1 id="modal__confirm-label">Confirmar Ação</h1>
    <p>Dentro da modal</p>
    <button id="modal__cancel-button">Cancelar</button> event
    whitespace
    <button id="modal__confirm-button">Confirmar</button> event
  </div>
</div>
```

Fonte: Autor.

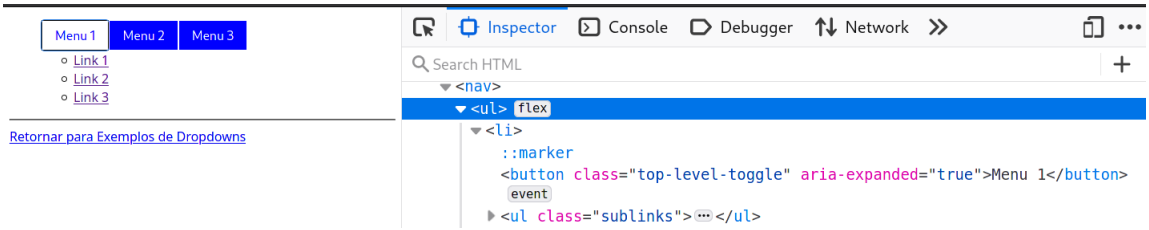
mais simples, utilizando *dropdowns* para um *menu* de navegação.

Figura 49 – *Dropdown* de navegação, colapsado.



Fonte: Autor.

Figura 50 – *Dropdown* de navegação, expandido.



Fonte: Autor.

O ponto de destaque desse exemplo é relacionado ao controle de visibilidade dos elementos. Para isso, foram utilizados os atributos `hidden`, e `aria-expanded` que, respectivamente, permitem esconder um elemento tanto visualmente quanto para tecnologias assistivas, e informar para tecnologias assistivas se o conteúdo alternável está ou não expandido. A lógica para alternar o estado desses atributos é realizada via JavaScript, como visto na Figura 51.

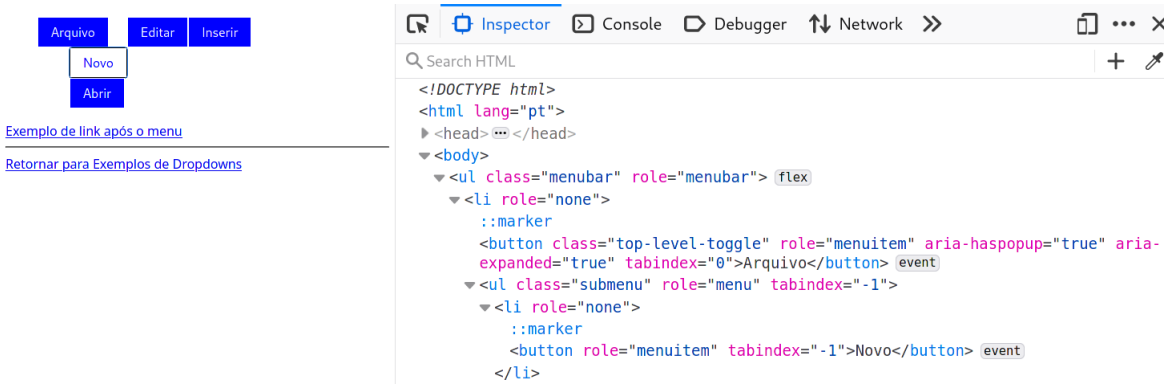
Já o segundo exemplo, exibido na Figura 52, apresenta características mais complexas, referente a um *menu* de configurações de uma página. Essa implementação faz uso de `ARIA roles` mais específicas ao contexto de um menu: `menubar`, aplicada ao container externo do componente, `menu`, aplicada aos botões "Arquivo", "Inserir"e "Editar", e `menuitem`, aplicada aos controles internos de cada *menu*.

Figura 51 – Lógica de controle do *dropdown*.

```
toggleButton.onclick = (e) => {
  const links = toggleButton.parentElement.querySelector('.sublinks');
  if (buttonWasActive(toggleButton)) {
    toggleButton.setAttribute('aria-expanded', 'false');
    links.setAttribute('hidden', 'hidden');
  } else {
    links.removeAttribute('hidden');
    toggleButton.setAttribute('aria-expanded', 'true');
  }
  toggleButtonsState[i] = !toggleButtonsState[i];
}
```

Fonte: Autor.

Figura 52 – Exemplo de *menu dropdown*.



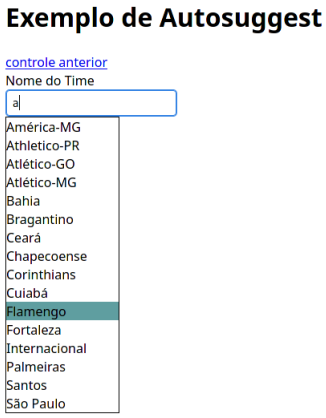
Fonte: Autor.

O ponto que exige maior atenção, no entanto, é relacionado ao controle de navegação pelo teclado. Em componentes nesse formato, especialmente para *menus* com muitos elementos, é sugerido que os controles sejam retirados do fluxo de navegação padrão da página, alterando seu atributo `tabindex` para -1. Dessa forma, os *links* e botões que constitutem o *menu* não são acessíveis através da tecla `Tab`, à exceção do primeiro elemento no menu (exibido aqui explicitamente com `tabindex=0`. Isso é realizado para que os usuários não tenham que navegar por toda a lista de itens do *menu* até alcançar outros controles na página. Para que o *menu* ainda seja acessível, é implementada lógica adicional para navegar pelo componente através das setas do teclado. As setas para a esquerda e para a direita alternam entre os botões externos, e as setas para cima e para baixo permitem navegar pelos itens de cada *menu*. Esse é um dos raros casos em que se recomenda mexer no atributo `tabindex`.

O último exemplo implementado foi relacionado ao uso de *inputs* com sugestões automáticas. A interface desenvolvida pode ser vista na Figura 53, e o código relevante é apresentado na Figura 54. Mais uma vez, são utilizados atributos da ARIA específicos

para esse contexto. O formato desse componente é conhecido como `combobox`, que também é o nome da `ARIA role` do elemento que contém o `input`. A lista de sugestões automáticas é marcada com um outro papel complementar, de `listbox`. Os atributos `aria-owns`, `aria-controls` e `aria-haspopup` auxiliam tecnologias assistivas a anunciarem corretamente a relação existente entre o *input* e a lista de sugestões, garantindo que os elementos serão anunciados em uma ordem coerente. O atributo `aria-autocomplete="list"` explicita ainda que existe a funcionalidade de *autocomplete* para aquele controle.

Figura 53 – Exemplo de *input* com sugestões automáticas.



Fonte: Autor.

Este é outro exemplo com lógica customizada para navegação, utilizando as setas do teclado para acessar elementos `li`, que por padrão não são focáveis. Para que esse comportamento esteja correto, é utilizado ainda outro atributo ARIA, o `aria-activedescendant`. Esse atributo indica qual subitem do componente está visualmente sob foco do teclado, enquanto o foco real da DOM permanece no `input`. O valor desse atributo é alterado dinamicamente via JavaScript, sempre que o usuário navega pela lista de sugestões.

5.4 Trabalhos relacionados

Durante a realização da pesquisa, foram encontrados alguns projetos com propósitos similares ao deste trabalho. No geral, são *websites* criados por organizações especializadas em acessibilidade ou indivíduos interessados no tema. Muitos deles tem um foco específico nas normas de conformidade do WCAG, através de *checklists* ou guias de consulta rápida, como a *checklist* do A11Y Project [1] ou o resumo traduzido das diretrizes do WCAG, do Guia WCAG [2]. Um deles, no entanto, apresenta caráter informativo de cunho semelhante à proposta desta pesquisa. O mesmo será detalhado a seguir. Dentre o material obtido a partir de consulta em bases científicas, a maioria se concentra no ensino de acessibilidade e assuntos relacionados em programas de ensino superior (LUDI

[1] <https://www.a11yproject.com/checklist/>. Acessado pela última vez em 08/05/2021
[2] <https://guia-wcag.com/>. Acessado pela última vez em 08/05/2021

Figura 54 – HTML referente ao *input* com sugestões automáticas.

```
<div>
    <label id="input-label" for="team-input">Nome do Time</label>
</div>
<div class="input-container" id="input-container">
  <div
    class="combobox"
    role="combobox"
    aria-haspopup="listbox"
    aria-owns="suggestions"
    aria-expanded="false">
    <input
      id="team-input"
      type="text"
      aria-autocomplete="list"
      aria-controls="suggestions"
      aria-active-descendant="item-0">
  </div>
  <ul aria-labelledby="input-label" hidden id="suggestions" role="listbox">
    <li role="option" id="item-0">...</li>
    <li role="option" id="item-1">...</li>
  </ul>
</div>
```

Fonte: Autor.

et al., 2018; GAY; DJAFAROVA; ZEFI, 2017; EL-GLALY, 2020; COHEN et al., 2005). Destaca-se ainda o projeto WebAIM [3] (*Web Accessibility in Mind*), mantido por um grupo de pesquisa da Universidade de Utah, e que apresenta materiais informativos e resultados de pesquisas com desenvolvedores e usuários de tecnologias assistivas.

5.4.1 *Accessibility Developer Guide*

O *Accessibility Developer Guide* (ADG)[4] é uma iniciativa da fundação suíça *Access For All*, formada por especialistas em acessibilidade e desenvolvedores web, incluindo pessoas com deficiência. Conta ainda com o apoio técnico de um grupo de agências de desenvolvimento web.

O *site* é disponibilizado em inglês, e tem seu conteúdo dividido em três seções principais: *Setup*, Conhecimento e Exemplos. As duas primeiras apresentam informações teóricas sobre ferramentas e técnicas, enquanto a última seção apresenta uma coletânea de exemplos de soluções de acessibilidade, divididas em categorias. O guia ainda possui uma seção introdutória e outra contendo informações de contribuição. O site é *open source*,

[3] <https://webaim.org>. Acessado pela última vez em 08/05/2021
[4] https://www.accessibility-developer-guide.com/. Acessado pela última vez em 08/05/2021

licenciado sob a Licença MIT [5].

5.4.1.1 Comparação com o trabalho atual

Apesar das semelhanças em objetivos e estrutura, os dois trabalhos apresentam diferenças em alguns pontos. Apesar de ambos possuírem catálogos de exemplos, suas implementações e conteúdos não são idênticos, ainda que abordem tópicos em comum. Entende-se que, dada a reconhecida escassez de recursos de acessibilidade disponíveis, a adição de diferentes práticas e pontos de vista é benéfica.

O apoio *online* desta pesquisa procura ainda expor maior variedade de métodos para a avaliação de acessibilidade, em complemento ao uso de ferramentas automáticas. Por fim, destaca-se que todo o conteúdo elaborado para a presente solução é disponibilizado em português. Nota-se que a grande maioria dos recursos disponíveis atualmente é disponível exclusivamente em inglês, ou com traduções parciais. Do corpo oficial de documentos do WCAG, existe tradução apenas para a norma técnica em si. Os demais documentos de apoio são disponibilizados apenas em inglês. Dentro desse contexto, considera-se um diferencial importante para a comunidade brasileira a existência de um apoio em língua portuguesa.

5.5 Considerações finais do capítulo

Este capítulo detalhou as atividades realizadas para o desenvolvimento do projeto, refinando os principais requisitos elicitados em relação ao conteúdo e apresentação da solução.

Foram apresentadas tanto atividades desenvolvidas previamente à implementação concreta do apoio, como protótipos e outras provas de conceito, bem como foram destacados os principais aspectos do *website* desenvolvido, com um foco especial para o catálogo de exemplos de interfaces acessíveis e inacessíveis.

Por fim, foram apresentados alguns trabalhos relacionados. Um deles foi discutido em maiores detalhes, expondo semelhanças e diferenças com o trabalho atual. Entende-se que ainda há carência de recursos informativos sobre acessibilidade, especialmente ao se considerar materiais disponíveis em português.

[5] https://opensource.org/licenses/MIT. Acessado pela última vez em 08/05/2021

6 Resultados

Este capítulo descreve e comenta as atividades de análise de resultados realizadas sobre o produto implementado e visto no Capítulo 5, bem como registra as ações de melhoria identificadas e aplicadas. Os procedimentos de coleta de dados e análise de resultados foram feitos seguindo a metodologia de pesquisa-ação acordada no Capítulo 4. Foram implantados três ciclos, e para cada um deles será detalhado o mecanismo de coleta de dados, a interpretação desses dados, os pontos de melhoria identificados e as ações praticadas a partir desses pontos.

O primeiro ciclo, descrito na Seção 6.1, foi realizado de forma introspectiva, a partir de pontos identificados pelo próprio autor durante o desenvolvimento. O segundo ciclo, visto na Seção 6.2, mapeou os tópicos tratados no apoio com as recomendações técnicas que guiaram sua escrita. Por fim, o terceiro ciclo trouxe *feedback* externo, a partir das opiniões e sentimentos do público alvo do apoio, obtidas através de um questionário.

6.1 Ciclo 1 - Pontos identificados durante o desenvolvimento

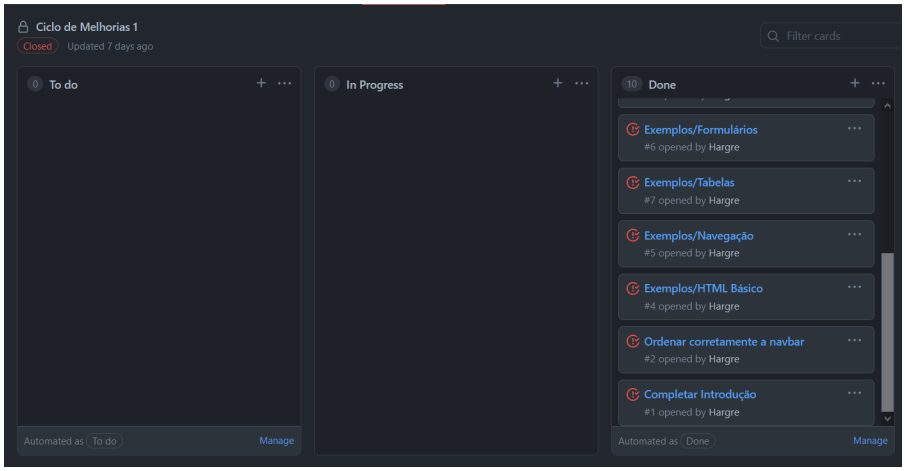
Esse ciclo correspondeu ao registro de pontos de melhoria variados, encontrados de forma espontânea durante o desenvolvimento do projeto. Alguns desses pontos corresponderam a refinamentos pontuais de texto e organização dos artigos, enquanto outros implicaram na criação de novas seções do apoio e adições mais substanciais a um exemplo ou seção informativa. Essas atividades foram registradas em um *board* Kanban no próprio Github, ilustrado na Figura 55. Nessa seção serão destacadas algumas das melhorias mais significativas. A relação completa das ações pode ser vista no próprio *board*[1].

6.1.1 Exemplos com áudio

No artigo de exemplos de HTML Básico, julgou-se apropriado adicionar trechos em áudio para facilitar a compreensão dos exemplos relacionados ao atributo `lang`, que define o idioma da página. Para isso, foram realizadas duas gravações de uso do NVDA, acessando a interface de exemplo da seção, tanto em sua versão correta quanto incorreta. Essa ação pôde ser validada posteriormente, no terceiro ciclo de análise de resultados, quando uma das participantes da pesquisa elogiou de forma específica o exemplo com áudio.

[1] Disponível em: <https://github.com/Hargre/a11y-catalog/projects/1>

Figura 55 – *Board* de registro do primeiro Ciclo de Melhorias.



Fonte: Autor.

6.1.2 Guia rápido

Durante o desenvolvimento do apoio, notou-se a necessidade de criar uma forma de acesso mais resumida às suas informações, especialmente no que dizia respeito ao catálogo de exemplos, devido à grande quantidade de informação sendo transmitida. Para aumentar as possibilidades de uso do apoio por um número variado de usuários, optou-se por criar uma página [2] contendo uma versão sucinta dos conceitos expostos nos exemplos detalhados, servindo assim como um guia rápido. Dessa forma, o formato de uso do *website* é expandido, com os usuários podendo consultar o guia rápido para ter uma visão geral de vários tópicos de uma forma mais simples, podendo sempre recorrer aos exemplos mais completos quando necessário. Essa melhoria também pôde ser validada posteriormente, como detalhado na Seção 6.3. A página de guia rápido foi considerada unanimemente pelo público alvo como um dos pontos mais importantes do apoio.

6.2 Ciclo 2 - Inspeção do conteúdo do apoio quanto às diretrizes

Para este ciclo, realizou-se uma inspeção do conteúdo tratado no apoio, de acordo com as bases técnicas utilizadas para sua elaboração. Essa ação foi realizada a partir da forma tabular das operacionalizações do catálogo referenciado na Seção 5.1.1. Cada uma dessas operacionalizações se refere a uma ou mais técnicas apresentadas pelo W3C para cumprir com as diretrizes do WCAG. O resultado completo da inspeção, com uma versão adaptada dessa tabela, pode ser visto no Apêndice B. Aqui, serão apresentados os critérios utilizados para a inspeção, bem como discutidos seus principais resultados.

Inicialmente, foram realizados alguns ajustes no escopo da tabela, para determinar quais operacionalizações seriam de fato consideradas na verificação. Três tipos de técnicas

[2] Disponível em <https://tender-leavitt-ddf88b.netlify.app/pages/quick_guide/>. Acessado pela última vez em 10/05/2021.

foram desconsideradas, sendo elas:

- Técnicas relacionadas a tecnologias obsoletas, como Flash e Silverlight;
- Técnicas relacionadas a formatos externos à Web, como arquivos PDF, e
- Técnicas relacionadas a diretrizes de nível AAA do WCAG. Nesse primeiro momento, optou-se por restringir o escopo do apoio até o nível AA, por conter regras com escopo mais geral e mais facilmente aplicáveis para contextos variados.

A Figura 56 mostra os valores obtidos. Das 85 operacionalizações consideradas, identificou-se que 47 técnicas foram abordadas na versão atual do apoio.

Figura 56 – Resultados da inspeção.

Status	**Quantidade**
Não se aplica (obsoleto)	5
Não se aplica (formato)	7
Não se aplica (AAA)	35
Não implementado	38
Implementado	47

Fonte: Autor.

Ao investigar esses valores, surgiram duas considerações importantes. Pôde-se confirmar que uma análise puramente quantitativa é insuficiente para tratar de um critério como acessibilidade, pois o esforço necessário para compreender e implementar as operacionalizações foi amplamente desigual entre cada operacionalização. Algumas técnicas se encaixam em um padrão binário: por exemplo, "evitar alinhamento justificado"ou "usar espaçamento mínimo entre linhas de 1.5"são ações facilmente descritas, aplicadas e verificadas, sem a existência de meio-termos; outras, como "Fornecer alternativas ao mouse", "Garantir que o usuário não fique preso em algum conteúdo"ou "Utilizar elementos de marcação com semânticas apropriadas"exigem interpretações e implementações distintas de acordo com o contexto e com os elementos existentes em uma determinada página. Todas as cinco técnicas mencionadas foram contempladas no apoio, mas o esforço empenhado para as três últimas foi muito superior, sendo tópicos abordados em praticamente todos os exemplos. Dessa forma, avaliar unicamente a quantidade de operacionalizações implementadas ou não implementadas não é um indicador completo.

Apesar disso, realizar esse levantamento permtiu identificar pontos fortes e fracos do apoio. Tópicos relativos à variedade de formas de apresentação, navegação e interação com o conteúdo de uma página foram bem contemplados. Por outro lado, houve carência de informações e exemplos quanto ao uso de mídias como vídeos e áudios, bem como

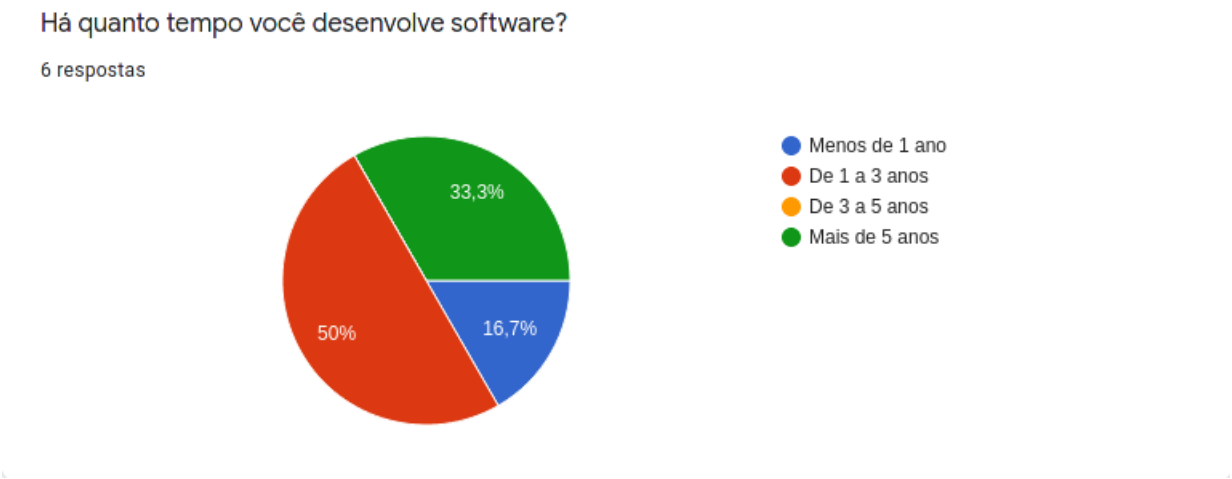
de controle de elementos com tempo. Esses temas serão objeto de trabalhos futuros, em evoluções do apoio.

6.3 Ciclo 3 - *Feedback* com o público alvo

Por fim, foi realizado um ciclo com participação do público alvo do apoio, isto é, desenvolvedores de software. Obteve-se uma amostra de seis pessoas, com experiência em desenvolvimento e conhecimentos sobre acessibilidade em níveis variados. Cada um desses participantes acessou e leu uma mesma versão do apoio, para em sequência responder a um questionário. O objetivo principal desse questionário foi de obter *feedback* sobre o uso do apoio, bem como sugestões de melhorias. Não é possível extrair conclusões mais generalistas sobre o tema a partir de suas respostas, devido ao pequeno tamanho da amostra.

O questionário foi dividido em quatro seções. A primeira buscou obter informações do *background* de cada participante em relação à experiência com desenvolvimento de software no geral, e quanto a conhecimentos e percepções sobre acessibilidade. Seus resultados são exibidos nas Figuras 57 e 58. As escalas utilizadas para as perguntas são exibidas na Figura 59.

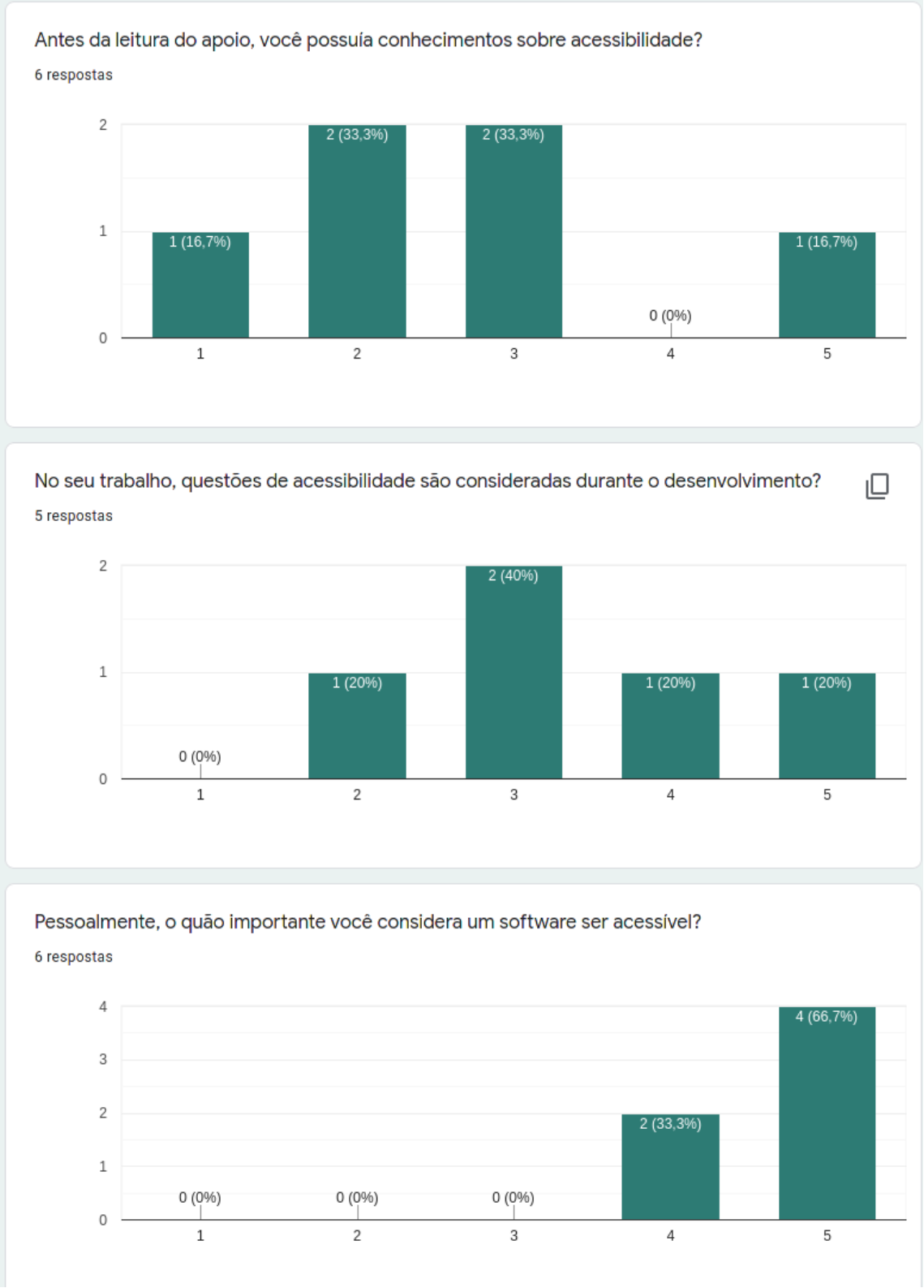
Figura 57 – Respostas à pergunta "Há quanto tempo você desenvolve software?"



Fonte: Autor.

Quanto à experiência com desenvolvimento no geral, metade dos participantes respondeu desenvolver software de 1 a 3 anos. Dos demais, um desenvolve a menos de 1 ano e outros dois desenvolvem a mais de 5. Quanto aos conhecimentos e percepções sobre acessibilidade, todos os participantes afirmaram considerar a acessibilidade de um software algo muito ou extremamente importante. Apesar disso, apenas um informou possuir maiores conhecimentos sobre o tema.

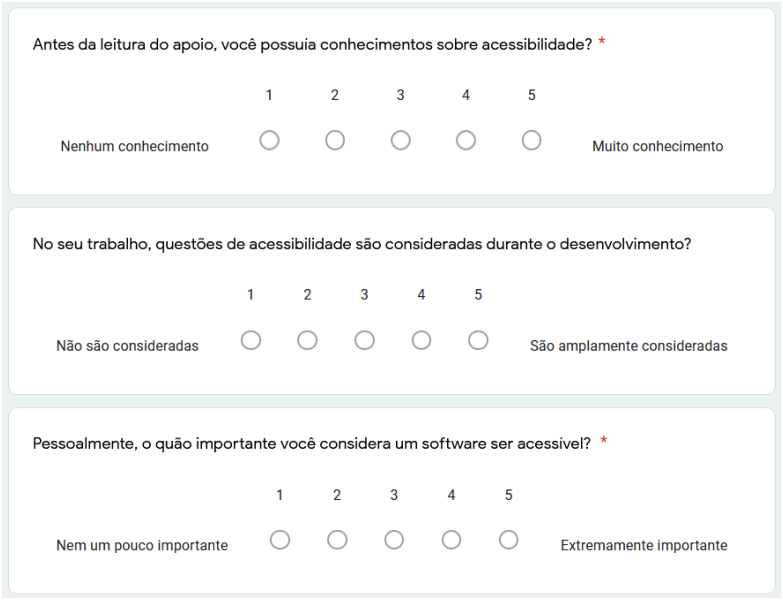
Figura 58 – Respostas a perguntas sobre conhecimentos de acessibilidade.



Fonte: Autor.

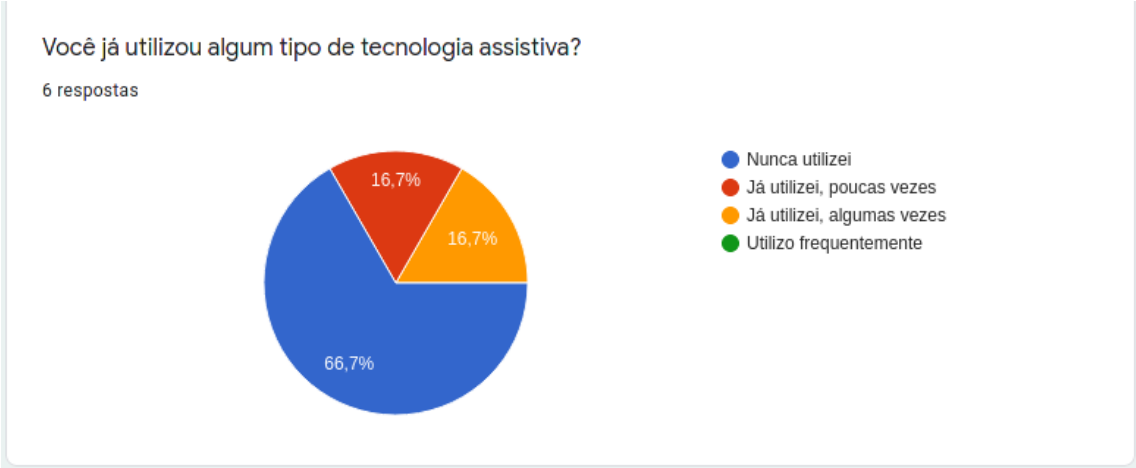
A segunda seção investigou o grau de familiaridade dos participantes com o uso de tecnologias relacionadas à acessibilidade, sejam elas tecnologias assistivas ou ferramentas de avaliação. Seus resultados são exibidos nas Figuras 60 e 61.

Figura 59 – Escalas utilizadas nas perguntas sobre conhecimentos de acessibilidade.



Fonte: Autor.

Figura 60 – Respostas à pergunta "Você já utilizou algum tipo de tecnologia assistiva?"

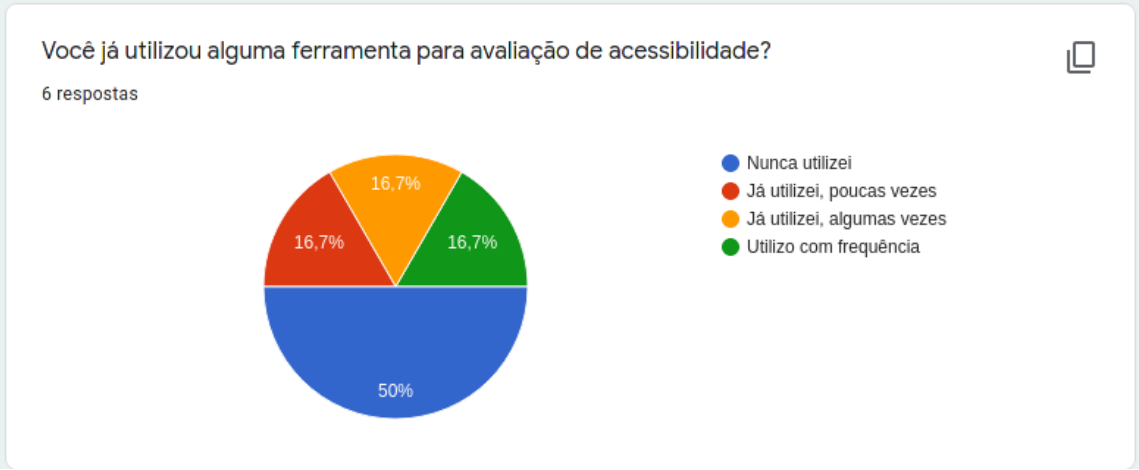


Fonte: Autor.

Quanto ao uso de tecnologias assistivas, a maior parte dos participantes relatou nunca ter utilizado alguma, em qualquer contexto. Dos dois participantes que responderam já ter utilizado em algum momento, um informou ter utilizado a aplicação VLibras para testes durante o desenvolvimento, enquanto outra participante utilizou o ampliador de tela por necessidade, após realizar uma cirurgia ocular. Embora não tenha considerado como tecnologia assistiva, outra participante relatou utilizar com frequência a funcionalidade de *zoom* do navegador, devido a um alto grau de miopia.

Quanto ao uso de ferramentas de avaliação, metade dos participantes relatou nunca ter utilizado alguma. Dos demais, apenas um disse utilizar com frequência. As ferramentas relatadas foram Google Lightouse e WAVE.

Figura 61 – Respostas à pergunta "Você já utilizou alguma ferramenta para avaliação de acessibilidade?".



Fonte: Autor.

A terceira seção buscou averiguar o nível de contato que os participantes tinham com problemas relacionados à acessibilidade, seja profissionalmente ou em produtos de software utilizados de forma pessoal. Suas respostas estão disponíveis na Figura 62.

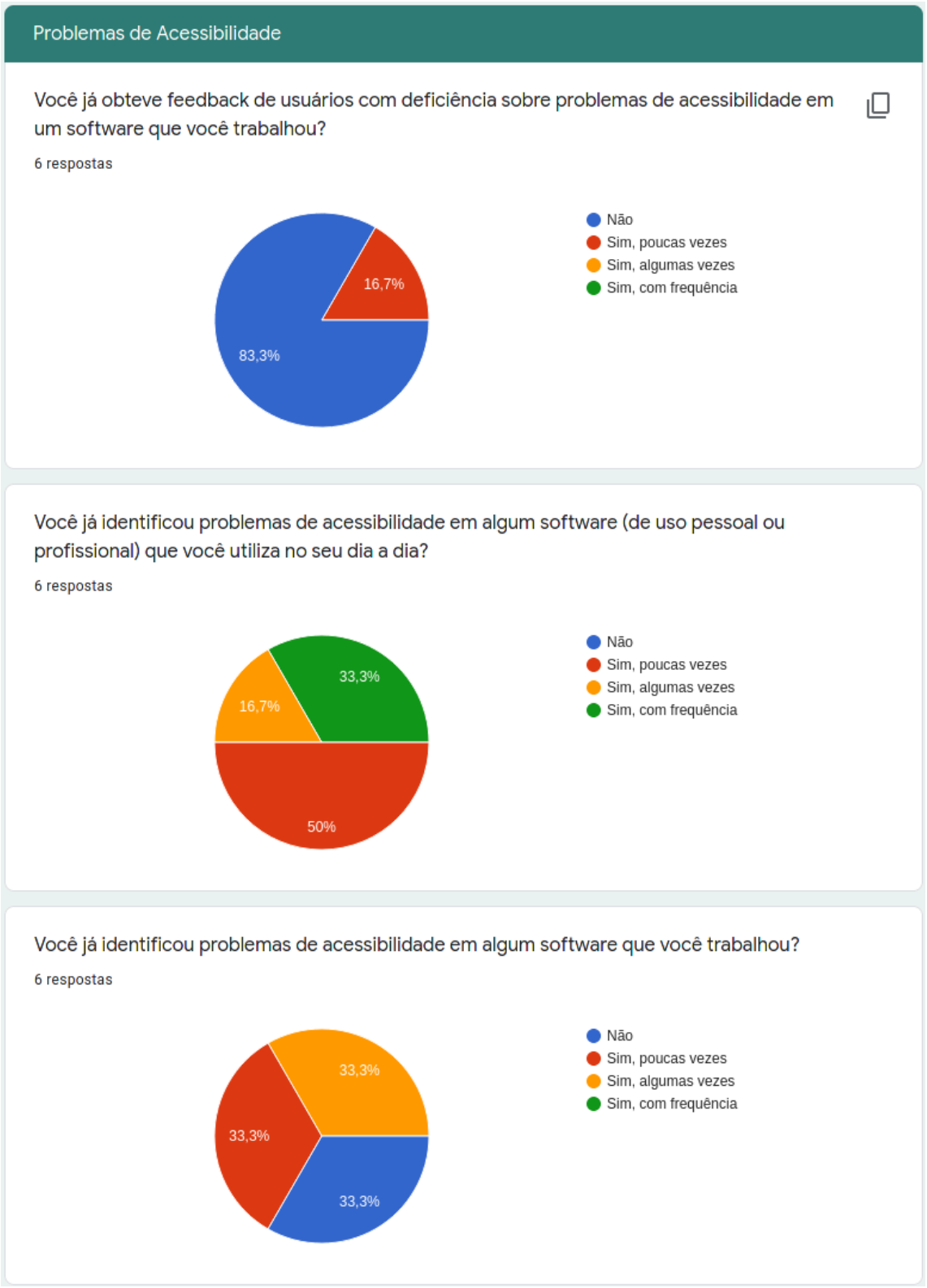
Todos os participantes afirmaram já ter encontrado problemas de acessibilidade em algum dos contextos descritos, em especial em produtos de software utilizados no dia a dia. No contexto profissional, a maioria diz já ter encontrado algum tipo de problema. Apenas um participante relatou já ter obtido *feedback* de usuários com deficiência. Os problemas informados vão de encontro a vários dos assuntos tratados pelo apoio, como baixo contraste, dificuldades de redimensionar a interface e dificuldades de navegação. A lista completa de respostas pode ser vista na Figura 63.

Por fim, buscou-se obter *feedback* mais específico sobre o apoio. Cinco dos seis participantes responderam que após a leitura do apoio conseguiriam identificar problemas de acessibilidade que não conheciam anteriormente. A única resposta negativa foi do participante que já possuía maiores conhecimentos sobre acessibilidade, que, apesar disso, teve uma percepção extremamente positiva do apoio, comentando que

> [...] por mais que tenhamos os conteúdos do w3c ou algo do tipo, eles nem sempre são resumidos de uma forma simples como você fez. Além de você ter agregado bastante conteúdo focado inteiramente em acessibilidade em uma página só. Se eu tivesse a oportunidade de ter esse conteúdo quando comecei a atuar profissionalmente no desenvolvimento web, me pouparia muuuuitos meses de pesquisa e de softwares não tão acessíveis. (trecho de resposta de participante da pesquisa)

Outros participantes também descreveram positivamente a experiência de uso do apoio, como visto na Figura 64.

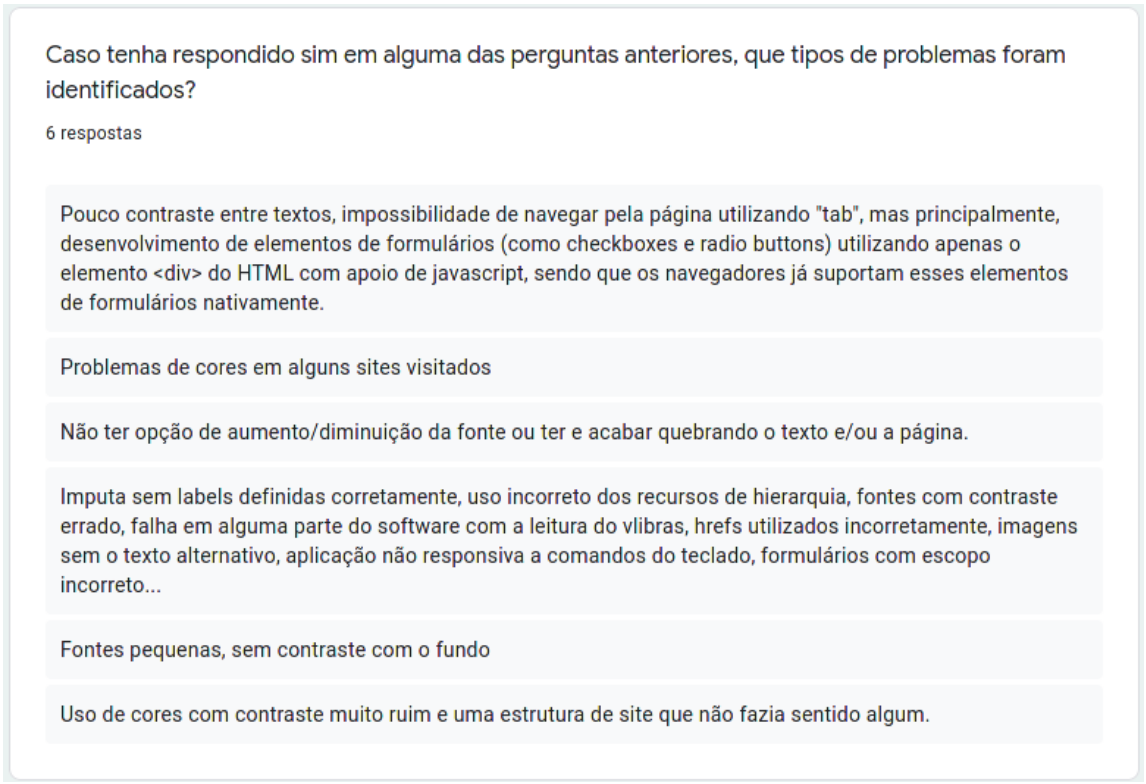
Figura 62 – Respostas a perguntas sobre identificação de problemas de acessibilidade.



Fonte: Autor.

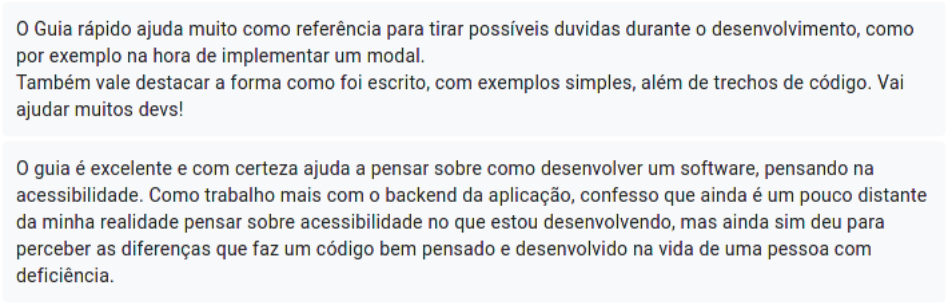
Quanto à importância específica de cada seção do guia, todos os participantes marcaram as seções de "Exemplos detalhados"e "Guia Rápido", como visto na Figura 65. Essa informação valida a ação implementada no primeiro ciclo, onde foi identificada a

Figura 63 – Respostas a perguntas sobre identificação de problemas de acessibilidade.



Fonte: Autor.

Figura 64 – Feedback aberto sobre o apoio.



Fonte: Autor.

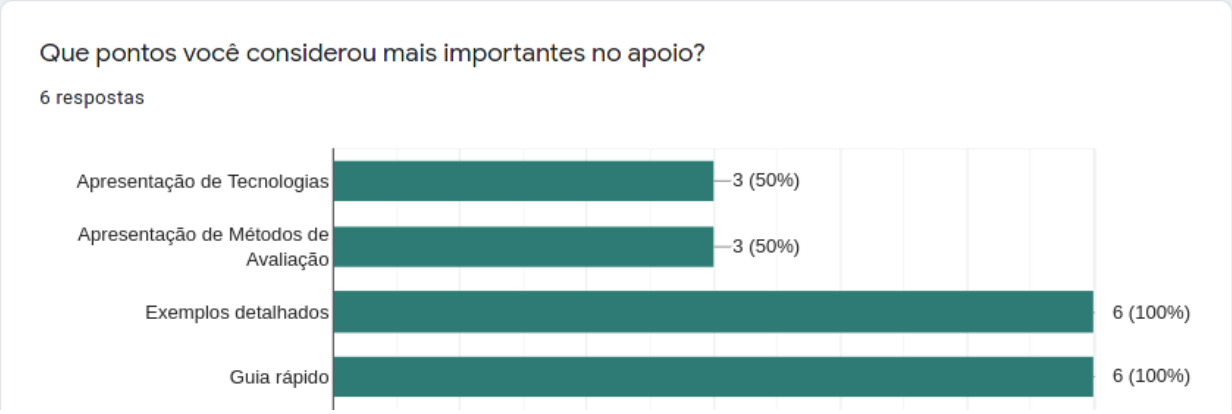
necessidade de criação da versão resumida dos exemplos.

Além disso, foram apresentadas algumas sugestões e dúvidas, que serviram de insumo para o mapeamento e elaboração de novas melhorias ao projeto. Esses pontos foram novamente rastreados através de um *board* no Github, da mesma forma que no primeiro ciclo. Uma descrição mais detalhada de cada ação é apresentada a seguir.

6.3.1 Ação: Melhorar descrição do projeto

- **Feedback relacionado (sic)**: "Talvez o título e o texto introdutório da “home” deveriam descrever melhor o conteúdo do site. Por exemplo, eu por ler “apoio para

Figura 65 – Respostas à pergunta "Que pontos você considerou mais importantes no apoio?"



Fonte: Autor.

a acessibilidade da web" e o texto da home ainda não tinha deixado claro do que se tratava. Apenas quando eu dei uma olhada em todo o conteúdo que vi que é um "manual" de boas práticas para o desenvolvimento de aplicações web acessíveis. Talvez se colocasse um parágrafo explicando isso ou deixando isso mais transparente no título, transmita melhor o conteúdo do site."

- **Detalhes da ação**: Optou-se por aperfeiçoar a descrição do projeto em sua página inicial, de forma a transmitir melhor o propósito do apoio.

- **Resultados da ação**: As Figuras 66 e 67 mostram a página inicial antes e após a implementação da melhoria.

6.3.2 Ação: Melhorar explicação sobre user-scalable

- **Feedback relacionado (sic)**: "eu fiquei confusa tanto no guia rapido quanto na parte de exemplo basico quando vc fala sobre user-scalable=no. eu tava achando q eu nao poderia usar user-scalable, mas no caso é pra evitar colocar ele como no, é isso?."

- **Detalhes da ação**: O texto existente foi reescrito para explicar de forma mais detalhada o propósito e uso do atributo `user-scalable`, bem como o comportamento padrão da *tag* de *viewport* sem ele ser configurado. Ainda adicinou-se explicação sobre outra *tag* com considerações de acessibilidade similares.

- **Resultados da ação**: As Figuras 68 e 69 mostram o texto do exemplo antes e após a implementação da melhoria.

Figura 66 – Página inicial, pré-melhorias.

Fonte: Autor.

6.3.3 Ação: Melhorar explicação sobre níveis de headings

- **Feedback relacionado (sic)**: "uma coisa que eu fiquei curiosa, vc fala q nao é pra pular as numeracoes de cabecalho ne? ai se eu quero algo menorzinho q um h3, eu coloco ele vazio e depois coloco o h4 q eu quero?"
- **Detalhes da ação**: Acrescentou-se uma breve explicação à seção de *headings*, esclarecendo que as *tags* `h1-h6` não devem ser utilizadas para fins estritamente visuais, e que o tamanho dos blocos de texto deve ser manipulado via CSS.

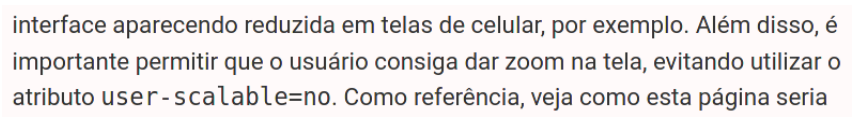
6.3.4 Ação: Adicionar imagem para exemplo correto de uso de cores

- **Feedback relacionado (sic)**: "talvez no exemplo de cores e contrantes, quando vc fala da cor nao ser o unico elemento indicativo, colocar um exemplo de como seria uma versao ideal"
- **Detalhes da ação**: Foi adicionado o exemplo de uma versão ideal, construído a partir da mesma interface apresentada nos exemplos incorretos.
- **Resultados da ação**: A Figura 70 mostra o exemplo adicionado. Para referência, a Figura 33 mostra os exemplos anteriores, de casos incorretos.

Figura 67 – Página inicial, pós-melhorias.

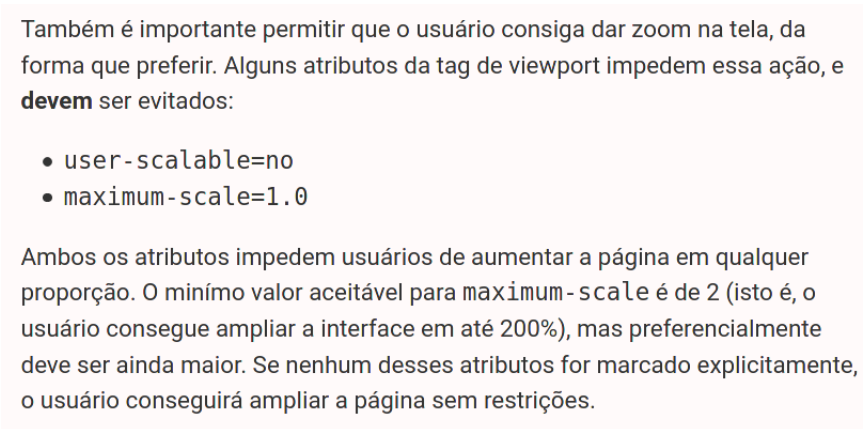
Fonte: Autor.

Figura 68 – Explicação sobre user-scalable, pré-melhorias.



Fonte: Autor.

Figura 69 – Explicação sobre user-scalable, pós-melhorias.



Fonte: Autor.

Figura 70 – Exemplo adicionado após sugestão.

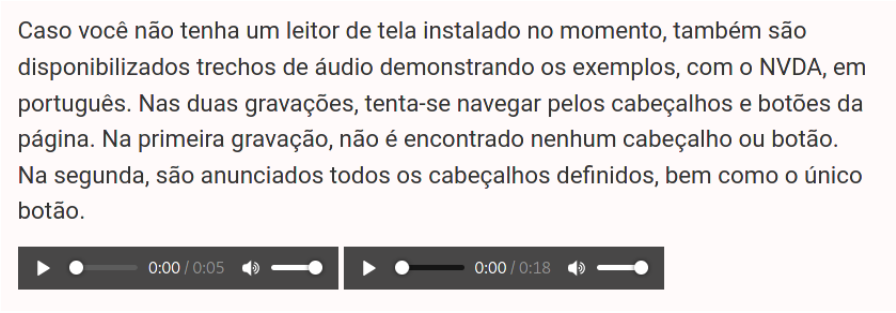
Fonte: Autor.

6.3.5 Outras Ações

Além das ações descritas anteriormente, outros dois pontos de melhoria foram identificados. O primeiro se referiu a pequenas correções de *hyperlinks* que haviam sido implementados incorretamente e não estavam funcionando.

O segundo surgiu a partir de um elogio específico sobre o exemplo com áudio sobre o atributo `lang`. Embora o comentário não desse sugestões adicionais, a partir dele se elaborou uma outra ação para acrescentar mais exemplos com áudio ao longo do apoio. Essa ação foi implementada parcialmente, ao se criar versões em áudio para os exemplos do artigo de Semântica, como visto na Figura 71. Futuramente, pretende-se incluir exemplos nesse formato para outras seções do apoio.

Figura 71 – Exemplos de áudio com descrição textual, no artigo sobre Semântica.



Fonte: Autor.

7 Considerações finais

Conforme discutido inicialmente no Capítulo 1, a acessibilidade ainda é um assunto insuficientemente considerado por muitos desenvolvedores de software, impondo barreiras ao acesso à informação e à interação com serviços, muitas vezes essenciais, para um grande número de usuários. Estima-se que cerca de 15% da população mundial (aproximadamente 1 bilhão de pessoas) possua algum tipo de deficiência (WHO, 2011). No Brasil, o levantamento mais recente indica que 24% da população (cerca de 46 milhões de pessoas) possui algum tipo de impedimento visual, auditivo, motor ou cognitivo (IBGE, 2010), sendo que 6,7% da população (cerca de 12,5 milhões de pessoas) possui impedimentos graves ou totais (IBGE, 2018). A solução deste trabalho, proposta inicialmente na Seção 1.4, implementada e descrita em detalhes no Capítulo 5, visou contribuir com a promoção de recursos disponíveis sobre o tema.

Para o sucesso da elaboração do apoio, com apresentação de informações corretas e relevantes, foi necessário um aprofundamento teórico, que permitiu a compreensão e definição mais precisa de termos importantes, bem como a identificação das principais normas técnicas e legais vigentes na atualidade relacionadas à acessibilidade. Os resultados dessa pesquisa, apresentados no Capítulo 2, garantiram o embasamento teórico do conteúdo elaborado para a solução, de acordo com o que há de mais atualizado na literatura especializada.

O produto final obtido ao término desse trabalho foi estruturado de forma a ser utilizado de algumas formas distintas: como uma fonte introdutória para conceitos e materiais básicos relacionados à acessibilidade; como uma forma de ver em mais detalhes exemplos de componentes específicos, e como um guia de acesso rápido às considerações essenciais para a acessibilidade de uma página ou *app* na Web. Esse produto pôde ser validado frente a seu público alvo de desenvolvedores, obtendo resultados positivos.

7.1 *Status* do trabalho

Em relação aos objetivos acordados na Seção 1.4, entende-se que pôde-se cumprir completamente os três primeiros objetivos específicos, ainda durante a primeira etapa do trabalho: identificar as diretrizes e recomendações mais reconhecidas e utilizadas para o desenvolvimento de interfaces acessíveis; identificar e explorar tecnologias assistivas, e identificar ferramentas de apoio à acessibilidade no desenvolvimento.

O objetivo de estruturar e priorizar as informações obtidas foi cumprido ao longo das várias iterações do projeto, com a estrutura estabelecida inicialmente sendo refinada

para melhor atender às necessidades identificadas.

Também consideram-se cumpridos os objetivos de implementação e avaliação de resultados, tendo sido os objetos de estudo no escopo de TCC2.

7.2 Trabalhos futuros

A acessibilidade é um tópico vasto e em constante evolução. Isso também implica na necessidade de se manter o apoio atualizado, de acordo com as mudanças em diretrizes e em novas publicações e descobertas da literatura especializada. Uma ação relacionada a essa necessidade é a de tornar o projeto desenvolvido *open source*, permitindo que outras pessoas interessadas no assunto possam colaborar com a construção e evolução do *site*. A versão atual do repositório[1] já contém uma versão inicial de instruções para contribuição, no próprio README do projeto.

Além dessa característica, existente devido ao próprio contexto em que o trabalho se insere, outros pontos de melhoria podem ser destacados. Alguns correspondem a ações identificadas durante os procedimentos de análise de resultados descritos no Capítulo 6, enquanto outros resultam de impressões e dificuldades encontradas durante as atividades de pesquisa e desenvolvimento. Nesse contexto, seguem algumas sugestões de trabalhos futuros:

- Adicionar seção de exemplos de mídia, como áudio e vídeos;
- Adicionar seção de exemplos de controles de tempo, como páginas que expiram após uma quantidade determinada de minutos;
- Ampliar as alternativas em áudio aos exemplos existentes;
- Adicionar exemplos específicos para o contexto de dispositivos *mobile*, tanto para *websites* quanto para aplicativos nativos, e
- Obter *feedback* e contribuições de pessoas com deficiência, tanto em relação ao conteúdo disponibilizado no apoio, quanto à acessibilidade da sua interface.

[1] Disponível em: <https://github.com/Hargre/a11y-catalog>. Acessado pela última vez em 10/05/2021

Referências

ABILITYNET. *Keyboard and mouse alternatives and adaptations*. 2019. Disponível em: <https://abilitynet.org.uk/factsheets/keyboard-and-mouse-alternatives-and-adaptations#simple-table-of-contents-11>. Citado na página 33.

ALONSO, F. et al. On the testability of WCAG 2.0 for beginners. In: *Proceedings of the 2010 International Cross Disciplinary Conference on Web Accessibility (W4A) - W4A '10*. Raleigh, North Carolina: ACM Press, 2010. p. 1. ISBN 978-1-4503-0045-2. Disponível em: <http://portal.acm.org/citation.cfm?doid=1805986.1806000>. Citado na página 37.

ANDERSON, D. J. *Kanban: successful evolutionary change for your technology business*. [S.l.]: Blue Hole Press, 2010. Citado na página 45.

ANTONELLI, H. L. et al. A survey on accessibility awareness of Brazilian web developers. In: *Proceedings of the 8th International Conference on Software Development and Technologies for Enhancing Accessibility and Fighting Info-exclusion - DSAI 2018*. Thessaloniki, Greece: ACM Press, 2018. p. 71–79. ISBN 978-1-4503-6467-6. Disponível em: <http://dl.acm.org/citation.cfm?doid=3218585.3218598>. Citado na página 24.

BARBOSA, S.; SILVA, B. *Interação humano-computador*. [S.l.]: Elsevier Brasil, 2010. Citado 3 vezes nas páginas 23, 27 e 29.

BERSCH, R. d. C. R.; PELOSI, M. B. *Tecnologia assistida: recursos de acessibilidade ao computador*. [S.l.]: SEESP, 2007. Citado na página 34.

BRAJNIK, G. A comparative test of web accessibility evaluation methods. In: *Proceedings of the 10th international ACM SIGACCESS conference on Computers and accessibility - Assets '08*. Halifax, Nova Scotia, Canada: ACM Press, 2008. p. 113. ISBN 978-1-59593-976-0. Disponível em: <http://portal.acm.org/citation.cfm?doid=1414471.1414494>. Citado 2 vezes nas páginas 36 e 37.

BRAJNIK, G.; YESILADA, Y.; HARPER, S. Testability and validity of WCAG 2.0: the expertise effect. p. 8, 2010. Citado 2 vezes nas páginas 36 e 37.

BRASIL. *DECRETO Nº 5.296 DE 2 DE DEZEMBRO DE 2004.Regulamenta as Leis nos 10.048, de 8 de novembro de 2000, que dá prioridade de atendimento às pessoas que especifica, e 10.098, de 19 de dezembro de 2000, que estabelece normas gerais e critérios básicos para a promoção da acessibilidade das pessoas portadoras de deficiência ou com mobilidade reduzida, e dá outras providências*. 2004. Disponível em: <http://www.planalto.gov.br/ccivil_03/_Ato2004-2006/2004/Decreto/D5296.htm>. Citado 2 vezes nas páginas 23 e 35.

BRASIL. *Institui a Lei Brasileira de Inclusão da Pessoa com Deficiência (Estatuto da Pessoa com Deficiência)*. 2015. Disponível em: <http://www.planalto.gov.br/ccivil_03/_ato2015-2018/2015/lei/l13146.htm>. Citado na página 35.

BRASIL. MINISTéRIO DO PLANEJAMENTO, ORçAMENTO E GESTãO. SECRETARIA DE LOGíSTICA E TECNOLOGIA DA INFORMAçãO. *eMAG Modelo de Acessibilidade em Governo Eletrônico*. Brasília, 2014. Citado 2 vezes nas páginas 23 e 35.

BRASIL. SUBSECRETARIA NACIONAL DE PROMOçãO DOS DIREITOS DA PESSOA COM DEFICIêNCIA. COMITê DE AJUDAS TéCNICAS. *Tecnologia Assistiva*. [S.l.], 2009. Citado na página 30.

CHUNG, L. et al. *Non-functional requirements in software engineering*. [S.l.]: Springer Science & Business Media, 2000. v. 1. Citado na página 56.

COHEN, R. F. et al. Accessibility in introductory computer science. In: *Proceedings of the 36th SIGCSE technical symposium on Computer science education - SIGCSE '05*. St. Louis, Missouri, USA: ACM Press, 2005. p. 17. ISBN 978-1-58113-997-6. Disponível em: <http://portal.acm.org/citation.cfm?doid=1047344.1047367>. Citado 2 vezes nas páginas 24 e 81.

Comitê Gestor da Internet no Brasil. *Survey on the use of information and communication technologies in brazilian households: ICT households 2018 [livro eletrônico]*. [S.l.]: Núcleo de Informação e Coordenação do Ponto BR, 2018. Citado na página 23.

EL-GLALY, Y. N. Teaching Accessibility to Software Engineering Students. In: *Proceedings of the 51st ACM Technical Symposium on Computer Science Education*. Portland OR USA: ACM, 2020. p. 121–127. ISBN 978-1-4503-6793-6. Disponível em: <https://dl.acm.org/doi/10.1145/3328778.3366914>. Citado 2 vezes nas páginas 24 e 81.

ENGEL, G. I. Pesquisa-ação. *Educar em Revista*, SciELO Brasil, n. 16, p. 181–191, 2000. Citado na página 44.

FREIRE, A. P.; RUSSO, C. M.; FORTES, R. P. M. A survey on the accessibility awareness of people involved in web development projects in Brazil. In: *Proceedings of the 2008 international cross-disciplinary workshop on Web accessibility (W4A) - W4A '08*. Beijing, China: ACM Press, 2008. p. 87. ISBN 978-1-60558-153-8. Disponível em: <http://portal.acm.org/citation.cfm?doid=1368044.1368064>. Citado na página 24.

GAY, G.; DJAFAROVA, N.; ZEFI, L. Teaching Accessibility to the Masses. In: *Proceedings of the 14th Web for All Conference on The Future of Accessible Work - W4A '17*. Perth, Western Australia, Australia: ACM Press, 2017. p. 1–8. ISBN 978-1-4503-4900-0. Disponível em: <http://dl.acm.org/citation.cfm?doid=3058555.3058563>. Citado 2 vezes nas páginas 24 e 81.

GERHARDT, T. E.; SILVEIRA, D. T. *Métodos de pesquisa*. [S.l.]: Plageder, 2009. Citado 2 vezes nas páginas 43 e 44.

GIL, A. C. *Métodos e técnicas de pesquisa social*. [S.l.]: 6. ed. Ediitora Atlas SA, 2008. Citado na página 44.

GLOBAL INITIATIVE FOR INCLUSIVE INFORMATION AND COMMUNICATION TECHNOLOGIES. *Convention on the Rights of Persons with Disabilities 2016 ICT Accessibility Progress Report*. [S.l.], 2016. Citado na página 24.

HENRY, S. L. *Just ask: integrating accessibility throughout design*. [S.l.]: Lulu. com, 2007. Citado 3 vezes nas páginas 36, 37 e 38.

HENRY, S. L.; ABOU-ZAHRA, S.; BREWER, J. The role of accessibility in a universal web. In: *Proceedings of the 11th Web for All Conference on - W4A '14*. Seoul, Korea: ACM Press, 2014. p. 1–4. ISBN 978-1-4503-2651-3. Disponível em: <http://dl.acm.org/citation.cfm?doid=2596695.2596719>. Citado na página 29.

INSTITUTO BRASILEIRO DE GEOGRAFIA E ESTATíSTICA. *Censo Demográfico 2010*: Características gerais da população, religião e pessoas com deficiência. [S.l.]: Instituto Brasileiro de Geografia e Estatística, 2010. Citado na página 97.

INSTITUTO BRASILEIRO DE GEOGRAFIA E ESTATíSTICA. Censo demográfico 2010 nota técnica 01/2018: Releitura dos dados de pessoas com deficiência no censo demográfico 2010 à luz das recomendações do grupo de washington. 2018. Citado na página 97.

INTERNATIONAL ORGANIZATION FOR STANDARDIZATION. *IEC 40500: 2012*. 2012. Citado na página 34.

KELLY, B. et al. Accessibility 2.0: people, policies and processes. In: *Proceedings of the 2007 international cross-disciplinary conference on Web accessibility (W4A) - W4A '07*. Banff, Canada: ACM Press, 2007. p. 138. Disponível em: <http://portal.acm.org/citation.cfm?doid=1243441.1243471>. Citado na página 44.

LUDI, S. et al. Teaching Inclusive Thinking to Undergraduate Students in Computing Programs. In: *Proceedings of the 49th ACM Technical Symposium on Computer Science Education*. Baltimore Maryland USA: ACM, 2018. p. 717–722. ISBN 978-1-4503-5103-4. Disponível em: <https://dl.acm.org/doi/10.1145/3159450.3159512>. Citado na página 81.

MELO, A.; CECíLIA, M.; BARANAUSKAS, M. C. Design e avaliação de tecnologia web-acessível. 01 2005. Citado 3 vezes nas páginas 27, 31 e 32.

NIELSEN, J. *Usability engineering*. [S.l.]: Morgan Kaufmann, 1993. Citado 2 vezes nas páginas 28 e 29.

NIELSEN, J. *Accessibility Is Not Enough*. 2005. Disponível em: <https://www.nngroup.com/articles/accessibility-is-not-enough/>. Citado 2 vezes nas páginas 29 e 37.

OLIVEIRA, M. F. D. Metodologia científica: um manual para a realização de pesquisas em administração. *Universidade Federal de Goiás. Catalão–GO*, 2011. Citado 2 vezes nas páginas 43 e 44.

OLIVEIRA, R. et al. Eliciting accessibility requirements an approach based on the NFR framework. In: *Proceedings of the 31st Annual ACM Symposium on Applied Computing - SAC '16*. Pisa, Italy: ACM Press, 2016. p. 1276–1281. ISBN 978-1-4503-3739-7. Disponível em: <http://dl.acm.org/citation.cfm?doid=2851613.2851759>. Citado na página 56.

OLIVEIRA, R. F. D. Um Método Semi-automatizado para Elicitação de Requisitos de Acessibilidade Web. p. 205, 2014. Citado 2 vezes nas páginas 56 e 111.

PICHILIANI, T. C. P. B.; PIZZOLATO, E. B. A survey on the awareness of brazilian web development community about cognitive accessibility. In: *Proceedings of the 18th Brazilian Symposium on Human Factors in Computing Systems*. Vitória Espírito Santo Brazil: ACM, 2019. p. 1–11. ISBN 978-1-4503-6971-8. Disponível em: <http://dl.acm.org/doi/10.1145/3357155.3358448>. Citado na página 24.

PREECE, J.; SHARP, H.; ROGERS, Y. *Interaction design: beyond human-computer interaction*. [S.l.]: John Wiley & Sons, 2019. Citado 3 vezes nas páginas 27, 28 e 30.

PRODANOV, C. C.; FREITAS, E. C. de. *Metodologia do trabalho científico: métodos e técnicas da pesquisa e do trabalho acadêmico-2ª Edição*. [S.l.]: Editora Feevale, 2013. Citado na página 44.

SCHWABER, K.; BEEDLE, M. *Agile software development with Scrum*. [S.l.]: Prentice Hall Upper Saddle River, 2002. v. 1. Citado na página 45.

SHINOHARA, K. et al. Who Teaches Accessibility?: A Survey of U.S. Computing Faculty. In: *Proceedings of the 49th ACM Technical Symposium on Computer Science Education*. Baltimore Maryland USA: ACM, 2018. p. 197–202. ISBN 978-1-4503-5103-4. Disponível em: <https://dl.acm.org/doi/10.1145/3159450.3159484>. Citado na página 24.

SHNEIDERMAN, B. et al. *Designing the user interface: strategies for effective human-computer interaction*. [S.l.]: Pearson, 2018. Citado 3 vezes nas páginas 27, 29 e 34.

SILVA, J. et al. Accessibility in Software Engineering: Pursuing the Mainstream from a Classroom. In: ZAPHIRIS, P.; IOANNOU, A. (Ed.). *Learning and Collaboration Technologies. Learning and Teaching*. Cham: Springer International Publishing, 2018. v. 10925, p. 505–517. ISBN 978-3-319-91151-9 978-3-319-91152-6. Series Title: Lecture Notes in Computer Science. Disponível em: <http://link.springer.com/10.1007/978-3-319-91152-6_39>. Citado na página 24.

STARON, M. *Action Research in Software Engineering*. Springer International Publishing, 2020. Disponível em: <https://doi.org/10.1007/978-3-030-32610-4>. Citado 2 vezes nas páginas 50 e 51.

WORLD HEALTH ORGANIZATION. *World report on disability 2011*. [S.l.]: World Health Organization, 2011. Citado na página 97.

WORLD WIDE WEB CONSORTIUM. *User Agent Accessibility Guidelines 1.0*. World Wide Web Consortium, 2002. Disponível em: <https://www.w3.org/TR/ATAG/>. Citado na página 34.

WORLD WIDE WEB CONSORTIUM. *Guidance on Applying WCAG 2.0 to Non-Web Information and Communications Technologies (WCAG2ICT)*. World Wide Web Consortium, 2013. Disponível em: <https://www.w3.org/TR/wcag2ict/>. Citado na página 34.

WORLD WIDE WEB CONSORTIUM. *Authoring Tool Accessibility Guidelines (ATAG) 2.0*. World Wide Web Consortium, 2015. Disponível em: <https://www.w3.org/TR/ATAG/>. Citado na página 34.

WORLD WIDE WEB CONSORTIUM. *Mobile Accessibility: How WCAG 2.0 and Other W3C/WAI Guidelines Apply to Mobile*. World Wide Web Consortium, 2015. Disponível em: <https://www.w3.org/TR/mobile-accessibility-mapping/>. Citado na página 34.

WORLD WIDE WEB CONSORTIUM. *Accessibility, Usability, and Inclusion*. World Wide Web Consortium, 2016. Disponível em: <https://www.w3.org/WAI/fundamentals/accessibility-usability-inclusion/>. Citado na página 29.

WORLD WIDE WEB CONSORTIUM. *Accessible Rich Internet Applications (WAI-ARIA) 1.1*. [S.l.], 2017. Disponível em: <https://www.w3.org/TR/wai-aria/>. Citado 2 vezes nas páginas 23 e 34.

WORLD WIDE WEB CONSORTIUM. *Tools and Techniques*. World Wide Web Consortium, 2017. Disponível em: <https://www.w3.org/WAI/people-use-web/tools-techniques/>. Citado 4 vezes nas páginas 30, 31, 32 e 34.

WORLD WIDE WEB CONSORTIUM. *Understanding Success Criterion 1.4.6: Contrast (Enhanced)*. [S.l.], 2018. Disponível em: <https://www.w3.org/WAI/WCAG21/Understanding/contrast-enhanced.html>. Citado na página 57.

WORLD WIDE WEB CONSORTIUM. *WCAG 2 Documents*. World Wide Web Consortium, 2018. Disponível em: <https://www.w3.org/WAI/standards-guidelines/wcag/docs/>. Citado na página 36.

WORLD WIDE WEB CONSORTIUM. *Web content accessibility guidelines (WCAG) 2.1*. [S.l.], 2018. Disponível em: <https://www.w3.org/TR/2018/REC-WCAG21-20180605/>. Citado na página 23.

WORLD WIDE WEB CONSORTIUM. *Introduction to Web Accessibility*. World Wide Web Consortium, 2019. Disponível em: <https://www.w3.org/WAI/fundamentals/accessibility-intro/>. Citado 2 vezes nas páginas 29 e 30.

YAN, S.; RAMACHANDRAN, P. G. The Current Status of Accessibility in Mobile Apps. *ACM Transactions on Accessible Computing*, v. 12, n. 1, p. 1–31, fev. 2019. ISSN 19367228. Disponível em: <http://dl.acm.org/citation.cfm?doid=3312747.3300176>. Citado na página 24.

Apêndices

APÊNDICE A – Procedimentos da Pesquisa Bibliográfica

A.1 Critérios de Pesquisa

No que diz respeito à seleção das bases de pesquisa e artigos a serem utilizados para o trabalho, foram seguidos os seguintes critérios:

- Os artigos devem estar disponíveis em bases de pesquisa de renome científico, especificamente em contextos relacionados à Engenharia de Software;
- Os artigos devem estar disponíveis de forma gratuita, ou acessíveis através da rede da UnB;
- Os artigos devem estar disponíveis em formato eletrônico, em sua totalidade;
- Os artigos devem estar escritos em português ou inglês;
- Os artigos devem abordar um ou mais dos seguintes tópicos:
  - Acessibilidade de Software de forma geral;
  - Acessibilidade de Software no contexto de um ambiente específico (*Web*, *Mobile*, *Desktop*);
  - Acessibilidade de Software relacionada a outros critérios de Qualidade, como Usabilidade;
  - Métodos e normas de Avaliação de Acessibilidade de Software, e
  - Tecnologias relacionadas à Acessibilidade de Software.

A principal fonte de pesquisa para a seleção dos artigos foi a *ACM Digital Library* (ACM-DL) [1]. Também foram utilizadas a *IEEE Xplore Digital Library* [2] e a *Scopus* [3].

A.2 *Strings* de Busca e Resultados

De acordo com as atividades descritas na Seção 4.2.3, foram elaboradas várias *strings* de busca ao longo da pesquisa. Novas *strings* foram elaboradas ou refinadas à

[1] https://dl.acm.org/
[2] https://ieeexplore.ieee.org/Xplore/home.jsp
[3] https://www.scopus.com/home.uri

medida que novas necessidades de pesquisa eram evidenciadas, de acordo com os resultados obtidos a cada etapa. A seguir são apresentadas as principais *strings* de busca utilizadas, bem como a quantidade de artigos selecionada em cada busca.

- **String de Busca: *software accessibility***
  - Resultados Selecionados - ACM-DL: 17
  - Resultados Selecionados - IEEE Xplore: 3
  - Resultados Selecionados - Scopus: 2
  - **Comentários**: *String* inicial de pesquisa. Resultados abordam o tema em contextos diversos, de forma ampla. Levou à criação de novas *strings* para explorar o tema em contextos e ambientes mais específicos.
- **String de Busca: *(software accessibility) AND (mobile)***
  - Resultados Selecionados - ACML-DL: 12
- **String de Busca: *(software accessibility) AND (desktop) NOT (web)***
  - Resultados Selecionados - ACML-DL: 3
  - **Comentários**: *String* produzida devido à dificuldade de encontrar artigos abordando o assunto da Acessibilidade no contexto de aplicações *Desktop*. Sem o filtro "*NOT (web)*"os resultados eram muito mais relacionados à Acessibilidade da Web.
- **String de Busca: *accessibility survey***
  - Resultados Selecionados - ACML-DL: 4
  - **Comentários**: Utilizada com o objetivo de encontrar *surveys* relacionados à percepção e consciência por parte de desenvolvedores e estudantes em relação à Acessibilidade de Software. Ainda foram selecionadas outras pesquisas, a partir de referências citadas nos artigos selecionados.
- **String de Busca: *(software accessibility) AND (survey) AND (web OR mobile OR desktop)***
  - Resultados Selecionados - Scopus: 2
  - **Comentários**: Versão refinada da *string* anterior, usada para obter resultados mais precisos ao contexto de Software.
- **String de Busca: *assistive technology***
  - Resultados Selecionados - ACM-DL: 2

- **String de Busca:** ***(accessibility) AND (evaluation OR testing)***
  - Resultados Selecionados - ACM-DL: 3
- **String de Busca:** ***WCAG***
  - Resultados Selecionados - ACM-DL: 3
  - **Comentários**: Pesquisa realizada para obter opiniões acadêmicas sobre os padrões e normas técnicas da WCAG.
- **String de Busca:** ***(universal design) AND (web OR software OR accessibility OR digital*)**
  - Resultados Selecionados - ACM-DL 4

Total de Resultados Selecionados: ***55***

Complementarmente, ainda foram selecionadas fontes bibliográficas obtidas com outros procedimentos, tais como consulta a especialistas e uso de normas técnicas e legais.

APÊNDICE B – Ciclos de pesquisa-ação

B.1 Ciclo 2 - Tabela completa

A seguir é apresentada a tabela completa utilizada para a inspeção do segundo ciclo de análise de resultados do projeto. Devido ao tamanho da tabela, a mesma foi dividida em vários pedaços, sendo apresentada aqui das Figuras 75 a 81. A tabela foi criada pelo autor, a partir da relação de operacionalizações de (OLIVEIRA, 2014), disponível em <http://omnesweb.dimap.ufrn.br/index.php/artefatos/acessibilidade/tabela> e acessada pela última vez em 10/05/2021.

Figura 72 – Tabela de inspeção de operacionalizações (1/10).

Acessibilidade				
Princípios	**Requisitos**		**Operacionalizações**	**Implementado?**
	Diretrizes	**Refinamentos**		
Conteúdos disponibilizados de maneira perceptível	Alternativas para mídias baseadas no tempo de execução	Alternativas para conteúdos de áudio (ao vivo)	Fornecer alternativas textuais para conteúdo de áudio (ao vivo)	Não se aplica (AAA)
			Fornecer um link para descrever textualmente o conteúdo de áudio	Não se aplica (AAA)
		Disponibilidade de legendas para mídias pré-gravadas e ao vivo	Fornecer bibliotecas de legendas para vídeos	Não
			Fornecer legendas através de SMIL (Synchronized Multimedia Integration Language)	Não se aplica (AAA)
		Alternativas pré-gravadas para tipo de conteúdo vídeo	Fornecer um link que aponte para as opções do conteúdo de vídeo	Não
			Utilizar o corpo da tag object para prover uma alternativa textual	Não
			Associar descrições estendidas de áudio para vídeos	Não
		Disponibilidade de linguagens de sinais	Incluir um intérprete de língua gestual no fluxo de vídeo	Não se aplica (AAA)
			Fornecer um vídeo sincronizado contendo o intérprete de língua gestual em uma janela diferente	Não se aplica (AAA)
		Alternativas para mídias pré- gravadas do tipo áudio e vídeo	Fornecer mídias alternativas para tipos de conteúdo baseados apenas em áudio	Não
			Fornecer documentos contendo as mesmas informações do conteúdo	Não
			Fornecer mídias alternativas para tipos de conteúdo baseados apenas em vídeo	Não
			Fornecer áudio que descreva o conteúdo de vídeo	Não
			Fornecer uma alternativa estática do conteúdo através do MediaElement em Silverlight	Não se aplica (obsoleto)
		Alternativas contendo descrições de áudio	Associar descrições estendidas de áudio para vídeos	Não
			Fornecer uma segunda faixa de áudio contendo a descrição do áudio principal	Não
			Fornecer um vídeo contendo as descrições do áudio	Não

Figura 73 – Tabela de inspeção de operacionalizações (2/10).

	Conteúdo apresentado de diferente maneiras	Conteúdo preservado independente do formato da apresentação	Separar a apresentação do conteúdo	Implementado
			Utilizar a marcação de tabelas para apresentar informações tabulares	Implementado
			Utilizar elementos de marcação com semânticas apropriadas	Implementado
		Informações disponibilizadas através de sequências bem definidas	Ordenar o conteúdo por sequências que preservem seu significado	Implementado
			Utilizar CSS para controlar espaçamento e posicionamento do conteúdo	Não
			Usar propriedade tabIndex em flash para viabilizar uma sequência lógica para conteúdos	Não se aplica (obsoleto)
		Conteúdos disponibilizados com instruções que não dependem de características sensoriais para serem compreendidas	Fornecer uma identificação textual de itens que dependem de características sensoriais para serem compreendidos	Implementado
	Facilidade na audição e visualização das informações disponibilizadas	Controle sobre conteúdos de áudio	Fornecer controle ao usuário permitindo que este interrompa o som quando quiser	Não
			Reproduzir sons até no máximo 3 segundos após a página ser acessada	Não
			Reproduzir som apenas a pedido do utilizador	Não
		Permissão para a manipulação de textos	Fornecer controles na página que permitam ao usuário alterar o tamanho do texto em até 200%	Não
			Utilizar porcentagem para definir tamanho de fonte	Não
		Controle para contrastes de cores	Equilibrar contraste entre o texto e a cor do plano de fundo	Implementado
			Permitir que os usuários alterem o primeiro plano e plano de fundo dos blocos textuais	Não se aplica (AAA)
		Controle para conteúdo de áudio executado em segundo plano	Efetuar configuração para que o áudio principal fique 20 decibéis mais alto que o áudio secundário	Não se aplica (AAA)
			Permitir que os usuários possam ajustar sons executados em segundo plano	Não se aplica (AAA)
		Controle para imagens de textos	Utilizar CSS para substituir imagens de textos por textos	Não

Figura 74 – Tabela de inspeção de operacionalizações (3/10).

			Executar OCR em um documento PDF para fornecer texto real	Não se aplica (formato)
		Mecanismos de controle para a apresentação visual dos blocos de textos	Evitar o uso de textos justificados: Especificar o alinhamento tanto para esquerda como para direita Alinhar textos para apenas um lado	Implementado
			Permitir que o usuário controle as cores de fundo: Não especificar cor de fundo e nem do texto Fornecer ao usuário um painel para seleção de cores	Não se aplica (AAA)
			Ajustar conteúdo para navegação vertical em dispositivos mobiles	Não
			Configurar o espaçamento entre as linhas para no mínimo 1.5	Implementado
			Permitir que o usuário redimensione o texto em até 200% sem necessidade de tecnologias assistivas: Utilizar layout líquido Utilizar script para calcular tamanho e posicionamento do conteúdo	Implementado
		Alternativas para associar ao uso de cores	Utilizar caracteres para indicar obrigatoriedade de preenchimento para campos de formulários	Implementado
			Alterar a apresentação de componentes de interface quando estes receberem o foco	Implementado
			Fornecer alternativas textuais para informações transmitidas através de cores	Implementado
	Alternativas para conteúdos não textuais	Conteúdos não textuais disponibilizados através de alternativas textuais	Substituir conteúdos não textuais por textos	Implementado
			Fornecer alternativas textuais curtas: Utilizar atributos alt em tags contendo imagens Fornecer alternativas textuais para arte ASCII, emoticons e leetspeak Utilizar atributos alt em tags contendo imagens Utilizar alternativa textual para um item dentro de um grupo de imagens que descreva todos os itens do grupo	Implementado
			Fornecer descrições para controles de formulário: Utilizar atributos alt em tags contendo imagens Fornecer descrições para elementos de mapas de imagens	Implementado

Figura 75 – Tabela de inspeção de operacionalizações (4/10).

Acessibilidade				
Princípios	**Requisitos**		**Operacionalizações**	**Implementado?**
	Diretrizes	**Refinamentos**		
Conteúdos disponibilizados de maneira perceptível	Alternativas para mídias baseadas no tempo de execução	Alternativas para conteúdos de áudio (ao vivo)	Fornecer alternativas textuais para conteúdo de áudio (ao vivo)	Não se aplica (AAA)
			Fornecer um link para descrever textualmente o conteúdo de áudio	Não se aplica (AAA)
		Disponibilidade de legendas para mídias pré-gravadas e ao vivo	Fornecer bibliotecas de legendas para vídeos	Não
			Fornecer legendas através de SMIL (Synchronized Multimedia Integration Language)	Não se aplica (AAA)
		Alternativas pré-gravadas para tipo de conteúdo vídeo	Fornecer um link que aponte para as opções do conteúdo de vídeo	Não
			Utilizar o corpo da tag object para prover uma alternativa textual	Não
			Associar descrições estendidas de áudio para vídeos	Não
		Disponibilidade de linguagens de sinais	Incluir um intérprete de língua gestual no fluxo de vídeo	Não se aplica (AAA)
			Fornecer um vídeo sincronizado contendo o intérprete de língua gestual em uma janela diferente	Não se aplica (AAA)
		Alternativas para mídias pré- gravadas do tipo áudio e vídeo	Fornecer mídias alternativas para tipos de conteúdo baseados apenas em áudio	Não
			Fornecer documentos contendo as mesmas informações do conteúdo	Não
			Fornecer mídias alternativas para tipos de conteúdo baseados apenas em vídeo	Não
			Fornecer áudio que descreva o conteúdo de vídeo	Não
			Fornecer uma alternativa estática do conteúdo através do MediaElement em Silverlight	Não se aplica (obsoleto)
		Alternativas contendo descrições de áudio	Associar descrições estendidas de áudio para vídeos	Não
			Fornecer uma segunda faixa de áudio contendo a descrição do áudio principal	Não
			Fornecer um vídeo contendo as descrições do áudio	Não

Figura 76 – Tabela de inspeção de operacionalizações (5/10).

			Codificar os dados de usuários como dados ocultos ou encriptados	Não se aplica (AAA)
		Controle para pausar, parar e ocultar apresentação de conteúdos	Permitir que o conteúdo seja pausado e reiniciado de onde foi pausado	Não
			Definir scripts para controle sobre gifs animados permitindo parar suas animações após n ciclos (ao menos 5 segundos)	Não
			Disponibilizar ao usuário um controle que permita ocultar, parar ou pausar a atualização ou movimentação de um conteúdo	Implementado
			Fornecer um link, um botão ou um mecanismo que permita carregar a página sem conteúdo piscando	Implementado
	Tratamento de conteúdo para que não cause convulsões	Quantidade de flashes limitados	Limitar para 3 o número de flashes por segundo em conteúdo animados	Não se aplica (AAA)
			Manter o conteúdo piscando em uma área pequena que seja menor do que 25 % em relação a 10 Graus da visão humana	Implementado
			Utilizar técnicas que possam garantir que o conteúdo não viole o limite de flashes por segundo	Não
	Suporte para a navegação no conteúdo	Identificação da finalidade de cada link	Fornecer alternativas textuais para elementos de área contidos em mapas de imagens	Não
			Fornecer texto descrevendo a finalidade do link	Implementado
		Alternativas para localizar uma página dentro do site	Ligar todas as páginas com a página principal	Implementado
			Fornecer links para navegar em páginas relacionadas	Implementado
			Fornecer um índice	Não
			Fornecer uma lista de links para todas as páginas	Não
			Fornecer uma função de pesquisa de páginas	Não
			Fornecer um mapa do site	Não
		Suporte para localização dentro do site	Fornecer a localização atual]	Não se aplica (AAA)
			Fornecer rastros para indicar uma trilha	Não se aplica (AAA)
			Fornecer um mapa do site	Não se aplica (AAA)

Figura 77 – Tabela de inspeção de operacionalizações (6/10).

		Conteúdos disponibilizados de maneiras bem definidas	Fornecer títulos para descrição das páginas	Implementado
			Fornecer elementos de cabeçalhos descrevendo os conteúdos	Implementado
			Fornecer elementos de etiquetas para os componentes interativos do conteúdo	Implementado
			Fornecer páginas utilizando elementos de cabeçalhos por sessão	Implementado
		Gerenciamento sobre blocos de conteúdos	Fornecer links que permitam pular blocos de conteúdo repetido: Adicionar um link no início de cada bloco de conteúdo repetido para direcionar ao fim do bloco Adicionar um link no topo de cada página que direciona o usuário para a página principal Adicionar um link no topo de cada página que direciona o usuário para a página principal Adicionar links no topo de cada página para áreas de conteúdo	Implementado
			Fornecer agrupamento de blocos para conteúdos repetidos de forma que possam ser ignorados: Usar mapas para agrupar links Fornecer elementos de cabeçalhos para cada seção de conteúdo Fornecer elementos de cabeçalhos para cada seção de conteúdo	Não se aplica (AAA)
		Ordenação do foco sobre o conteúdo	Ajustar os elementos interativos para seguir as sequências e relacionamento dentro do conteúdo	Implementado
			Garantir ordem para tabulação entre e links, controle de formulários e objetos	Implementado
		Foco sempre visível	Alterar a apresentação de um componente de interface quando este receber o foco	Implementado
			Utilizar componentes de interface que possam ter focos destacados por agentes de usuário	Implementado
Conteúdos disponibilizados de maneira compreensível	Conteúdos de de texto disponibilizados de maneira legível e compreensível	Páginas fornecidas com idiomas definidos	Utilizar atributos de linguagem lang	Implementado
			Especificar a linguagem para passagens ou frases em documentos PDF	Não se aplica (formato)

Figura 78 – Tabela de inspeção de operacionalizações (7/10).

		Tratamento para a complexidade dos textos disponibilizados	Fornecer textos resumidos que exijam pouca habilidade para compreensão	Não se aplica (AAA)
			Fornecer imagens e símbolos para ajudar na compreensão de textos	Não se aplica (AAA)
			Fornecer uma versão falada do texto	Não se aplica (AAA)
			Tornar textos simples de ler	Não se aplica (AAA)
		Tratamento para trechos contendo lingua estrangeira	Utilizar atributos lang com xml para identificar alterações no idioma	Não se aplica (formato)
			Especificar a linguagem com elemento land para uma passagem ou frase de documentos PDF	Não se aplica (formato)
		Tratamento para palavras incomuns	Fornecer uma função que possibilita pesquisar em um dicionário on line	Não se aplica (AAA)
			Fornecer definições para palavras ou frases utilizadas de maneira informal ou restrita	Não se aplica (AAA)
			Utilizar listas de definições	Não se aplica (AAA)
			Fornecer um glossário	Não se aplica (AAA)
			Utilizar elemento link para vincular termos a glossários	Não se aplica (AAA)
			Utilizar elemento dfn para identificar uma instância que define uma palavra	Não se aplica (AAA)
		Tratamento de pronúncias	Fornecer uma pronúncia imediatamente após a palavra	Não se aplica (AAA)
			Fornecer pronúncia falada	Não se aplica (AAA)
	Páginas com funcionamento previsível	Navegação consistente	Preservar a ordem dos mecanismos de navegação repetidos	Não
			Fornecer cabeçalho e rodapé na execução de documentos PDF	Não se aplica (formato)
		Tratamento do foco para mudança de contextos	Utilizar elemento "activate" em vez "focus" como gatilho para alterações de contextos	Não
			Abrir nova aba ou janela quando for realmente necessário	Não
			Informar ao usuário antes de abrir uma nova janela	Não
		Tratamento para mudanças de contextos	Fornecer botões para iniciar mudança de contexto	Implementado

Figura 79 – Tabela de inspeção de operacionalizações (8/10).

			Fornecer elemento select para executar uma ação	Não
		Alterações de contexto mediante a solicitações	Permitir que o usuário solicite atualizações de conteúdo ao invés de atualizar automaticamente	Não se aplica (AAA)
			Fornecer tratamento para redirecionamento automático: Fornecer instâncias para redirecionamentos automáticos no lado servidor da aplicação Fornecer instâncias para redirecionamentos automáticos no lado cliente da aplicação	Não se aplica (AAA)
			Tratar pop-up: Fornecer um atributo target para abrir uma nova janela Utilizar avanço progressivo via script para abrir novas janelas	Não se aplica (AAA)
		Conteúdos disponibilizados com identificação consistente	Utilizar labels, nomes e alternativas textuais de forma consistente para conteúdos que possuem mesmos sentidos	Implementado
			Utilizar textos consistentes para transmitir a função dos componentes de interface	Implementado
	Correção e prevenção de erros	Tratamento para identificação dos erros	Fornecer tratamento para informar ausência de dados em campos de preenchimento obrigatório: Fornecer descrição textual informando que o campo não foi preenchido Tratar validação e alerta pelo lado do servidor da aplicação Fornecer descrição textual informando que o campo não foi preenchido	Implementado
			Fornecer tratamento para dados que devem obedecer um determinado formato: Fornecer uma descrição textual informando que os dados informados não estão em um formato válido Fornecer uma descrição textual informando que os dados informados não estão em um formato válido Fornecer uma descrição textual informando que os dados informados não estão na lista de valores permitidos	Implementado
		Suporte para a entrada de informações	Fornecer etiquetas descrevendo os dados esperados: Fornecer descrição sobre o formato esperado para os dados fornecidos	Implementado

Figura 80 – Tabela de inspeção de operacionalizações (9/10).

			Fornecer instruções no inicio do formulário	
			Utilizar elementos etiqueta para associar textos a controle de formulários	Implementado
		Sugestões para correções para erros	Fornecer uma descrição textual informando os erros e sugerindo correções	Implementado
			Fornecer alertas para as validações efetuadas pelo lado cliente da aplicação	Implementado
		Suporte para usar o site	Tratar ajuda para entrada de texto: Fornecer link de ajuda em todas as páginas Fornecer verificação ortográfica e sugestões para entrada de texto Fornecer assistente de ajuda em todas as páginas	Não se aplica (AAA)
			Fornecer ajuda durante o preenchimento de formulários	Não se aplica (AAA)
		Prevenção contra todos os tipos de erros	Fornecer tratamento para prevenção de erros em transações legais: Fornecer a possibilidade do usuário rever as respostas antes de submeter Fornecer elemento checkbox além do botão de submissão	Não
			Fornecer tratamento para ações que possam excluir informações: Fornecer elemento checkbox além do botão de submissão	Não
			Fornecer formas de recuperar as informações	Não
			Solicitar confirmação para continuar a ação selecionada	Não
Conteúdos disponibilizados de maneira robusta	Conteúdos disponibilizados com o máximo de compatibilidade	Tratamento para os valores dos componentes de interface do usuário	Utilizar funcionalidades de marcação para mostrar os nomes e funções dos componentes: Fornecer atributos de identificação exclusivos para cada controle de formulário Fornecer elementos etiqueta para associar aos controles de formulário Fornecer elemento título para associar aos controles de formulário	Implementado
			Utilizar os recursos de API´s de acessibilidade para mostrar nomes e funções dos componentes	Não se aplica (formato)

Figura 81 – Tabela de inspeção de operacionalizações (10/10).

		Páginas analisadas quanto a formatação	Fornecer páginas em conformidade com as especificações de boas práticas para as linguagens de marcação: Garantir que os elementos não contêm atributos duplicados Garantir que as tags estão abertas e fechadas corretamente	Implementado
			Validar páginas: Avaliação manual para validação das normas de acessibilidade que exigem um julgamento humano Utilizar ferramentas de avaliação automática para validação do HTML , CSS e normas de acessibilidade	Implementado