INSTITUTO FEDERAL DE SANTA CATARINA

LAURA VIVAN GONÇALVES

DESENVOLVIMENTO DE UM SISTEMA PARA COMPARTILHAMENTO DE DIRETRIZES DE ACESSIBILIDADE NA WEB

Caçador

2025

LAURA VIVAN GONÇALVES

DESENVOLVIMENTO DE UM SISTEMA PARA COMPARTILHAMENTO DE DIRETRIZES DE ACESSIBILIDADE NA WEB

Monografia apresentada ao curso de Sistemas de Informação do Campus Caçador do Instituto Federal de Santa Catarina para a obtenção do diploma de Bacharelado.

Orientadora: Ma. Taynara Cerigueli Dutra

Coorientador: Dr. Jair José Ferronato

Caçador

2025

Gonçalves, Laura Vivan.
G658d Desenvolvimento de um sistema para compartilhamento de diretrizes de acessibilidade na web / Laura Vivan Gonçalves ; orientador: Taynara Cerigueli Dutra, coorientador: Jair José Ferronato. -- 2025.
74 f.

Trabalho de Conclusão de Curso (Graduação)-Instituto Federal de Educação, Ciência e Tecnologia de Santa Catarina, Caçador, 2025.
Inclui bibliografias.

1. Acessibilidade digital. 2. Acessibilidade digital na web. 3. Sistema web. 4. Sistema web acessível. I. Dutra, Taynara Cerigueli. II. Ferronato, Jair José. III. Instituto Federal de Educação, Ciência e Tecnologia de Santa Catarina – Graduação em Sistemas de Informação. IV. Título.

CDD 621

Ficha catalográfica elaborada pela Bibliotecária
Janice Moser Corrêa – CRB-14/1865

LAURA VIVAN GONÇALVES

DESENVOLVIMENTO DE UM SISTEMA PARA COMPARTILHAMENTO DE DIRETRIZES DE ACESSIBILIDADE NA WEB

Este trabalho foi julgado adequado para obtenção do título de Bacharel em Sistemas de Informação, pelo Instituto Federal de Educação, Ciência e Tecnologia de Santa Catarina, e aprovado na sua forma final pela comissão avaliadora abaixo indicada.

Caçador, 11 de julho de 2025.

gov.br Documento assinado digitalmente
TAYNARA CERIGUELI DUTRA
Data: 07/08/2025 17:41:50-0300
Verifique em https://validar.iti.gov.br

Profa. Taynara Cerigueli Dutra, Ma.
Orientadora
Instituto Federal de Santa Catarina

gov.br Documento assinado digitalmente
JAIR JOSE FERRONATO
Data: 07/08/2025 20:38:41-0300
Verifique em https://validar.iti.gov.br

Prof. Jair José Ferronato, Dr.
Co Orientador
Instituto Federal do Paraná

PAULO ROBERTO CORDOVA: 00830004963
Digitally signed by PAULO ROBERTO CORDOVA:00830004963
DN: CN=PAULO ROBERTO CORDOVA:00830004963, OU=IFSC - Instituto Federal de Santa Catarina, O=ICPEdu, C=BR
Reason: I am approving this document
Location: your signing location here
Date: 2025.08.07 18:44:10-03'00'
Foxit PhantomPDF Version: 10.1.1

Prof. Paulo Roberto Cordova, Dr
Instituto Federal de Santa Catarina

gov.br Documento assinado digitalmente
RENAN VINICIUS ARANHA
Data: 07/08/2025 16:04:51-0300
Verifique em https://validar.iti.gov.br

Prof. Renan Vinicius Aranha, Dr
Universidade Federal de Mato Grosso

AGRADECIMENTOS

Nem Sistemas de Informação, nem Caçador foram a minha primeira escolha, portanto eu não imaginava que esses últimos 4 anos seriam assim.

Os meus primeiros agradecimentos vão para os primeiros professores que tive contato: o professor Samuel que viu potencial em mim, me acolheu e me ensinou a gostar de Programação Orientada a Objetos e o professor Cristiano que contribuiu para a minha descoberta no mundo da programação *web*, na qual vi meu possível futuro como desenvolvedora.

Gostaria de agradecer a minha mãe, que é também a minha melhor amiga e sempre me incentivou, apoiou e enxergou coisas em mim que eu não consigo enxergar. Ao meu pai, no qual voltei a ter mais contato nos últimos anos e que sempre incentivou a educação e o trabalho duro. A todos os meus familiares, os que estão aqui, mas que também estão longe, por todo apoio e carinho ao longo dos anos. Principalmente a minha vó e meus tios que tive a oportunidade de ter mais contato estando aqui. Agradeço ao meu falecido avô, do qual não pude me despedir em vida, que sonhava em ver minha formatura. Meu avô, que foi professor de letras, sempre incentivou a leitura e os estudos.

Às minhas amigas, Maria Clara, uma longa amizade de mais de 10 anos. Agradeço por me ajudar a enxergar potencial nesse trabalho e por todo apoio incondicional e por ser essa pessoa com esse coração tão grande. Suiane, que conheci em Caçador, que me faz rir em todas as horas e que se tornou um dos pilares para eu não colapsar em diversos momentos. E Duda, que me incentivou muito e ouviu meus desabafos na época em que moramos juntas.

À minha orientadora Taynara e ao meu coorientador Jair, obrigada por toda orientação e parceria nesse 1 ano de trabalho juntos, tenho certeza que me ajudou tanto na minha evolução como acadêmica, como profissional e como pessoa.

Obrigada a todos que de alguma forma participaram da minha existência nesses anos de faculdade, foi realmente uma experiência que eu jamais imaginaria que me moldaria de tantas formas, e por isso, sou eternamente grata.

A inclusão acontece quando se aprende
com as diferenças e não com as igualdades
(Paulo Freire, 1998)

RESUMO

As Tecnologias da Informação e Comunicação (TIC) tornaram-se fundamentais na contemporaneidade, funcionando como facilitadoras de uma vasta gama de atividades. Contudo, pessoas com deficiência frequentemente enfrentam maiores dificuldades para utilizá-las de maneira satisfatória. Embora existam diversas ferramentas e diretrizes de acessibilidade disponíveis na *Web*, muitos sistemas ainda apresentam carência nesse aspecto. Para abordar essa lacuna, o desenvolvimento deste trabalho seguiu uma metodologia baseada no modelo de *software* Cascata, com etapas sequenciais bem definidas. Isso garantiu uma abordagem estruturada desde a concepção. Este trabalho objetivou a criação de um sistema *web* voltado à organização e categorização de diretrizes de acessibilidade. O propósito é capacitar desenvolvedores a conhecerem e aplicarem essas diretrizes em seus projetos, conforme as necessidades do público-alvo com deficiência. Ao propor uma ferramenta para auxiliar na acessibilidade digital, tornou-se imprescindível que o próprio sistema fosse acessível, com aproximadamente 22 diretrizes selecionadas e implementadas. Assim, o sistema desenvolvido não somente auxilia na organização e validação de diretrizes essenciais para projetos de sistemas *web* acessíveis, contribuindo para a disseminação da acessibilidade digital, mas também se destaca por ser inerentemente acessível a um público mais amplo de usuários.

Palavras-Chave: acessibilidade digital; acessibilidade digital na web; sistema web; sistema web acessível.

ABSTRACT

Information and Communication Technologies (ICT) have become fundamental in contemporary times, functioning as facilitators of a wide range of activities. However, people with disabilities often face greater difficulties in using them satisfactorily. Although there are several accessibility tools and guidelines available on the web, many systems still lack accessibility in this regard. To address this gap, the development of this work followed a methodology based on the Cascade software model, with well-defined sequential steps. This ensured a structured approach from conception. This work aimed to create a web system focused on organizing and categorizing accessibility guidelines. The purpose is to enable developers to understand and apply these guidelines in their projects, according to the needs of their target audience with disabilities. When proposing a tool to assist with digital accessibility, it became imperative that the system itself be accessible, with approximately 22 guidelines selected and implemented. It is concluded that the developed system not only helps in the organization and validation of essential guidelines for accessible web systems projects, contributing to the dissemination of digital accessibility, but also stands out for being inherently accessible to a wider audience of users.

Keywords: digital accessibility; web digital accessibility; web system; accessible web system.

LISTA DE ILUSTRAÇÕES

LISTA DE ABREVIATURAS E SIGLAS

API	*Application Programming Interface*
ARIA	*Accessible Rich Internet Applications*
ATAG	*Authoring Tool Acessibility Guidelines*
CLI	*Command-Line Interface*
CRUD	*Create, Read, Update, Delete*
CSS	*Cascading Style Sheets*
DTO	*Data Transfer Object*
eMAG	Modelo de Acessibilidade em Governo Eletrônico
ER	Entidade-Relacionamento
E2E	*End-2-End*
FK	*Foreign Key*
GPS	*Global Positioning System*
HTML	*HyperText Markup Language*
HTTP	*Hypertext Transfer Protocol*
IHC	Interação Humano-Computador
JWT	*JSON Web Tokens*
MDN	*Mozilla Developer Network*
NVDA	*NonVisual Desktop Access*
ORM	*Object-Relational Mapping*
PcD	Pessoa com Deficiência
PNAD	Pesquisa Nacional por Amostra de Domicílios
PK	*Primary Key*
REST	*Representational State Transfer*
Sass	*Syntactically Awesome Style Sheets*
SQL	*Structured Query Language*
SWEBOK	*Software Engineering Body of Knowledge*
TA	Tecnologia Assistiva
TDD	*Technical Design Document*
TEA	Transtorno do Espectro Autista
TIC	Tecnologia da Informação e Comunicação
UAAG	*User Agent Acessibility Guidelines*
UI	*User Interface*

UML	*Unified Modeling Language*
URI	*Uniform Resource Identifier*
UX	*User Experience*
WAI	*Web Accessibility Initiative*
WCAG	*Web Content Accessibility Guidelines*
WWW	*World Wide Web*
W3C	*World Wide Web Consortium*
XML	*eXtensible Markup Language*

SUMÁRIO

1 INTRODUÇÃO

As Tecnologias da Informação e Comunicação (TICs) consolidaram-se como um dos pilares centrais da sociedade moderna. Com o uso de dispositivos móveis e computadores, os indivíduos podem realizar tarefas de forma mais eficiente, acessar serviços como GPS, efetuar compras online, participar de plataformas de ensino à distância, comunicar-se instantaneamente por meio de aplicativos de mensagens e explorar uma infinidade de informações disponíveis na internet. Além disso, as TICs desempenham um papel fundamental na promoção da inclusão digital, elas conectam pessoas em áreas remotas e ampliam sua participação na esfera social, econômica e cultural (Ragnedda; Ruiu; Addeo, 2022; Roza, 2020).

Entretanto, pessoas com deficiência frequentemente enfrentam maiores dificuldades para se inserir plenamente nesse contexto digital. Conforme dados da Pesquisa Nacional por Amostra de Domicílios Contínua (PNAD Contínua) de 2022, estima-se que o Brasil tenha cerca de 18,6 milhões de pessoas com deficiência (IBGE, 2023). Esse número ressalta a importância de desenvolver sistemas acessíveis.

A acessibilidade envolve a garantia de condições para que todas as pessoas possam acessar, com segurança e autonomia, espaços, serviços, informações e direitos (Brasil, 2015). Com base nessa perspectiva legal, ela deve ser compreendida como um princípio fundamental para a promoção de ambientes inclusivos e igualitários, assegurando que barreiras físicas, comunicacionais ou atitudinais, sejam progressivamente eliminadas da sociedade. Ela deve ser também garantida no ambiente digital. Essa busca pela equidade no uso das tecnologias é conhecida como acessibilidade digital, e visa assegurar que todos, independentemente de suas condições físicas ou sensoriais, possam usufruir das TICs de maneira plena e inclusiva (Ferreira Sobrinho Junior, Sonza e Fernandes, 2024; Fraz *et al*., 2021; Soler; Farina; Florian, 2021).

Nesse contexto, este trabalho visa desenvolver um sistema *web* que permitirá a desenvolvedores de *software* e outros interessados realizar uma triagem das diretrizes de acessibilidade relevantes para seus projetos de desenvolvimento de sistemas *web* acessíveis. Também, organizar essas diretrizes para promover uma melhor compreensão e disseminação das práticas de acessibilidade, o que contribui para a criação de soluções mais inclusivas. Para isso, tem-se como base a página

web intitulada *AcessibiWeb*[1] que foi resultado do artigo de Souza e Dutra (2024).

1.1 Justificativa

Dado o crescente impacto das tecnologias digitais no cotidiano, é essencial garantir que essas ferramentas sejam inclusivas e acessíveis para mais pessoas. Com base no *AcessibiWeb* (Souza; Dutra, 2024), o presente trabalho visou expandir a ideia da página *web* que reuniu diretrizes a partir de uma revisão sistemática da literatura. Essas diretrizes foram categorizadas nas diferentes deficiências, tais quais auditiva, visual, motora, cognitiva e neural, por meio de breves descrições em cartões interativos. Além das deficiências dispostas no Estatuto (Brasil, 2015), o Transtorno do Espectro Autista (TEA) também foi incluído (Souza; Dutra, 2024).

A proposta de expansão consiste em transformar uma página *web* informativa, que atualmente apresenta diretrizes de acessibilidade categorizadas, em um sistema *web* interativo com recursos de acessibilidade. Esse sistema permitirá que os usuários cadastrem seus projetos e associem a eles as diretrizes mais relevantes para seu contexto. Assim, não se trata de uma ferramenta que implementa e avalia automaticamente a acessibilidade, mas de um meio estruturado para os usuários terem maior controle e visibilidade sobre os requisitos necessários para tornar seus projetos mais acessíveis.

1.2 Definição do problema

Embora existam diversas diretrizes de acessibilidade disponíveis, desenvolvedores de *software* frequentemente enfrentam dificuldades para identificá-las, compreendê-las e aplicá-las de maneira eficaz em seus projetos (Souza; Dutra, 2024).

Atualmente, há várias ferramentas que auxiliam na implementação da acessibilidade digital na *Web*, desde verificadores de contraste de cores, como *WebAIM Color Contrast Checker* e *Contrast Ratio*, até extensões de navegador e bibliotecas específicas para *frameworks front-end*, como *Vue A11y*, *React Aria* e *Inclusive Components*. Além disso, há outros *sites* que centralizam as diretrizes da WCAG, como o Guia WCAG[2]. No entanto, ferramentas intuitivas para a triagem e

[1] A página *AcessibiWeb* pode ser acessada no seguinte link: https://acessibiweb.github.io/
[2] O Guia WCAG pode ser acessado no seguinte link: https://guia-wcag.com/en/

organização dessas diretrizes ainda são raras. Apesar da disponibilidade de recursos, a adesão às práticas de acessibilidade continua baixa, o que resulta em interfaces digitais que não atendem adequadamente às necessidades das pessoas com deficiência.

A questão de pesquisa que norteia este trabalho é: Como pode ser estruturada uma plataforma digital acessível e intuitiva para centralizar e organizar diretrizes de acessibilidade digital na *Web*, visando facilitar o acesso, a compreensão e a aplicação dessas diretrizes por desenvolvedores e demais interessados?

1.3 Objetivos

Esta seção detalha os objetivos específicos, que guiam as etapas para alcançar o objetivo geral deste trabalho.

1.3.1 Objetivo geral

Desenvolver uma plataforma digital acessível e intuitiva que centralize e organize as diretrizes de acessibilidade digital na *Web*, visando facilitar seu acesso, compreensão e aplicação por desenvolvedores e demais interessados, por meio de etapas bem delimitadas.

1.3.2 Objetivos específicos

a) Levantar requisitos funcionais e não funcionais, incluindo as diretrizes de acessibilidade digital, do sistema;
b) Modelar o sistema por meio de diagramas conforme Bourque e Fairley (2014);
c) Prototipar as interfaces do sistema e seus elementos de acessibilidade;
d) Implementar um sistema *web* acessível que siga as diretrizes de acessibilidade digital escolhidas;
e) Implementar testes funcionais, conforme Bourque e Fairley (2014), para avaliar requisitos do sistema.

2 FUNDAMENTAÇÃO TEÓRICA

Nesta seção serão apresentados os fundamentos teóricos que dão embasamento para o projeto. São abordados os conceitos de exclusão e inclusão social, acessibilidade digital e, principalmente, acessibilidade digital na *Web* que sustenta e aborda as diretrizes para a criação de um sistema *web* acessível, além das principais tecnologias e ferramentas a serem utilizadas.

2.1 Inclusão e a sociedade

O processo de exclusão social pode ser dividido em três dimensões: precarização do trabalho, precarização da sociabilidade primária e estigma. O indivíduo classificado em uma ou duas dimensões é visto no âmbito da vulnerabilidade social; ao se enquadrar nas três, já pode ser considerada uma exclusão (Costa; Ianni, 2018).

A primeira dimensão visa analisar a relação do indivíduo com o trabalho. O trabalho constitui um dos pilares essenciais para a criação de identidade e pertencimento de indivíduos em grupos sociais e, portanto, na sociedade. No novo cenário mundial, altamente globalizado e com uma intensa velocidade na inovação tecnológica, permeado pelas bases de produtividade e competitividade, faz-se necessário que os indivíduos se reinventem de forma contínua para acompanhar a rápida evolução e atualização profissional. Quem consegue se qualificar constantemente possui um privilégio sobre quem não consegue (Costa; Ianni, 2018).

A segunda dimensão foca nas relações sociais. Visto que a economia muda a percepção e comportamento em sociedade, em conjunto com a primeira dimensão, a instabilidade nas relações sociais pode ser ainda maior. O mundo está muito mais individualizado, os indivíduos estão cada vez mais em busca de realizações pessoais. Assim, as relações se tornam objetos de consumo, são frágeis, descartáveis e de pouca duração (Costa; Ianni, 2018).

A última se baseia no estigma. O estigma é construído por meio da categorização dos indivíduos. Indivíduos com determinadas características são reunidos em grupos considerados diferentes do comum. Nada mais é do que um estereótipo. Ao invés do indivíduo ser visto pelas suas características próprias, ele é taxado por características que visam desprezar sua imagem. O estigma dificulta

facilmente as relações sociais e a inserção no mercado de trabalho, o que desemboca também no isolamento social (Costa; Ianni, 2018).

As três dimensões, que individualmente e em conjunto, colaboram para dificultar as relações sociais, podem levar ao que Coelho e Sousa (2024) chamam de desvinculação, no qual pode ser compreendido também como afastamento. Isso corrobora para o indivíduo perder progressivamente seus direitos em sociedade. Em concordância com Costa e Ianni (2018), o indivíduo necessita da noção de pertencimento, seja pelos vínculos sociais, bem como seu papel na sociedade. Determinados estigmas que geram essa desvinculação, fazem com que o indivíduo não se veja como útil ou indispensável no convívio social. Portanto, para Coelho e Sousa (2024), a exclusão está intrinsecamente relacionada às relações de um indivíduo com os demais, em que gradativamente o indivíduo é marginalizado e deixado de lado, o que cria mais rupturas em diversos setores da vida em sociedade.

A inclusão social vem como um processo para contrapor esse contexto, e nesse sentido, é necessário entender a causa e gerar estratégias para alteração ou aniquilamento desse estado (Costa; Ianni, 2018). É o resultado de uma transformação social. Ao trabalhar com as pessoas e não para as pessoas, promove-se capacitação, participação, solidariedade, cooperação, dimensão reflexiva, e conscientização (Coelho; Sousa, 2024).

Além disso, conforme levantam Sanfelice e Bassani (2020), essa inclusão faz parte de um conjunto de aspectos que atingem o que pode ser chamado de qualidade de vida. Trata-se de recursos para garantir uma boa saúde física e mental, como: água potável, saneamento básico, alimentos de qualidade, moradia, contato com o meio-ambiente, interações sociais satisfatórias, empregabilidade, acesso a médicos e medicamentos, independência, atividades de lazer que façam sentido para ela, acesso a bens, serviços etc. (Sanfelice; Bassani, 2020). Assim, como acesso à informação e tecnologia, ao considerar o cenário tecnológico da sociedade.

No cenário tecnológico da sociedade, um aspecto que surge e se torna um dos pilares dentre eles, é o acesso à informação e tecnologia. Esse acesso é fundamental para garantir a qualidade de vida e reduzir as desigualdades, o que reforça a importância da inclusão digital, além da inclusão social, na construção de uma sociedade mais equitativa (Carmo; Duarte; Gomes, 2020; Ragnedda; Ruiu; Addeo, 2022).

2.1.1 Sociedade digital

As Tecnologias da Informação e Comunicação (TICs) são introduzidas como um meio para a troca de informação que utiliza a comunicação digital. Para Roza (2020), a informação é algo que perpassa gerações desde os tempos antigos; porém, o que caracteriza essa nova era é a natureza digital. Ela pode ser dividida em ciberespaço e cibercultura. O primeiro são todos os dispositivos físicos necessários para a conexão ser possível, bem como os indivíduos que os utilizam. Cria-se, portanto, uma rede. A segunda é toda a troca intelectual que ocorre nesse meio, ou seja, valores, crenças, culturas, conhecimento etc.

Nesse meio, para ocorrer a inclusão digital, são necessários infraestrutura e conhecimento para o uso das TICs, visando a autoexpressão e a criatividade.

No entanto, a inclusão plena vai além do simples acesso e uso de tais tecnologias. Roza (2020), pontua que há uma diferença entre acesso à informação e acesso ao conhecimento. Para o conhecimento ser efetivamente acessado, é preciso, primeiramente, que haja a assimilação da informação. Esse processo depende diretamente da aprendizagem e, parcialmente, das experiências vivenciadas pelo indivíduo. Em outras palavras, “se utilizadas de maneira adequada, as TICs possuem a capacidade de transformar a aprendizagem e aprimorar a mente dos indivíduos” (Roza, 2020, p. 73). Portanto, as TICs por si só não garantem a obtenção de conhecimento, ou formação de entendimento.

Nesse contexto de complexidade da inclusão digital, ela se tornou um direito social como saúde e educação. Isso demonstra a relevância das TICs na sociedade (Carmo; Duarte; Gomes, 2020). O indivíduo que estiver desconexo do mundo digital estará, de certa forma, desconexo do mundo globalizado e da sociedade em si, o que leva à exclusão social. Em uma relação de mão dupla, essa exclusão social também se manifesta como exclusão digital, conforme pontuam Amadeu, Silva e Manochio-Pina (2022). Para esses autores, a exclusão digital ocorre quando os indivíduos não possuem acesso a aparelhos tecnológicos que permitem a comunicação no meio digital ou não conseguem utilizá-los, seja por questões de baixo aprendizado, econômicas etc., incluindo-se também os sistemas (*softwares*) a eles associados.

Desse modo, a exclusão digital atua como um amplificador da exclusão

social, uma vez que a inserção no ambiente digital proporciona inúmeras oportunidades na nova era da sociedade. Aqui se destaca a inclusão digital como o acesso a equipamentos e a competências como informação, comunicação, colaboração, elaboração de conteúdo e acesso à criatividade, segurança e resolução de problemas. Os benefícios dessa inclusão vão desde engajamento político a cultural (Ragnedda; Ruiu; Addeo, 2022).

Dentro desse panorama da inclusão digital, a garantia da participação de pessoas com deficiência (PcD) é um aspecto crucial para garantir que todos, independentemente de suas condições físicas, sensoriais ou cognitivas, possam participar plenamente da sociedade digitalizada. Para que isso ocorra, os recursos digitais, sejam eles aplicativos, *websites* e plataformas de ensino, precisam ser desenvolvidos com acessibilidade. A acessibilidade digital não é somente uma questão técnica, mas também ética e social, ao promover a inclusão de pessoas com deficiência, ampliando suas possibilidades de interação social, educação, trabalho e lazer no ambiente digital. Assim, ao fomentar a inclusão digital, a sociedade dá um passo importante para a inclusão social dessas pessoas, proporcionando-lhes uma maior autonomia e participação no mundo contemporâneo.

2.1.2 Inclusão da pessoa com deficiência

Segundo a Lei n.º 13.146 de 6 de julho de 2015:

> Considera-se pessoa com deficiência aquela que tem impedimento de longo prazo de natureza física, mental, intelectual ou sensorial, o qual, em interação com uma ou mais barreiras, pode obstruir sua participação plena e efetiva na sociedade em igualdade de condições com as demais pessoas (Brasil, 2015).

Conforme o capítulo 2, Artigo n.º 4 (Brasil, 2015), “Toda pessoa com deficiência tem direito à igualdade de oportunidades como as demais pessoas e não sofrerá nenhuma espécie de discriminação”. Considera-se discriminação:

> Toda forma de distinção, restrição ou exclusão, por ação ou omissão, que tenha o propósito ou o efeito de prejudicar, impedir ou anular o reconhecimento, ou o exercício dos direitos e das liberdades fundamentais de pessoa com deficiência, incluindo a recusa de adaptações razoáveis e de fornecimento de tecnologias assistivas (Brasil, 2015,

A inclusão das pessoas com deficiência (PcD) é essencial não somente para garantir o exercício pleno de seus direitos, mas também para reconhecer o potencial de cada indivíduo e proporcionar oportunidades para seu desenvolvimento. Muitas vezes, a sociedade se concentra nas limitações impostas pela deficiência (Rangel, 2023), e esquece das capacidades, habilidades e talentos que esses indivíduos podem oferecer. A inclusão social vai além da mera adaptação do ambiente; trata-se de transformar as estruturas sociais, educacionais e de trabalho para que todos, independentemente de suas condições, possam participar ativamente e contribuir de maneira significativa. Reconhecer o potencial das pessoas com deficiência e proporcionar as condições adequadas para seu desenvolvimento é fundamental para a construção de uma sociedade mais justa, equitativa e inclusiva, onde a diversidade é valorizada como uma fonte de enriquecimento coletivo.

2.2 Acessibilidade digital

O artigo 3 da Lei n.º 13.146 do Estatuto da Pessoa com Deficiência define acessibilidade como:

> Possibilidade e condição de alcance para utilização, com segurança e autonomia, de espaços, mobiliários, equipamentos urbanos, edificações, transportes, informação e comunicação, inclusive seus sistemas e tecnologias, bem como de outros serviços e instalações abertos ao público, de uso público ou privados de uso coletivo, tanto na zona urbana como na rural, por pessoa com deficiência ou com mobilidade reduzida (Brasil, 2015).

De acordo com Almeida e Cardoso (2023), a acessibilidade está dividida em 6 tipos: arquitetônica, comunicacional, metodológica, instrumental, programática e atitudinal.

Nesse sentido, o intuito da acessibilidade é garantir que não haverá barreiras para locomoção, transmissão de mensagens e comunicação, aprendizagem, uso de qualquer tipo de instrumento que auxilie na realização de tarefas e atividades. A ausência de mecanismos e ferramentas adequadas, bem como ambientes igualitários que promovam a autonomia do indivíduo, prejudicam a participação, principalmente das pessoas com deficiência. Isso compromete tanto a acessibilidade como a efetividade na utilização de sistemas e dispositivos (Fraz *et al.*, 2021).

Nesse contexto, acessibilidade digital é o meio para que as pessoas com

deficiência consigam utilizar *sites* e aplicativos de forma autônoma, sendo primordial para qualquer usuário, independentemente de sua necessidade específica, poder acessar informações e realizar tarefas da forma mais clara e confortável possível (Soler; Farina; Florian, 2021). A acessibilidade digital não só promove benefícios para pessoas com deficiência, mas também para outros grupos mais vulneráveis, como pessoas idosas.

Para Ferreira Sobrinho Junior, Sonza e Fernandes (2024), a acessibilidade digital deve garantir que todos os indivíduos possam acessar, compreender e interagir com o computador e seus recursos. Ainda mais que, como abordado por Barbosa et al. (2021), na grande maioria das vezes um sistema é construído com a mentalidade “dentro para fora” dando ênfase na representação e processamento de dados, arquitetura, entre outras coisas que irão garantir o funcionamento dele. Porém, não há um enfoque para o que está externo ao sistema, que é como ele será utilizado. Assim, é como se o desenvolvimento tivesse o pressuposto de que o externo irá se adaptar ao sistema sem esforço. Em vista disso, a acessibilidade agrega usabilidade e comunicabilidade quando se trata da qualidade no uso de sistemas. Esses conceitos juntos visam garantir o uso em igualdade de condições, com facilidade de navegação, linguagem simples, clara, direta e de fácil compreensão (Ferreira Sobrinho Junior; Sonza; Fernandes, 2024).

Um *software* ou *hardware* interativo não pode impor barreiras para o usuário em sua utilização. Se houver alguma barreira, ele não conseguirá desfrutar do apoio computacional que o sistema ou dispositivo oferece (Barbosa *et al.*, 2021). Um sistema no qual pessoas com deficiência irão utilizar, é necessário que a acessibilidade seja pensada desde o início do planejamento do projeto.

2.3 Acessibilidade digital na *Web*

Consoante a pesquisa realizada pela Big-DataCorp no ano de 2021, foram analisados 16,89 milhões de *sites* ativos no Brasil, menos de 1% obtiveram sucesso nos testes de acessibilidade (Forbes, 2021).

A acessibilidade na *Web* garante que as pessoas com deficiência também possam usufruir dos recursos e informações disponíveis na internet, podendo perceber, entender, navegar, interagir e contribuir para a *Web* (Soler; Farina; Florian, 2021; Guerra; Carneiro; Santarosa, 2022).

Bastos *et al.* (2023) ainda enfatiza que há sete princípios focados na *Web* para que o usuário com deficiência consiga navegar por ela sem a necessidade de ajuda externa, em que não haja barreiras inviabilizando a realização de tarefas. São eles: uso igualitário, uso flexível, uso simples e intuitivo, informação perceptível, tolerância a falhas, baixo esforço físico e dimensão e espaço para uso e interação.

No universo dos recursos acessíveis via navegadores, a acessibilidade se dá por meio de diretrizes que guiam a construção de conteúdos acessíveis. Há diversas diretrizes disponíveis para implementação que serão abordadas nas próximas seções.

2.3.1 WCAG

A *World Wide Web Consortium* (W3C) desenvolve padrões e diretrizes para auxiliar na criação de uma *Web* baseada em princípios como acessibilidade, internacionalização, privacidade e segurança (World Wide Web Consortium, 2024). Elaborou três guias focados na acessibilidade *Web*: *Web Content Accessibility Guidelines* (WCAG), *Authoring Tool Accessibility Guidelines* (ATAG) e *User Agent Accessibility Guidelines* (UAAG) (Bastos *et al.*, 2023).

Para Bastos *et al.* (2023), cada uma delas apresenta as seguintes razões:

- WCAG: tornar o conteúdo disponível na *Web* acessível para pessoas com deficiência.
- ATAG: ferramentas que auxiliam na criação de conteúdos acessíveis, focadas em desenvolvedores.
- UAAG: como tornar acessível um documento para pessoas com diferentes condições motoras, linguísticas e sensoriais.

Conforme a Caldwell *et al.* (2008), um ambiente será considerado acessível se for perceptível, operável, compreensível e robusto. A perceptibilidade é quando um usuário que está utilizando uma interface consegue perceber os componentes que estão na tela. A operabilidade refere-se à operação dos componentes do sistema, portanto, o usuário precisa conseguir navegar por eles e utilizá-los. Compreensibilidade diz respeito à compreensão do conteúdo com o qual o usuário interage, e consome. Por fim, a robustez deve garantir que o conteúdo possa ser interpretado de forma confiável por uma variedade ampla de agentes de usuário, incluindo tecnologias assistivas (TAs) (Caldwell *et al.*, 2008).

Divididos em 4 grupos, esses princípios garantem a satisfação do usuário, o que pode ter o efeito contrário se um ou mais deles não forem implementados. Portanto, se o *site* for de difícil utilização, o usuário irá desistir de acessá-lo (Guerra; Carneiro; Santarosa, 2022).

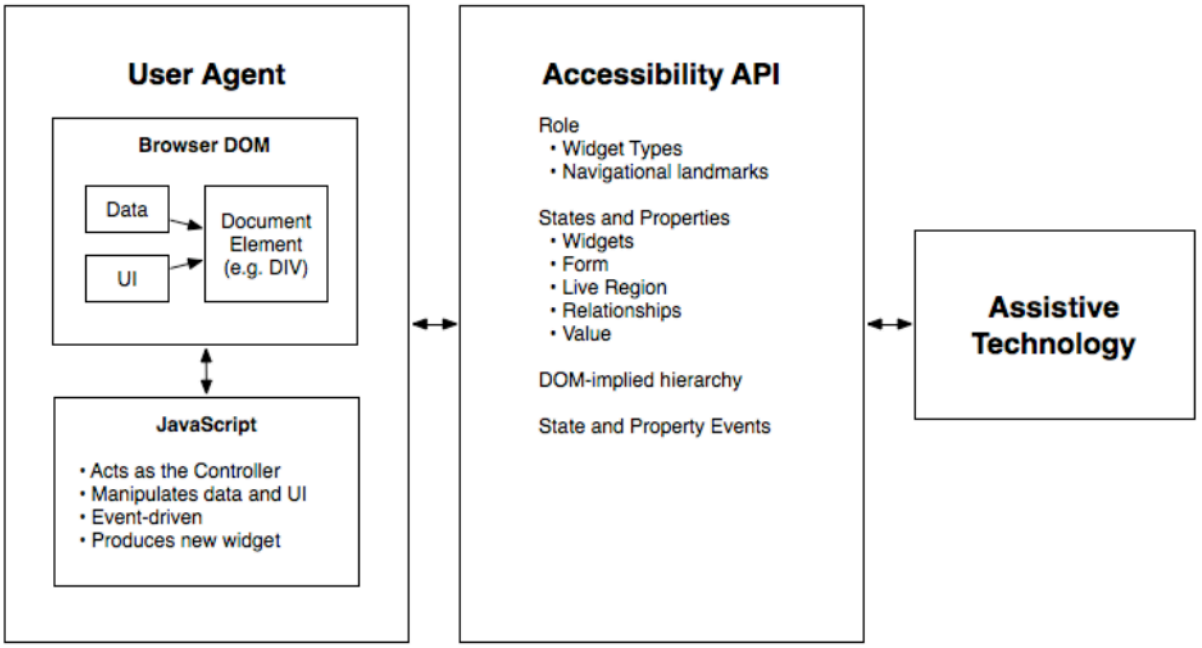
Para cada um desses princípios há um conjunto de diretrizes a serem seguidas (Guerra; Carneiro; Santarosa, 2022). Como exemplo, pode-se citar a garantia do acesso a todas as funcionalidades do sistema por teclado, bem como, o sistema provê maneiras de auxiliar o usuário a navegar, encontrar conteúdo e saber em qual parte do sistema está[3].

Além das diretrizes, a WCAG utiliza também critérios para determinar o sucesso de cada uma delas. Para cada diretriz, três níveis de conformidade são utilizados: A (mais baixo), AA e AAA (mais alto) (Caldwell *et al.*, 2008).

Em conjunto com a WCAG, o *Accessible Rich Internet Applications* (WAI-ARIA), do português Aplicações Web Ricas e Acessíveis (Diggs *et al.*, 2023) ajuda a criar elementos mais acessíveis no HTML, auxiliando principalmente Tecnologias Assistivas (TAs) utilizadas por pessoas com deficiência. O WAI-ARIA traz semanticidade para elementos não semânticos, agregando contexto e significado. Diggs *et al.* (2023) traz um diagrama explicativo que retrata o funcionamento da tecnologia assistiva com o conteúdo *web* por meio de uma API de acessibilidade (Figura 1).

[3] As diretrizes da WCAG podem ser consultadas no seguinte link: https://www.w3.org/TR/2008/REC-WCAG20-200812/

Figura 1 — Modelo de contrato com APIs de acessibilidade



Fonte: W3C (2023).

A WAI-ARIA (Diggs *et al.*, 2023) define *role* como o papel que um *widget* irá desempenhar, no qual *widget* será o objeto de interação do usuário. Esses papéis possuem diferentes categorias, sendo elas *Abstract*, *Widget*, *Document Structure*, *Landmark*, *Live Region* e *Window*.

- *Abstract*: papéis nessa categoria não devem ser utilizados no HTML, eles são utilizados para dar suporte aos modelos do WAI-ARIA com o propósito de definir conceitos gerais dos papéis disponíveis.
- *Widget*: representam componentes padrões da interface, podem ser tanto independentes como fazerem parte de outros *widgets*.
- *Document Structure*: descrevem estruturas para organização do conteúdo em uma página, geralmente não são interativos.
- *Landmark*: o *landmark* é uma região da página onde o usuário necessita possivelmente de acesso rápido a ela. Os seguintes propósitos se enquadram: navegação, pesquisa, conteúdo primário.
- *Live Region*: *live regions* são regiões de uma página nas quais são geralmente atualizadas por meio do resultado de um evento externo, em que o foco do usuário está em um local diferente. Porém, nem sempre elas são atualizadas a partir de uma interação do usuário. Ao se tratar de TAs como leitores de tela, geralmente elas não têm conhecimento dessa área ou não

conseguem processar a informação para o usuário. Nisso, o WAI-ARIA disponibiliza uma série de propriedades para o usuário conseguir identificar essas regiões.

- *Window*: serve como uma janela no escopo do *browser* (navegador) ou da aplicação.

Além dos *roles,* há também *states (estados)* e *properties (propriedades).* Eles se referem a natureza do papel de um objeto, provendo informações específicas.

Os estados e propriedades são categorizados em: *Widget*, *Live Region*, *Drag-and-Drop* e *Relationship.*

- *Widget*: são utilizados com elementos comumente achados em interfaces de usuário na qual recebem entradas de usuários e processam ações de usuários. Esses atributos (estados e propriedades) são utilizados para dar suporte a papéis *Widget*. Abaixo segue uma breve explicação de atributos desse tipo no qual serão possivelmente utilizados no sistema:
    - *aria-errormessage*: nele é indicado o elemento no qual possui a mensagem de erro, portanto pode ser utilizado o identificador do elemento. Conforme Diggs *et al.* (2023), ele **precisa** ser utilizado em conjunto com *aria-invalid*. Quando um elemento válido tem essa propriedade, é recomendado que o elemento que possua a mensagem de erro esteja escondido, assim ela não será pertinente. No caso de o elemento ser inválido, o elemento que possui a mensagem **precisa** ser mostrado na tela, para o usuário conseguir navegar até ele e examinar. É aconselhável utilizar *live regions* para anunciar para os usuários quando eles entrarem com um valor inválido no sistema. Ao utilizar uma mensagem de erro, é necessário indicar o que está errado e o que é necessário informar.
    - *aria-hidden*: indica se o elemento estará exposto para uma API de acessibilidade (utilizadas por TA). Esse atributo precisa ser utilizado cautelosamente, somente em cenários no qual a sua remoção melhora a experiência do usuário, em que o elemento possa indicar conteúdo redundante ou que não indica um propósito claro. É importante que ao utilizar *aria-hidden* seja indicada a TA algo idêntico ou com significado e funcionalidade equivalentes.
    - *aria-label*: serve como um rótulo para um elemento. TAs irão buscar um

label pelo nome acessível do elemento. No caso de um botão ou um *link,* seus textos já servem como o nome acessível. Porém, em alguns cenários é necessário definir um nome acessível por um texto alternativo (invisível).

- *aria-invalid*: indica se o elemento é inválido, em casos de o valor de entrada não ser no formato esperado pelo sistema.
- *aria-haspopup*: um *popup* é um elemento na página que aparece acima de outro/s elemento/s. Esse atributo indica se um elemento irá abrir um *popup* na tela e qual seu tipo, nesse caso, deve ser **garantido** que o elemento *popup* seja do tipo menu, *listbox*, *tree*, *grid* ou *dialog*. E, em *aria-haspopup*, esse tipo deve ser indicado. Para que o *popup* seja acessível por comando de teclado, é necessário que o elemento que abre o *popup* seja focalizado, havendo um comando que abre o *popup* e que o *popup* controle o foco de todos os seus elementos descendentes. Se não há um indicativo visual de que o elemento irá abrir um *popup* é indicado repensar a utilização deste atributo.
- *aria-expanded*: utilizado com um elemento que irá controlar a visibilidade de outro elemento, indicando uma mudança entre dois estados (*toggle*). No caso informa se o elemento está estendido (visível) ou colapsado (não visível). Se o elemento que está sendo controlado não estiver dentro (descendente) do elemento que o controla, é necessário utilizar o atributo *aria-controls* no elemento que controla a visibilidade.
- *aria-modal*: indica se o elemento é um *modal*. O *modal* é um elemento no qual limita o usuário a interagir somente com ele, até que perca o foco ou não seja mais mostrado em tela. No caso de haver um botão para fechar o *modal*, esse botão deve ser um descendente desse *modal.* É indicado que os outros elementos fora do *modal* sejam deixados “inertes”.
- *aria-pressed*: utilizado com *toggle buttons*, botões que indicam mudança de estado, geralmente entre dois estados (*true* e *false*). O *aria-pressed* indica qual o atual estado do botão. Se esse atributo não estiver presente, o botão não é considerado um *toggle button*.

- *Live Region*: esses atributos são utilizados com *live regions* para indicar

quando um conteúdo muda sem o elemento estar com foco, provendo informações para TAs para como processar o conteúdo atualizado. Os atributos disponíveis são:

  - *aria-atomic*: por padrão, TAs irão anunciar mudanças em uma *live region* somente do local que sofreu a mudança. Com o uso de *aria-atomic*, elas apresentam todo o conteúdo da *live region*.
  - *aria-busy*: esse atributo indica para a TA que ela precisa esperar até que as mudanças sejam completas para então anunciar para o usuário.
  - *aria-live*: quando uma mudança ocorre, o usuário pode receber essa informação no momento do evento ou posteriormente. Para isso, o *aria-live* pode ser indicado como *polite* ou *assertive,* portanto, ele serve como um indicador de prioridade ao anunciar uma mudança. *Assertive* irá interromper qualquer atividade atual que o usuário estiver realizando, pois se entende que é um anúncio imediato. Já o *polite* irá aguardar um momento em que a tarefa foi pausada para fazer o anúncio.
  - *aria-relevant*: seus possíveis valores são *additions*, *removals*, *text* ou *all*. *Additions* indica notificação de elementos adicionados. *Removals* para elementos removidos. *Text* para mudanças em textos. *All* para todos. Por padrão, o único elemento não utilizado nesse atributo é *removals*, que deve ser utilizado somente se realmente necessário.

- *Drag-and-Drop*: utilizados com elementos *drag-and-drop* (arrastar e soltar).
- *Relationship*: esses atributos indicam relação ou associação entre elementos no qual não pode ser determinado pela estrutura do documento. Alguns dos possíveis a serem utilizados:
  - *aria-activedescendant*: quando utilizado elementos compostos, tais quais *combobox*, *textbox*, *group* ou *application* o *aria-activedescendant* indica o elemento ativo atual sem remover o foco do componente pai. O exemplo utilizando *combobox*, é quando o usuário possui um elemento de *input* e ao digitar algo é apresentado a uma lista de itens que podem ser focáveis, porém, não se deseja perder o foco do input, caso queira continuar digitando. Mesmo com esse foco no elemento pai, o usuário ainda consegue gerenciar o foco nos elementos filhos. O

elemento que estiver com esse atributo **precisa** estar visível na tela.

- *aria-controls*: indica qual elemento ou elementos são controlados pelo elemento que possui *aria-controls*.
- *aria-describedby*: uma descrição permite trazer mais informações sobre um elemento. É possível utilizar o *id* de um elemento com uma descrição em diversos outros elementos ou em um grupo de elementos.
- *aria-labelledby*: similar ao *aria-describedby*, porém, ao invés de prover uma descrição, provê um rótulo. Inclusive, se houver um elemento visível que serve como rótulo deve ser utilizado ao invés de *aria-label*, onde a TA sempre irá priorizar o *aria-labelledby* com relação ao *aria-label*.

2.3.2 e-MAG

O e-MAG tem a intenção de garantir a acessibilidade em portais e *sites* eletrônicos de administração pública (Brasil, 2014).

> O eMAG foi desenvolvido em 2004 baseado no estudo de 14 normas existentes em outros países acerca da acessibilidade digital. Dentre as normas analisadas, estavam a Section 508 do governo dos Estados Unidos, os padrões CLF do Canadá, as diretrizes irlandesas de acessibilidade e documentos de outros países como Portugal e Espanha. Também foi realizada uma análise detalhada das regras e pontos de verificação do órgão internacional WAI/W3C, presentes na WCAG 1.0 (Brasil, 2014)

Ainda segundo Brasil (2014), às recomendações feitas no modelo entram de acordo tanto com os padrões internacionais como com as necessidades específicas brasileiras. Portanto, é uma versão especializada da WCAG voltada para o contexto brasileiro.

Além dessas diretrizes, há diversas outras disponíveis na literatura.

2.3.3 Literatura

O artigo de Souza e Dutra (2024), que serve como fundamento para a construção deste trabalho, conseguiu reunir um conjunto de diretrizes de acessibilidade com base em revisão sistemática.

Essas diretrizes foram divididas pelas deficiências — auditiva, visual, motora, cognitiva e neural, e TEA —, e reunidas com o intuito de simplificar o acesso a elas, e dar suporte principalmente a desenvolvedores *web*, mas também a qualquer pessoa interessada em acessibilidade digital voltada para a *Web*, com foco em *websites* mais inclusivos e acessíveis para usuários com deficiência (Souza; Dutra, 2024).

As diretrizes foram disponibilizadas em uma página *web* estática, a qual foi transformada em um sistema *web* acessível para auxiliar os usuários na implementação de diretrizes em seus projetos.

2.3.4 Aspectos Técnicos e Validação

Esta seção apresenta os diagramas, tecnologias e estratégias de teste que fundamentaram o desenvolvimento do sistema.

***Unified Modeling Language* (UML):** os diagramas UML são artefatos que auxiliam na modelagem de um sistema, de forma abstraída. Essa modelagem pode ser feita tanto de forma gráfica quanto formal. No caso da modelagem gráfica, a abordagem que será seguida neste projeto, a linguagem geralmente utilizada é a *Unified Modeling Language* (UML) (Sommerville, 2011).

Diagrama de Caso de Uso (comportamental): um caso de uso é uma técnica que identifica atores que realizarão determinadas interações, elas sendo descritas sucintamente por meio de texto ou outras representações gráficas. Ao primeiro realizar o levantamento de requisitos do sistema, é possível representar por meio do Diagrama de Caso de Uso todas as possíveis interações descritas nos requisitos (Sommerville, 2011).

Diagrama de Classes (estrutural): se no Diagrama de Caso de Uso há a representação dos possíveis cenários, no Diagrama de Classes se tem a representação dos objetos que serão manipulados nessas interações, chamados de classes (Pressman; Maxim, 2021). O conceito objeto/classe é apresentado na Orientação a Objetos, um paradigma presente no desenvolvimento de sistemas. Uma classe na Orientação a Objetos serve como molde para a criação de objetos que possuem certas características (atributos) e certas funções que podem ser realizadas (métodos) (Carvalho, 2016). Desse modo, suponha que um sistema possua uma classe pessoa, e essa pessoa possui um nome e uma idade, e a ação

que ela pode desempenhar nesse sistema é fazer aniversário. A partir dessa classe pode-se criar diferentes pessoas (objetos) e gerar nomes e idades específicas de cada uma. A ação será a mesma, porém os resultados serão referentes a cada objeto.

Diagrama Entidade-relacionamento (informacional): Parreira Júnior (2010) define o ER também como Modelo de Dados. O ER se assimila ao Diagrama de Classes, porém é voltado para uma abordagem mais focada na construção do banco de dados, considerando o tipo escolhido.

HTML, CSS e JavaScript: HTML, CSS e *JavaScript* são as tecnologias pilares necessárias para a construção de conteúdos *web*. A *Mozilla Developer Network* (MDN) (Mozilla Foundation, 2024) desde 2005 documenta sobre essas e outras tecnologias do universo do desenvolvimento *web* para facilitar o compartilhamento de conhecimento entre interessados. Conforme à Mozilla Foundation (2024) pode-se definir HTML, CSS e *JavaScript* como:

- *HyperText Markup Language* (HTML): define a estrutura e sentido do conteúdo *web*. *HyperText* em português, Hipertexto, se refere a conexão de páginas, sejam elas do mesmo ou de diferentes *websites*, na *Web*. Quando se trata da acessibilidade na *Web*, é importante garantir um código HTML semântico para que leitores de tela e outros recursos de TAs interpretem a estrutura HTML (Maria, 2020).
- *Cascading Style Sheets* (CSS): é uma linguagem de estilo para descrever como um documento será apresentado na *Web*. Não necessariamente esse documento precisa ser escrito em HTML, outras tecnologias como XML também podem ser utilizadas.
- *JavaScript*: é uma linguagem de programação conhecida como linguagem de *script* para páginas *web*. Ela consegue ser interpretada tanto em navegadores como em servidores ao utilizar tecnologias como *Node.js*. Ela permite a criação de dinamicidade e interações com os elementos da página.

Com base nisso, as próximas seções abordarão as tecnologias que serão utilizadas para a construção do sistema *web* que é o enfoque deste projeto.

TypeScript: Baumgartner (2020) define *TypeScript* como sendo uma camada adicional para se trabalhar em cima de *JavaScript*. No fim, todo o código *TypeScript* se tornará *JavaScript*, mas é uma poderosa ferramenta para incluir funcionalidades que a linguagem *JavaScript* por si só não fornece. A principal funcionalidade pela

qual *TypeScript* é amplamente utilizado em projetos *web* modernos, além da familiaridade com *JavaScript*, é a tipagem. Tipagem no mundo da programação indica o tipo do dado no qual o programa está trabalhando. O *TypeScript* adiciona uma camada de tipagem, e consegue transformar *JavaScript* em uma ferramenta mais segura, que garante uma integridade maior dos dados e que também consegue lidar melhor com erros.

Next.js: *Next.js* é um *framework* baseado em outra tecnologia chamada *React.js*. (Vercel, 2024). *React.js* é uma biblioteca que permite a utilização de *TypeScript* para a criação de interfaces. Ela é baseada em componentes, os quais são considerados um pedaço da interface. Cada componente tem sua própria lógica e aparência, podendo ir desde um simples botão na tela até a tela toda (Meta Platforms, 2024).

Nest.js: *Nest.js* é um *framework* moderno para a construção de aplicações do lado do servidor, desenvolvido com *TypeScript* e fortemente inspirado em conceitos do *Angular*. Ele oferece uma arquitetura modular e orientada a componentes, facilitando o desenvolvimento de aplicações escaláveis, testáveis, e de fácil manutenção (Mysliwiec, 2024).

PostgreSQL: *PostgreSQL* é um banco de dados relacional *open source*. Possui diversas funcionalidades que auxiliam na criação de aplicações, proteção da integridade dos dados e criação de ambientes tolerantes a falhas, além de auxiliar a gerenciar os dados, independentemente do tamanho do conjunto de dados. Ao se tratar de um banco de dados relacional, tem base na ideia de tabelas e utiliza a linguagem SQL para manuseio dos dados nelas (The PostgreSQL Global Development Group, 2024).

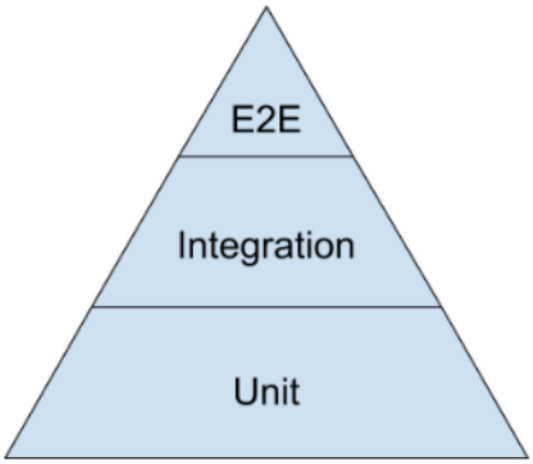
TypeORM: ORM é uma técnica para facilitar a interação entre dados de um banco relacional e linguagens de programação orientadas a objetos. Portanto, é criado um objeto virtual do banco de dados que pode ser facilmente utilizado com a linguagem de programação. A *TypeORM* é comumente utilizada em projetos que utilizam o *Nest.js* (TypeORM, 2024).

Testes: de acordo com Bourque e Fairley (2014) pode-se realizar testes de diferentes níveis em um sistema durante os processos de desenvolvimento e manutenção. Os níveis são segundo o objeto de teste (alvo) ou o propósito (objetivo). Para a etapa de testes, optou-se por testes com base no alvo. Para testar o alvo foram escolhidos testes unitários e de integração.

Testes unitários auxiliam na testagem de funções ou classes (unidades ou as menores partes de um sistema) isoladamente, para garantir que uma unidade do sistema faz o que se espera dela (Bourque; Fairley, 2014; Valente, 2020). Já os testes de integração verificam o funcionamento das unidades em conjunto (Valente, 2020).

Juntamente com estes testes, há também os testes *end-2-end* (E2E), que visam testar a aplicação de ponta a ponta, também são chamados de Testes de Sistema, conforme Valente (2020). Na Figura 2, Wacker (2015) traz a Pirâmide de Testes adaptada para os testes unitários, de integração e E2E. Os testes unitários representam a maior parcela ao se tratar de quantidade de testes, isto porque, são menos custosos e mais rápidos de implementar. Já os testes E2E caracterizam uma parcela pequena, mas são igualmente importantes (Wacker, 2015; Valente, 2020).

Figura 2 — Pirâmide de testes



Fonte: Wacker (2015).

Jest: *Jest* é um *framework* de testes para *JavaScript* que possibilita a criação de testes unitários e de integração, no qual auxilia na automatização do processo de testagem do sistema. Com a *Command-Line Interface* (CLI) também existe a possibilidade de gerar relatórios que indicam a cobertura de testes (The OpenJS Foundation, 2024).

3 METODOLOGIA

Nesta seção serão apresentadas as metodologias de pesquisa e do projeto que guiaram a construção deste trabalho.

3.1 Metodologia de pesquisa

Quanto à natureza dessa pesquisa, pode-se considerar a Pesquisa Aplicada, definida por Silva e Menezes (2001). A Pesquisa Aplicada tem como intuito a solução de algum problema em específico, se propondo a resolvê-lo, buscando atender a uma demanda social.

Referente à abordagem do problema, essa pesquisa se enquadra em quantitativa, pois será realizado um levantamento da cobertura de testes unitários e de integração.

Com relação aos objetivos, essa é uma Pesquisa Exploratória-Descritiva, visto que consiste na realização de uma pesquisa bibliográfica em livros, publicações periódicas, impressos diversos etc, e depois, tem-se a busca por entender o problema ou fenômeno (Gil, 2002). Para o embasamento desta pesquisa, foi realizada uma pesquisa bibliográfica aprofundada nos principais conceitos: exclusão/inclusão social, acessibilidade e acessibilidade digital na *Web*. E depois, descreveu-se o processo de desenvolvimento de um produto tecnológico.

Segundo os procedimentos técnicos estabelecidos por Gil (2002), esta pesquisa pode ser classificada como bibliográfica e pesquisa-ação. A pesquisa-ação se caracteriza pela participação ativa do pesquisador no processo investigativo, sendo, ao mesmo tempo, sujeito e agente da pesquisa. Nesse contexto, a autora deste trabalho atuou diretamente no desenvolvimento de melhorias para um projeto já existente, embasando-se tanto na identificação do problema quanto na implementação de soluções para sua otimização.

Com foco na pesquisa em Ciência da Computação, conforme classificação definida por Wazlawick (2009), tem-se o estilo “Apresentação de algo diferente”. Para o autor, é quando o trabalho se propõe a resolver um problema de forma diferente, realizando uma comparação de técnicas, baseando-se em hipóteses. Com base na análise da página *AcessibiWeb* (Souza; Dutra, 2024) e da necessidade de aprimoramento de uma página estática para um sistema *web* acessível, buscou-se criar algo novo a partir dela.

3.2 Metodologia de projeto

Este trabalho foi desenvolvido com base na adoção do modelo de processo de software Cascata, em razão de sua estrutura sequencial e clara definição de etapas, que se adequa a projetos com requisitos bem definidos e menor probabilidade de alterações significativas. Esse modelo prevê uma única entrega final, após a conclusão de etapas sequenciais bem delimitadas conforme necessidade do projeto em questão. As etapas do modelo Cascata definidas foram: levantamento de requisitos, planejamento, modelagem, desenvolvimento, testes e implantação.

Na etapa inicial, utilizou-se técnica de engenharia reversa para levantar os requisitos do sistema. Analisou-se a página *AcessibiWeb* (Souza; Dutra, 2024) por meio de suas interfaces já em produção, visando identificar pontos a serem aprimorados e novas funcionalidades necessárias para atender às demandas do sistema. Esse processo permitiu compreender melhor as lacunas presentes na página estática atual. Além disso, foram selecionadas diretrizes de acessibilidade para serem incluídas no sistema, com o intuito de torná-lo acessível. Para isso, buscou-se analisar as diretrizes disponibilizadas pela WCAG e pelo próprio *AcessibiWeb*, em que, na etapa de prototipação já com as interfaces encaminhadas foi possível identificar quais elementos de acessibilidade seriam necessários e viáveis de serem implementados com o tempo disponível de desenvolvimento.

Em seguida, no planejamento, foram definidas as principais tecnologias a serem utilizadas durante o desenvolvimento do sistema. Essa escolha incluiu a seleção de uma linguagem de programação adequada para a *Web*, *frameworks* modernos e um banco de dados eficiente.

Na etapa de modelagem, foram elaborados os diagramas de Caso de Uso, Classes e Entidade-Relacionamento (ER). Esses diagramas foram desenvolvidos com base nos requisitos elicitados, o que proporciona uma base sólida para a etapa de desenvolvimento.

Anterior a etapa de desenvolvimento, foram realizadas prototipações das interfaces do sistema com base nos requisitos levantados, para assim ter uma melhor noção dos fluxos de interação dos usuários com o sistema e quanto aos recursos de acessibilidade. Foi feita tanto uma prototipação de baixo nível quanto

uma de alto nível, que se assemelha ao produto final.

O nome escolhido para o sistema foi *AcessiWeb*, ao invés do nome original *AcessibiWeb*. Ainda não havia sido revelado o novo Símbolo Internacional de Acessibilidade, porém, tendo ciência da troca e da aprovação de projeto pela substituição dos símbolos pelo Senado (Agência Senado, 2025) e que essa mudança irá ocorrer também para o ambiente digital, será futuramente modificado.

Para o desenvolvimento do sistema, foi criado um *Technical Design Document* (TDD)[4], do português Documento de Especificação Técnica do Sistema, para realizar um levantamento mais detalhado do processo de desenvolvimento. Como, definição da arquitetura, design de todas as partes do sistema (*front-end*, *back-end* e banco de dados), especificações dos modelos de dados com relação ao banco escolhido, que foi o *PostgreSQL*. Com relação à camada da API, foram levantadas também as dependências necessárias, tratamento de erros, fluxo de autenticação, especificações de paginação, e especificações das rotas. Com relação à camada de *front-end* também foi realizado o levantamento de dependências e aspectos dos recursos de acessibilidade. Ainda, algumas considerações adicionais como segurança.

Na etapa de testes, o sistema foi submetido a um conjunto de verificações para garantir a qualidade e integridade de suas funcionalidades. Para isso, foram utilizados testes automatizados, no caso de testes unitários que permitiram testar funções do sistema e testes de integração para verificar o funcionamento dos serviços em conjunto com o banco de dados.

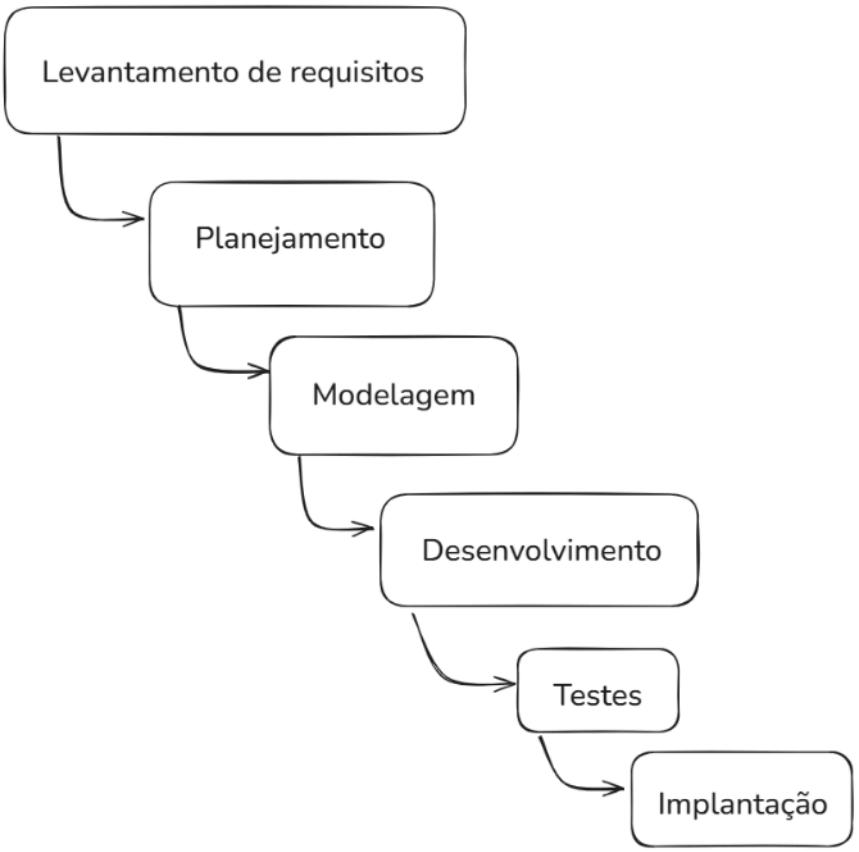
Por fim, a última etapa envolve a implantação do sistema, que foi hospedado em um ambiente de produção para garantir sua disponibilidade e acessibilidade às partes externas. Essa fase inclui a configuração de servidores e a publicação do sistema, garantindo que ele esteja preparado para atender às demandas dos usuários finais de forma eficiente e confiável.

A Figura 3 mostra as etapas sequenciais definidas para este projeto.

---

[4] O TDD pode ser acessado por meio do link: https://kind-parent-217.notion.site/TDD-1b1fcc794298803a9903faa23e46a487

Figura 3 — Etapas da metodologia de projeto



Fonte: elaborado pela autora (2025).

4 ANÁLISE E DISCUSSÃO DOS RESULTADOS

Os resultados alcançados neste projeto baseiam-se na aplicação do modelo de processo de software Cascata, que prevê etapas sequenciais bem definidas e estruturadas. As etapas realizadas foram Levantamento de Requisitos, Planejamento, Modelagem e Prototipação, Desenvolvimento, Testes e Implantação.

4.1 Requisitos

Os seguintes requisitos foram levantados para nortear o desenvolvimento do sistema:

4.1.1 Requisitos funcionais

a) Acesso de Usuários (visitante): O usuário que não estiver logado será considerado um usuário visitante, que poderá visualizar as diretrizes cadastradas, porém não poderá criar uma solicitação de diretriz ou gerenciar um projeto. Para tal, será necessário criar uma conta no sistema ou, se já possuir uma conta, realizar login. A tela de criação de conta conterá campos de (NOME DE USUÁRIO, EMAIL/CELULAR, SENHA E CONFIRMAÇÃO DE SENHA) e ela poderá ser acessada a partir da tela de login, na qual também conterá uma opção para redefinição de conta, caso por algum motivo o usuário esqueceu sua senha ou não possui mais acesso ao e-mail/celular.
b) Acesso de Usuários (comum): O usuário comum precisará estar logado no sistema para gerenciar seus projetos e suas solicitações.
c) Acesso de Usuários (*admin*): O usuário administrador possuirá uma tela de login diferente do usuário comum, ela poderá ser acessada por meio de um *endpoint* diferente no sistema. Para o usuário administrador não haverá as opções de criação e recuperação de conta, visto que essa inclusão/modificação pode ser feita diretamente pela API ou pelo banco de dados.
d) Gerenciamento de Diretrizes: O usuário comum ao acessar a tela inicial do sistema terá acesso a todas as diretrizes disponíveis cadastradas ou aceitas por um usuário administrador, organizadas por paginação. Ele poderá realizar a filtragem por (TEXTO ou DEFICIÊNCIA). Para cada diretriz terá um ícone

de carrinho para que assim ele possa montar seu projeto. O usuário administrador terá acesso às funcionalidades de incluir, editar ou excluir uma diretriz, também poderá utilizar os mesmos filtros. Além disso, o usuário comum poderá solicitar a criação de diretrizes, onde no menu ele também poderá acessar suas solicitações e filtrar por (TEXTO, DEFICIÊNCIA ou SITUAÇÃO). Para cada diretriz solicitada será mostrada a situação atual da solicitação. Ao acessar a solicitação, se houver, ele poderá também visualizar uma mensagem explicando o motivo da situação. O usuário administrador terá acesso a uma tela de solicitações, em que poderá verificar a solicitação, mudar a situação (APROVAR ou REJEITAR), que por padrão será em análise, e informar o motivo dela. Na tela de solicitações, o usuário administrador poderá filtrar por (SITUAÇÃO, DATA E HORA DA SOLICITAÇÃO ou DEFICIÊNCIA).

e) Gerenciamento de Projetos: O usuário comum poderá gerenciar seus projetos. Ao acessar o sistema, no menu ele terá a possibilidade de ir para a tela de projetos. Essa tela mostrará uma lista de projetos cadastrados pelo usuário, paginados. Ele poderá filtrar por (TEXTO ou DATA DE CRIAÇÃO). Nessa mesma tela haverá um botão para incluir um novo projeto e para cada projeto na lista haverá botões de edição e exclusão. Quando escolhida a opção de inclusão, a interface aberta será a do carrinho, que mostra as diretrizes selecionadas e possibilita a criação de um nome e descrição para o projeto.
f) Preferências: Todos os usuários (visitante, comum e administrador) poderão visualizar e editar suas preferências do sistema, que incluirão (TEMA, BRILHO, FONTE, TAMANHO DA FONTE, CONTRASTE, ESPAÇAMENTO ENTRE LINHAS, ESPAÇAMENTO ENTRE LETRAS, TAMANHO DO CURSOR e COR DO CURSOR).

4.1.2 Requisitos não funcionais

- Viabilizar Portabilidade: Por ser uma aplicação *web* funciona em qualquer sistema operacional, seja ele *Linux*, *Windows*, *MacOS* etc.
- Ser acessível: com base na WCAG e nas diretrizes identificadas no artigo de Souza e Dutra (2024), foram levantadas diretrizes a serem implementadas

para o sistema possuir recursos de acessibilidade. Foram elas:

- Texto descritivo como alternativa para elementos não-textuais;
- Não dependência de orientação de tela para qualquer funcionalidade;
- Inputs de formulários com propósitos claros;
- Uso de contraste;
- Uso de texto, cores e elementos;
- Utilização de comandos por teclado;
- Responsividade;
- Título em todas as páginas;
- Acesso de conteúdo por mais de um meio (menu, *search*…);
- Lógica sequencial e navegação intuitiva
- *Labels* e *headings* com propósitos claros;
- Mostrar ao usuário visualmente o ponto focal de onde ele está;
- Botões com descrições bem-definidas;
- *Breadcrumb*;
- Uso de WAI-ARIA;
- *Tooltips*;
- Definir o idioma da página;
- Cada item em uma lista precisa ter um valor único;
- Possibilidade de mais de um estilo de fonte, reajuste do tamanho do texto, brilho e contraste;
- Escrita por comando de voz;
- Leitor de tela;
- Teclado virtual

Esses requisitos foram elaborados com base em uma análise da página estática *AcessibiWeb* (Souza; Dutra, 2024), visando identificar pontos de melhoria e funcionalidades adicionais necessárias para o sistema proposto.

4.2 Planejamento

Durante a etapa de planejamento, foi definida a seguinte *stack* tecnológica, escolhida pelo tempo disponível para desenvolvimento, familiaridade e adequação aos objetivos do projeto:

- Linguagem de programação: *TypeScript*, pela sua tipagem estática e

camada de segurança para o desenvolvimento.

- *front-end*: *framework Next.js*, pela otimização de imagens, separação de componentes cliente e servidor, *middleware*, padronização, abstração para questões como *bundling* e *compiling* (o desenvolvedor foca em construir a aplicação), entre outros, o que auxilia na manutenção a longo prazo visto que não é somente uma biblioteca como *React.js,* mas sim um ecossistema (Vercel, 2024).
- *back-end*: *framework Nest.js*, escolhido por sua arquitetura modular e extensível. Neste projeto optou-se pela construção de uma API REST, sendo um padrão amplamente adotado na comunicação entre sistemas na *Web.* O *Nest.js*, além de suportar esse modelo, também oferece suporte nativo a outras abordagens de *back-end*, como microsserviços e GraphQL, reforçando sua versatilidade no desenvolvimento de *Application Programming Interfaces* (APIs), do português Interfaces de Programação de Aplicação (Mysliwiec, 2024).
- Banco de dados: *PostgreSQL*, um banco relacional amplamente adotado, integrando-se à biblioteca *TypeORM* para facilitar o mapeamento objeto-relacional. Ao escolher um banco de dados relacional, cria-se uma camada maior de organização, confiabilidade e integridade dos dados (The PostgreSQL Global Development Group, 2024). Além disso, é importante para a manutenção de um sistema no qual as entidades possuem relação entre si.

4.3 Modelagem

Na etapa de modelagem, utilizou-se a ferramenta online *Lucidchart* para a criação dos diagramas para representação estática, comportamental, estrutural e informacional do sistema. Os diagramas escolhidos foram o Diagrama de Caso de Uso, o Diagrama de Classes e o Diagrama Entidade-Relacionamento (ER), conforme descritos nas subseções seguintes.

4.3.1 Diagrama de Caso de Uso

O Diagrama de Caso de Uso mapeou três atores principais: usuário visitante, usuário logado (que no texto será referido como usuário comum) e usuário administrador. Cada ator desempenha diferentes ações no sistema:

O usuário visitante pode:

- Criar e recuperar uma conta;
- Visualizar e filtrar diretrizes;
- Fazer login

O usuário comum pode:

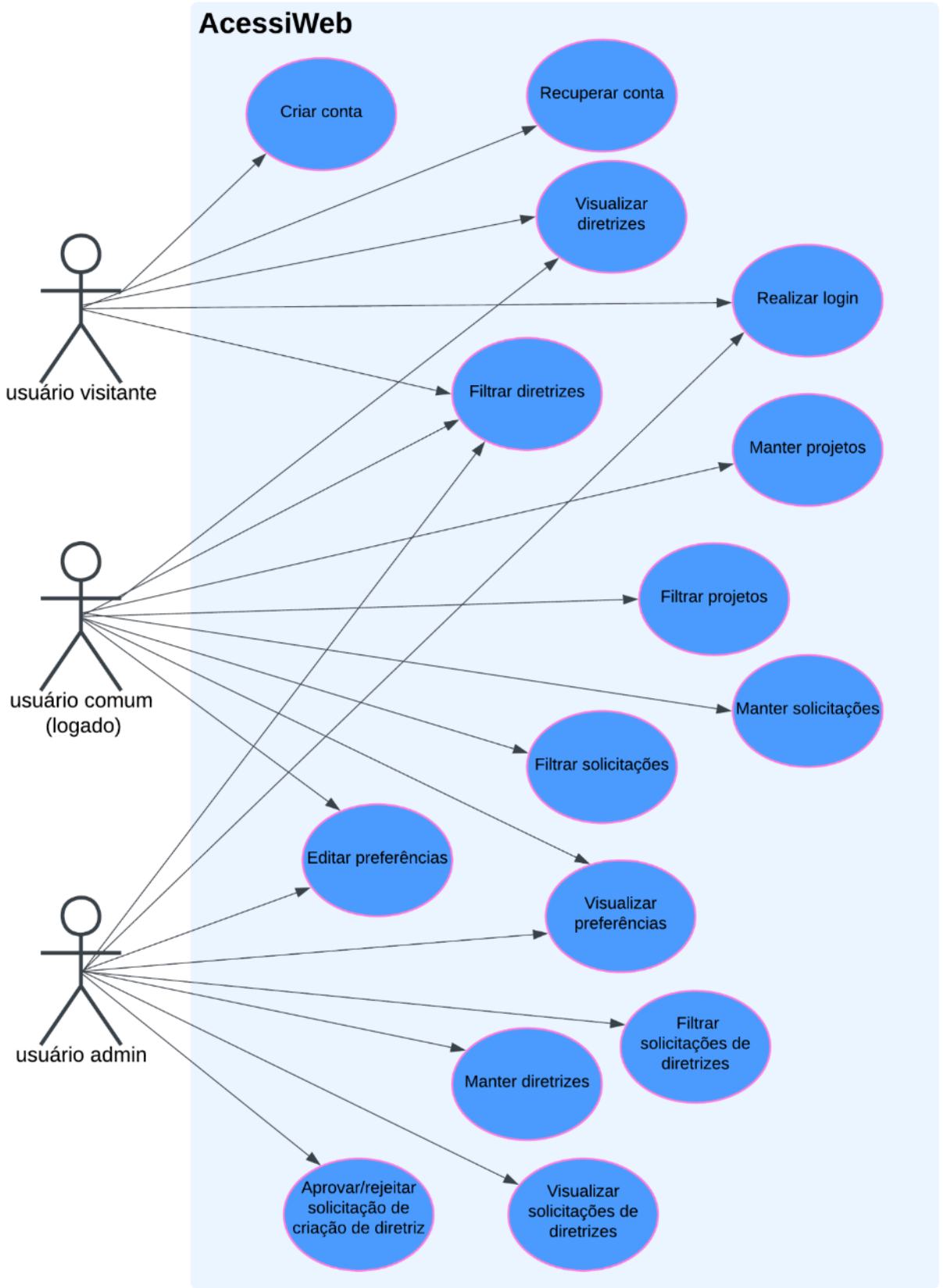
- Criar, visualizar, atualizar, remover e filtrar projetos;
- Criar e recuperar uma conta;
- Fazer login;
- Visualizar e filtrar diretrizes;
- Criar, visualizar, atualizar, remover e filtrar solicitações de diretrizes;
- Visualizar e editar preferências

O usuário administrador pode:

- Fazer login;
- Criar, visualizar, atualizar, remover e filtrar diretrizes;
- Visualizar, filtrar, aprovar ou rejeitar solicitações de usuários

A Figura 4 ilustra essas interações.

Figura 4 — Diagrama de Caso de Uso

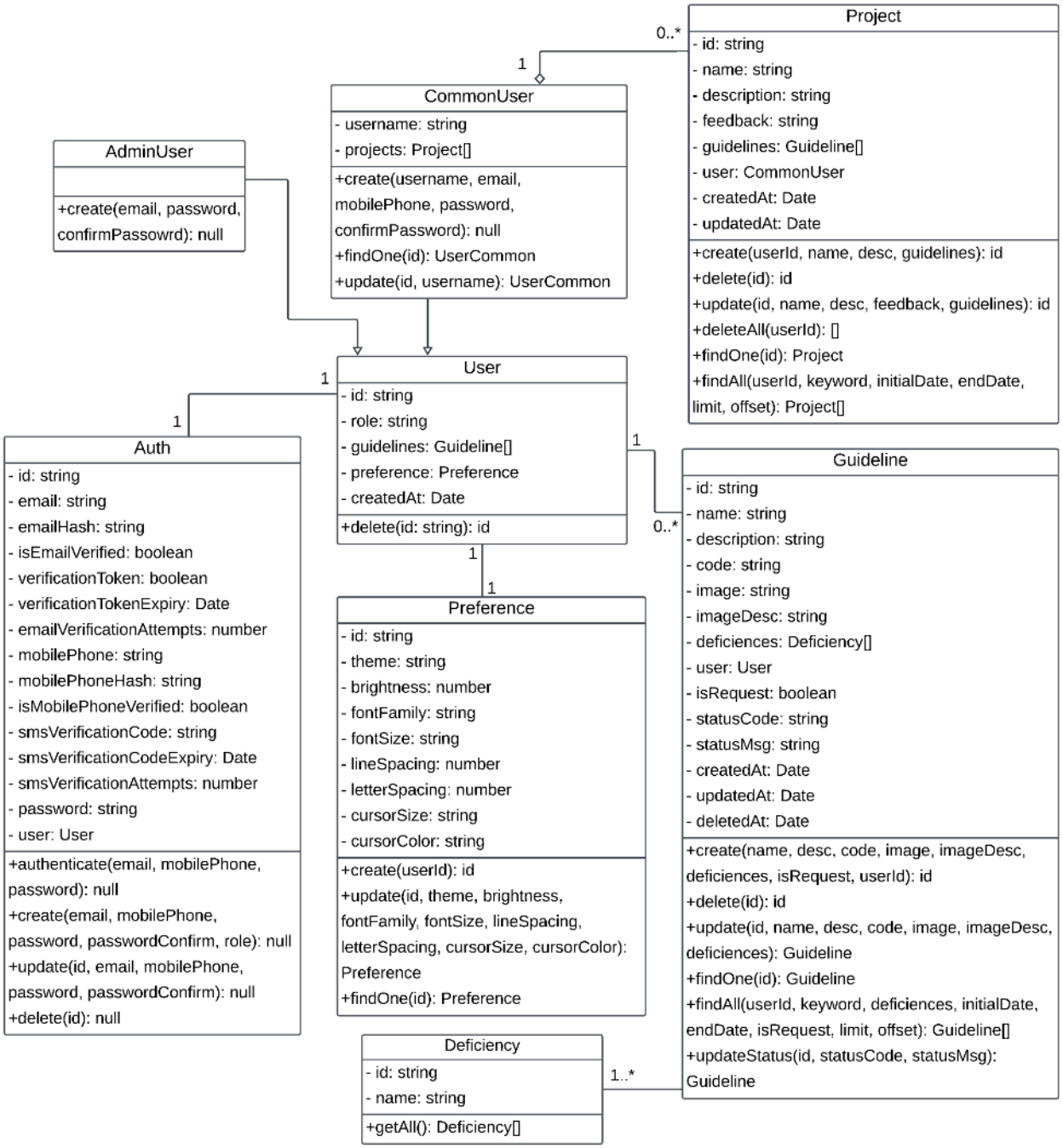


Fonte: elaborada pela autora (2025).

4.3.2 Diagrama de Classes

O Diagrama de Classes foi projetado com os seguintes objetos: *Auth*, *User*, *AdminUser*, *CommonUser*, *Project*, *Guideline*, *Deficiency* e *Preference*, como representado na Figura 5.

Figura 5 — Diagrama de Classes



Fonte: elaborado pela autora (2025).

Relações entre as classes:

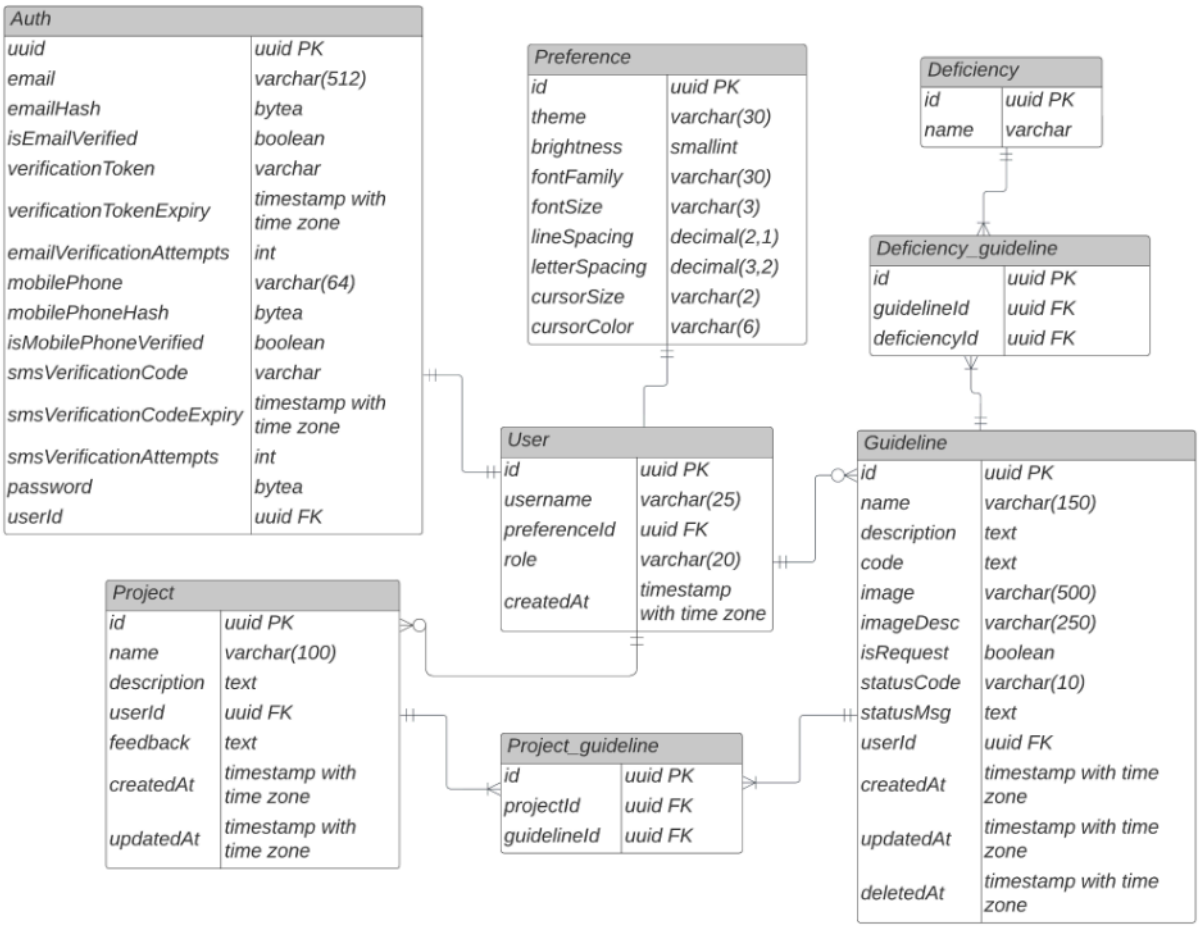
- A classe ***Auth*** possui uma relação com a classe ***User***, em que:
  - Um **usuário** estará relacionado a **uma autenticação**, e cada autenticação pertence a **um único usuário.**
- A classe ***User*** é abstrata e serve como base para as classes ***AdminUser*** e ***CommonUser***, que herdam suas propriedades e métodos.
- A classe ***Project*** possui uma relação de agregação com a classe ***CommonUser***, indicando que:
  - Um **usuário comum** pode criar **zero ou mais projetos**.
  - Cada **projeto** pertence a **um único usuário comum**.
- A classe ***Guideline*** possui uma relação com a classe ***User***, em que:
  - Um **usuário** pode criar **zero ou mais diretrizes**, mas cada diretriz está associada a **um único usuário**.
- A classe ***Deficiency*** possui uma relação com a classe ***Guideline***, onde:
  - Uma **diretriz** pode estar relacionada a **uma ou mais deficiências**.
  - Uma **deficiência** pode ser associada a **múltiplas diretrizes**.
- A classe ***Preference*** possui uma relação com a classe ***User***, na qual:
  - Um **usuário** terá um conjunto de **preferências**.
  - Cada conjunto de **preferências** pertence a **um único usuário**.

4.3.3 Diagrama Entidade-Relacionamento (ER)

O Diagrama Entidade-Relacionamento (ER) foi elaborado para representar a estrutura do banco de dados, destacando as tabelas e suas relações. Esse diagrama reflete os relacionamentos apresentados no Diagrama de Classes, com foco nos dados que serão armazenados.

A Figura 6 apresenta o diagrama ER do sistema.

Figura 6 — Diagrama Entidade-Relacionamento



Fonte: elaborado pela autora (2025).

4.4 Prototipação das interfaces

Além da modelagem, foram realizadas prototipações para mapear os fluxos de interação entre o usuário e o sistema. Primeiramente, foi feito um esboço inicial, que pode ser chamado também de prototipação de baixo nível. Posteriormente, com base nele, foi feita uma prototipação mais semelhante ao produto final, já buscando identificar cores, ícones, entre outros. Por fim, foi feita a prototipação de alto nível.

4.4.1 Prototipação de baixo nível

Com base nos requisitos funcionais levantados, foram criados os primeiros esboços de como seriam as interfaces do sistema, bem como seus fluxos essenciais. Pensou-se em um *header* (cabeçalho) com um menu para os usuários poderem navegar pelas páginas do sistema, diferenciando o que um usuário administrador e um usuário comum teriam acesso. Nesse esboço não foi

considerado o usuário visitante.

Para o usuário comum foram esboçadas as páginas:

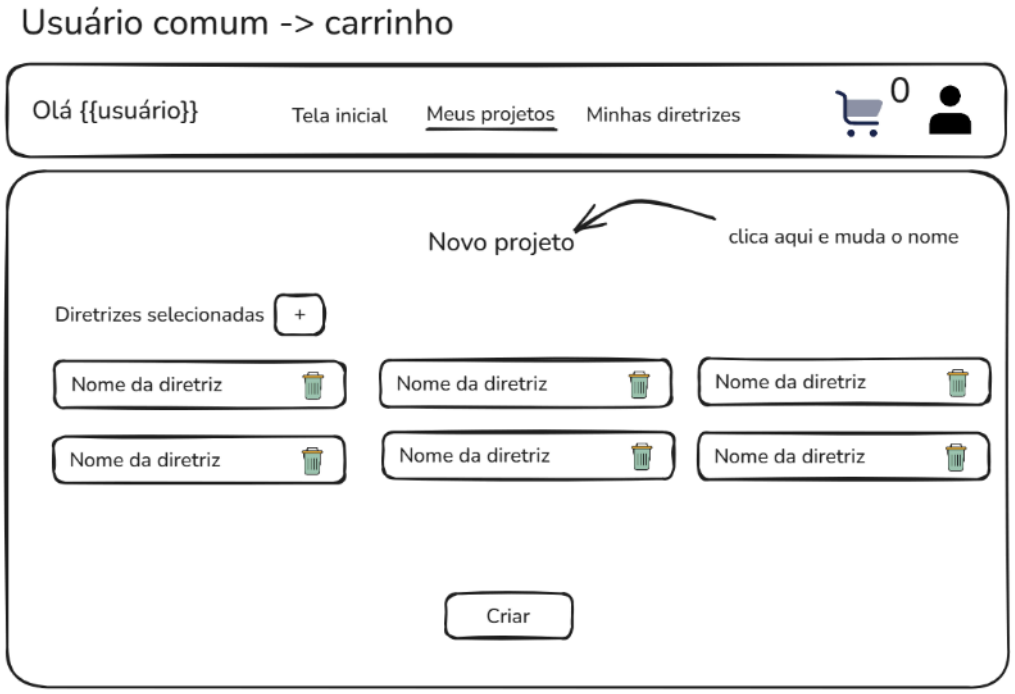
- *homepage* (página inicial): para acesso às diretrizes de acessibilidade cadastradas no sistema, com filtros por deficiência e texto. As diretrizes foram dispostas por meio de *cards* com a funcionalidade de inclusão de diretriz (no carrinho).
- projetos: para acesso aos projetos, com filtro por texto. Os projetos também foram dispostos por meio de *cards* com as funcionalidades de edição e remoção. Também, na página, foi colocado um botão para acesso ao carrinho, ou seja, a página de criação do projeto. Nesse caso, o usuário comum teria esse acesso tanto pelo botão quanto também pelo ícone de carrinho no *header.*
- solicitações: para acesso às solicitações de criação de diretriz. Uma página bastante semelhante à página de diretrizes, porém nos *cards* mostrando o *status* em conjunto com os botões de edição e remoção.
- carrinho: a página de carrinho é também a página para a criação do projeto, tendo sido pensada dessa forma para trazer mais facilidade na hora de incluir diretrizes. O usuário pode navegar tranquilamente por elas e escolher quais deseja para o seu projeto, sem precisar realizar muitos passos para chegar nesse objetivo.

Para o usuário administrador foram esboçadas as páginas:

- *homepage* (página inicial): para acesso às diretrizes de acessibilidade, com filtros por deficiência e texto. Diferentemente do usuário comum, as funcionalidades disponíveis por meio dos *cards* foram edição e remoção.
- solicitações: para acesso a todas as solicitações de usuários comuns. Foi esboçada também a página da solicitação, ao clicar em uma solicitação, para poder visualizar suas informações e aprová-la ou rejeitá-la.

Na Figura 7 há o esboço da tela de carrinho, em que o usuário comum pode escolher um nome para o seu projeto e visualizar todas as diretrizes adicionadas. Para cada diretriz tem a possibilidade de removê-la das selecionadas, tirando-a do projeto. O botão de mais serve como um atalho para voltar à tela inicial, mesmo tendo essa possibilidade também pelo menu principal.

Figura 7 — Esboço inicial da tela de carrinho (criação de projeto)



Fonte: elaborado pela autora (2024).

4.4.2 Prototipação *Figma*

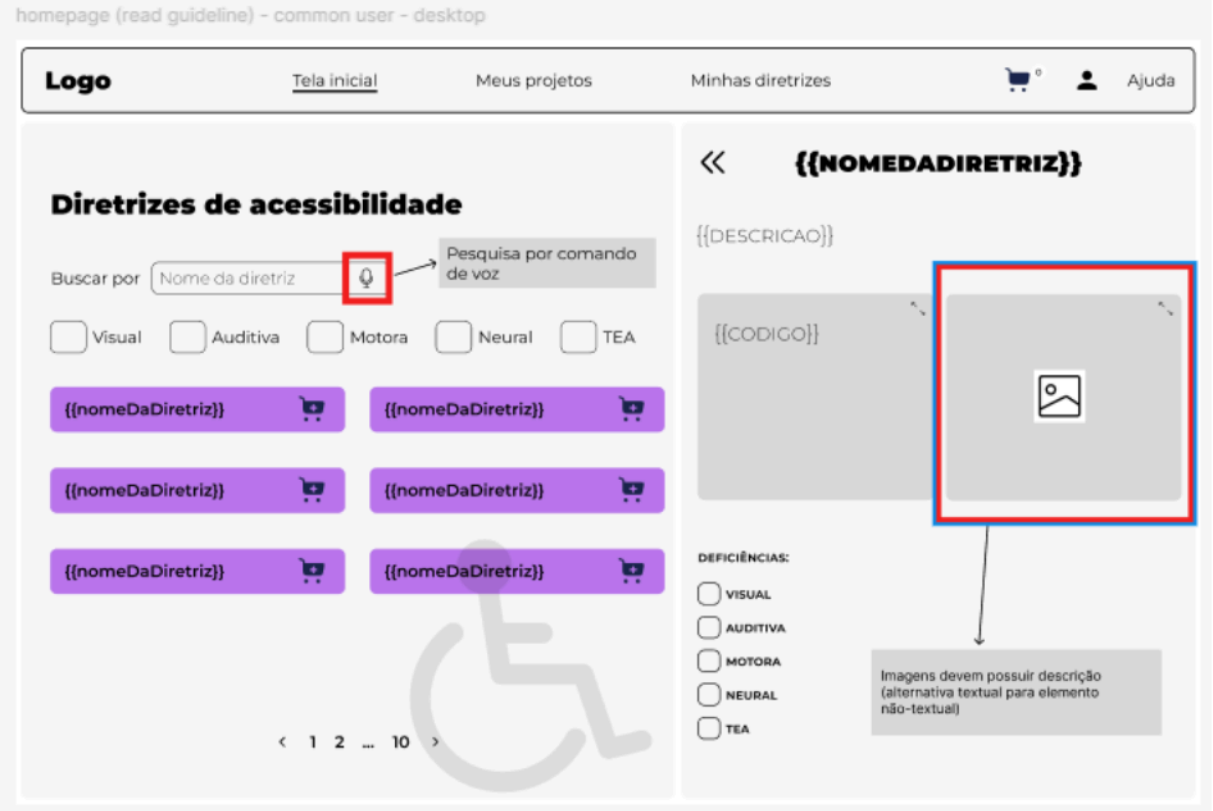
A segunda prototipação[5] foi realizada com a ferramenta *Figma*. Foram criadas as mesmas telas, porém, com um *design* pensado para o produto final, tendo em mente também quais cores seriam utilizadas, entre outros aspectos. Além das telas que já haviam sido pensadas, também foram incluídas as telas de autenticação (usuários visitantes e comuns) para acesso ao sistema.

Não somente questões como escolha dos ícones, tipografia, cores, espaçamentos, mas também, começou-se a estabelecer os aspectos de acessibilidade do sistema. Por meio dos requisitos não funcionais de acessibilidade levantados previamente, foi possível mapear no protótipo quais elementos e fluxos precisavam de acessibilidade.

Na Figura 8 é mostrada a prototipação da tela inicial para um usuário comum. Nela é possível observar a evolução do esboço inicial e dois aspectos de acessibilidade a serem implementados: pesquisa por comando de voz e descrição de imagens.

[5] Essa prototipação pode ser acessada no seguinte link: https://www.figma.com/design/574VUskfvvnZAQn6maxRsV/tcc-prot%C3%B3tipo-de-alto-n%C3%ADvel?node-id=0-1&t=EfAxafdXF8eu88UX-1

Figura 8 — Tela inicial para usuário comum



Fonte: elaborado pela autora (2024).

Outro aspecto importante incluído a partir dessa prototipação e que também se enquadra como uma diretriz de acessibilidade foi a responsividade: visualização em diferentes tamanhos de tela. Para isso, foi representado como seria a visualização em uma tela de celular (Figura 9).

Figura 9 — Tela inicial para usuário comum (visão *mobile*)

Fonte: elaborado pela autora (2024).

Na imagem o *header* apresentado em telas maiores é modificado e possui somente o nome do usuário e opção para fazer *logoff*. Todo o menu é movido para o *footer* da página, e seus elementos são representados por meio de ícones.

4.4.3 Prototipação de alto nível

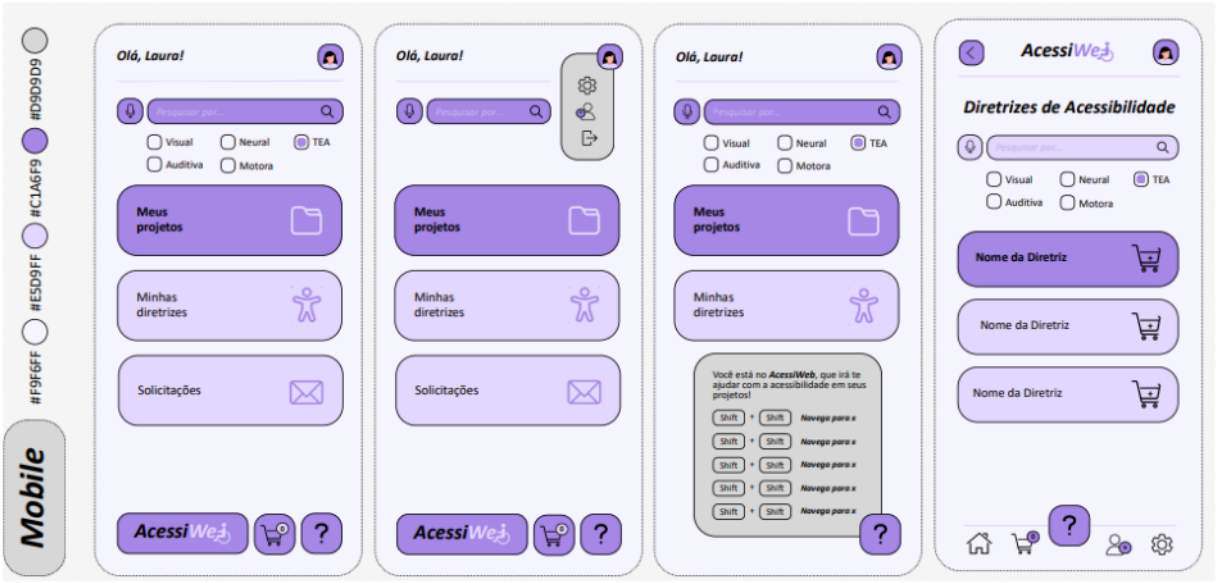
A prototipação de alto nível foi realizada com base na prototipação anterior, ao levar mais em consideração questões de identidade visual e uso de cores, criando um *design* mais agradável e amigável. Para essa prototipação foi feita a divisão entre *desktop* (telas maiores) e *mobile* (telas menores).

Nas Figuras 10 e 11 são apresentadas as telas iniciais para usuário comum. Nessa prototipação há a inserção da logo.

Figura 10 — Tela inicial para usuário comum (visão *desktop*)

Fonte: elaborado pela autora (2025).

Figura 11 — Telas iniciais para usuário comum (visão *mobile*)



Fonte: elaborado pela autora (2025).

O produto final foi pensado em trazer duas experiências conforme o tamanho da tela do usuário, em que, em telas menores, é muito similar à experiência de um aplicativo. Há também diferenciação para cada tipo de usuário, para cada um utilizar o sistema conforme as especificidades apresentadas nos requisitos.

Com as modelagens tanto do comportamento como da estrutura do sistema e

banco de dados, e as prototipações das interfaces, deu-se início ao desenvolvimento que será apresentado na seção a seguir.

4.5 Desenvolvimento do sistema

Previamente a escrita do código iniciou-se o desenvolvimento do *Technical Design Document* (TDD). Visto que, mesmo com a modelagem e prototipação, ainda foi necessário compreender outros aspectos do sistema. Nas seções seguintes serão abordadas questões técnicas e resultados da codificação[6] tanto do *front-end* como do *back-end*.

4.5.1 *Back-end*

O *back-end* foi construído com base em uma arquitetura comunicacional *RESTful* (o *front-end* e o *back-end* são separados, estado da sessão do cliente não é armazenado, os métodos HTTP são utilizados corretamente sendo identificados por URIs - endereço web único que identifica o recurso) e teve sua implementação e organização em camadas (*Controllers*, *Services* e *Repositories*). Foram utilizadas *Guards* para autenticação e controle de permissões, *filters* para tratamento de exceções, DTOs para transferência de dados e *TypeORM* para mapeamento de entidades no *PostgreSQL*, em que foram definidas *constraints* para cada dado, o que se pode traduzir como as regras que um dado irá seguir.

As principais *constraints* utilizadas foram:

- NULL: permissão para um dado não existir, campo vazio;
- NOT NULL: necessidade de informar o dado (obrigatoriedade);
- UNIQUE: a unicidade de um dado (dado único), não pode haver dois dados iguais;
- FK (*Foreign Key*): referência de um dado de uma tabela em outra, chamada de Chave Estrangeira;
- PK (*Primary Key*): referência principal do recurso na tabela, sendo chamada de Chave Primária, o que geralmente é o próprio identificador do recurso;

[6] O código pode ser acessado pelo github, por meio dos repositórios:
1- Repositório do *front-end*: https://github.com/lauravivan/acessiweb-front-end
2- Repositório do *back-end*: https://github.com/lauravivan/acessiweb-back-end

- CHECK: validação do dado conforme alguma especificidade. Por exemplo, a situação de solicitações só pode ser uma das seguintes: a) aprovada, b) rejeitada, c) pendente e d) aguardando.

Além disso, para auxiliar os diagramas, foram também definidas regras para questões como a situação da diretriz, preferências, exclusão de dados, relacionamentos, e outras. Elas foram definidas da seguinte forma:

- Regras do *status code* de uma solicitação de diretriz:
  - Os *status code* disponíveis são: “STANDBY”, “PENDING”, “REJECTED” e “APPROVED”;
  - Diretrizes cadastradas por usuário admin sempre estarão com situação “APPROVED”;
  - Se a situação estiver em “STANDBY” um usuário comum poderá realizar alterações na solicitação, ou excluí-la. Esse *status* indica que a solicitação ainda não foi enviada para um usuário *admin* analisá-la;
  - Se a situação estiver “PENDING” o usuário comum não poderá realizar alterações na solicitação nem a excluir, precisa aguardar até ter uma resposta “REJECTED” ou “APPROVED”;
  - Se a situação estiver “APPROVED” um usuário comum não poderá alterar sua solicitação nem a excluir. Essa solicitação se torna automaticamente uma diretriz “oficial”, portanto, não é mais uma solicitação. Assim, quando filtrar pelas solicitações será utilizado o identificador do usuário comum. O usuáro *admin* que for filtrar pelas solicitações irá utilizar o tipo (*isRequest*);
  - Se a situação estiver “REJECTED” um usuário comum pode realizar alterações e reenviar a solicitação, voltando para “PENDING”, ou pode a excluir.
- Regras — preferências:
  - Temas disponíveis: *light* e *dark;*
  - Fontes disponíveis: *arial*, *calibri*, *helvetica*, *tahoma*, *times new roman* e *verdana* (foram pensadas conforme acessibilidade);
  - Cores disponíveis do cursor: F7C8D4, A7C7E7, A8E6CF, C9A7E4, F5C6A5, F8E79D;
  - Faixa de brilho: 0 a 100;
  - Tamanhos de fonte disponíveis: *sm* (pequena), *md* (média), *lg*

(grande), *xlg* (extra-grande);
    - Tamanhos de cursor disponíveis: *sm* (pequena), *md* (média), *lg* (grande);
    - Faixa de espaçamento de linha: 1,5 a 3;
    - Faixa de espaçamento entre letras: 0,01 a 0,2.
- Regras — exclusão:
    - Uma solicitação ao ser excluída será permanentemente excluída do sistema;
    - Um projeto ao ser excluído será permanentemente excluído do sistema;
    - Uma diretriz ao ser excluída poderá ser recuperada, ao ser recuperada voltará para a situação “APPROVED”;
    - Ao deletar a conta do usuário, todos os seus dados relacionados (projetos, preferências, solicitações, autenticação) ficam permanentemente excluídos.
- Regras — relacionamentos:
    - Somente diretrizes APROVADAS podem ter relação com algum projeto;
    - Somente usuários comuns podem ter relação com projeto;
    - No momento em que uma solicitação de diretriz é aprovada, seu tipo muda (*isRequest*), mesmo assim, continuará relacionada ao usuário comum que a criou;
    - Usuários *admin* estarão sempre relacionados as diretrizes “oficiais”, que não tenham sido uma solicitação previamente.
- Regras — outras:
    - Se o usuário escolher colocar uma imagem é necessário informar a descrição dessa imagem (obrigatório);
    - Os roles disponíveis de usuário são “ADMIN” e “USER”;
    - Somente usuários *admin* poderão alterar a situação ou mensagem de uma solicitação.

Sobre o tratamento de erros, no quesito formato das respostas, foi criado um padrão a ser seguido, incluindo o código do erro, mensagem principal e um *array* com objetos detalhados para cada possível erro. O TDD inclui a tabela com todos os códigos de erro do sistema para tratamento de cada especificidade. Por exemplo, se

o usuário tentar enviar um dado com tipo incorreto, o código de erro a ser lançado deve ser *INVALID_TYPE,* pois se trata de tipo de dado incorreto, o número do código será o 400 e o contexto de uso validação de entradas. Além da tabela, foi incluído um exemplo da estrutura da resposta do erro. Esse tratamento no sistema foi centralizado em um *filter* facilitando a captura de erros e formatação da resposta para o *front-end*.

Ainda na questão de tratamento de erros, foram incluídos casos *Non-Error* para questões que são um possível erro do usuário, porém não devem interromper o fluxo (pensado na UX). Isso foi implementado, por exemplo, ao enviar um *array* com as diretrizes a serem incluídas no projeto. Se nele houver algum *id* inexistente ou duplicado, o sistema deve fazer um tratamento desses dados e somente lançar um erro se de fato não houver nenhuma diretriz válida para o cadastro.

A autenticação escolhida foi por meio de JWT *Token*, um padrão muito utilizado em sistemas *web*. Um JWT *Token* pode ser facilmente decifrado e não deve guardar dados sensíveis do usuário, somente o que for necessário para sua autenticação. Portanto, para cada rota que for necessária autenticação, o usuário deverá apresentar seu JWT *Token* gerado (quando é realizado *login* no sistema). Esse *token* terá duração de uma hora, após sua expiração o lado cliente da aplicação (*front-end*) deve se responsabilizar em gerar um novo *token* por meio do *Refresh Token,* um *token* com um tempo de expiração maior. Isso auxilia na questão de segurança. Como informado anteriormente, foram utilizadas *Guards* para definir políticas de acesso de rotas e permissões. Como o próprio nome indica, os *Guards* funcionam como guardas de uma rota. Portanto, toda rota que não for pública e necessitar do *token* de acesso, precisa utilizar um guarda para fazer essa validação. Portanto, ele protege a rota conforme a lógica criada, não permitindo que um usuário acesse se não apresentar um *token* válido. Com relação à política de acesso, essa mesma proteção é realizada em rotas que necessitam saber o tipo do acesso do usuário. Essa proteção foi inserida na API visto que há dois tipos de usuário no sistema, e, portanto, com permissões diferentes.

Todas as rotas que retornam alguma coleção de dados suportarão parâmetros como *limit* e *offset*. O *limit* refere-se ao número máximo de registros que serão retornados naquela requisição, isso auxilia no tempo de retorno e sobrecarga do servidor, pois ele não precisa retornar todos os dados registrados de uma vez só. Já o *offset* indica a posição inicial na qual serão recuperados os registros. É

importante que o sistema trate valores inválidos, portanto ambos possuem valores padrões. Além disso, foram abordadas também as informações a serem retornadas na requisição como: os registros encontrados, o total de registros encontrados, se existem mais dados a serem buscados, se existem dados anteriores a serem buscados, entre outros. Foram criados *decorators* (métodos especiais executados antes do chamador do *decorator*) customizados para lidar com paginação e filtros e obter seus dados recebidos.

No TDD, foi incluída uma seção para abordar somente as especificações das rotas do sistema, ao mostrar detalhadamente o URI, a descrição da rota, os dados a serem enviados por meio do *header,* as respostas, utilização de diagramas para representação de fluxos, entre outros pontos importantes.

4.5.2 *Front-end*

O *front-end* é uma aplicação *Next.js* que segue padrão de componentes reutilizáveis (princípios de UI/UX e *design system*) e separação de responsabilidades. A comunicação foi feita por meio de autenticação JWT e autorização baseada em *roles* (papéis). Utilizou-se *Context API* para gerenciamento de estado, estratégia de *fetching* de dados com *React Query*, *React Hook Form* em conjunto com *Zod* para validação e gerenciamento de formulários, *React Icons* para utilização de ícones padronizados, Sass para estilização, *React Hotkeys Hook* para realização de ações por meio de atalhos de teclado e *NextAuth* para autenticação e autorização com provedores externos.

As rotas do *front-end* foram construídas utilizando o roteamento com base em *file system*, do português, sistema de arquivos do *Next.js*, chamado de *App Router*, o que significa que é possível utilizar pastas e arquivos para definir rotas (Vercel, 2024). Nesse modelo, é utilizada a pasta *app* responsável por renderizar todas as rotas. Não somente páginas *web* são permitidas nessa pasta, e o *App Router* identifica se é uma página a ser renderizada pelo nome *page*, portanto, o arquivo precisa ser denominado dessa forma (no caso de *client components*).

O projeto foi dividido em duas partes: *auth* e *main*. Utilizou-se a convenção de *Group routes* na qual se utiliza uma pasta na pasta *app* sem quebrar o roteamento. Em *auth* foi criado componentes e *hooks* para auxiliar as páginas, tudo com relação às telas de autenticação do sistema, desde usuários visitantes até usuários *admin*.

Já na pasta *main* foram criadas todas as rotas principais do sistema, sendo elas para projetos, solicitações, diretrizes e configurações. E também criou-se uma pasta chamada de *admin* dentro de *main* para rotas que somente usuários *admin* poderão acessar.

Além das páginas, é interessante abordar sobre *layouts.* Um *layout* permite a construção de um molde para diversas páginas. Nesse sentido, utilizou-se essa funcionalidade para criar uma base tanto para as rotas de *main* como para as rotas de *auth.* Isso se torna particularmente útil na reutilização de interfaces, em que uma ou mais interfaces podem utilizar a mesma estrutura.

Ainda, na pasta *app* foi inserida uma pasta chamada de *api.* Diferentemente do arquivo de nome *page* para renderização de páginas no lado cliente, o arquivo para renderização de conteúdo na pasta *api* deve ser denominado de *route* (lado servidor). Foi utilizada para renderizar a rota do *Next Auth,* interceptando chamadas de autenticação não somente para provedores externos, mas também para autenticação com *email* e senha no próprio *back-end* da aplicação. É extremamente útil para reunir todos os chamados em um só local, gerenciando *login*, *tokens*, *sessions*, entre outras coisas.

Foram criadas também pastas para armazenar componentes e *custom hooks*. Componentes são úteis também para reutilização de interfaces. Como exemplo, foi criado um componente chamado *Pagination,* pois diversas interfaces precisam utilizar desse bloco de UI. Portanto, ao invés de utilizar o mesmo código em diferentes páginas, criou-se um componente para reutilizar nelas todas, criando assim uma padronização, menos código e mais legibilidade. Não se faz mais necessário lembrar como foi feito em outra página, basta utilizar o mesmo componente, e se houver alguma mudança nele, é refletido em todos os blocos que o utilizam. Na questão de *hooks, hooks* são funções que permitem reutilização de lógica, estado, blocos de UI, entre outros. *Hooks* são uma das bases do *React.js*, além dos componentes, e são constantemente utilizados para os mais diversos cenários. No caso de *custom hooks,* são *hooks* customizados pelo desenvolvedor para algo mais específico da aplicação, que pode e geralmente inclui outros *hooks.* No caso do componente *Pagination*, foi criado um *hook* para reunir funções e estados reutilizáveis.

Além disso, foi criada uma pasta *context* para utilizar a *Context* API do *React.js*. A *Context* API é extremamente útil para dados que devem ser

compartilhados entre componentes, porém, ao invés de passar como uma *prop,* que nesse caso é um parâmetro de um componente *React*, esse dado é compartilhado globalmente na aplicação entre os componentes que estiverem enlaçados (*wrapper*) pelo *provider* do *context*. Isso faz com que essa relação entre componentes seja muito mais simples e menos aninhada. Para essa aplicação foi criado um contexto para o carrinho, possibilitando ter acesso à quantidade de diretrizes adicionadas nele, com funções para interagir com ele, como adicionar uma diretriz ao carrinho, remover um projeto do carrinho, entre outras. Foi incluído também um contexto para notificações *push* globais, facilitando seu gerenciamento em qualquer local da aplicação. Contextos, de certa forma, precisam ser utilizados com cautela, portanto, se houver muitos em uma aplicação (uso excessivo) pode ser considerado um *gap* técnico (falha de arquitetura) e precisa ser revisto para encontrar outras alternativas.

Foi criada uma pasta denominada *utils* para reunir dados e funções que poderiam ser reutilizados por toda a aplicação. E uma pasta chamada *types* para assegurar os tipos (de dados) utilizados.

Para a comunicação com o *back-end*, criou-se uma pasta separada chamada *routes*, que reuniu todas as funções necessárias para *fetching* de dados. Qualquer página que precisar realizar uma ação e chamar o *back-end* irá utilizar a função adequada para aquela ação, portanto cada função utiliza uma rota diferente do back-end. Isso cria um tipo de API para o *front-end* utilizar.

Por último, foi criada uma pasta para guardar os *schemas* criados com a biblioteca *Zod*. Eles funcionam como uma espécie de esqueleto para os dados, cuidando da validação em conjunto com o *React Hook Form*. Também, um arquivo chamado *middleware* para controle das rotas, sendo um *server component*, do português componente servidor, camada que consegue ter acesso ao *token* do usuário e permitir ou não seu acesso a uma determinada rota. Ali foi incluído também o tratamento das rotas públicas do sistema.

Na seção seguinte será tratado dos recursos de acessibilidade.

4.5.3 Acessibilidade

Foram levantados 22 requisitos de acessibilidade a serem implementados no tempo de desenvolvimento disponível. Esses requisitos tiveram como base diretrizes de acessibilidade disponíveis na WCAG (Caldwell *et al.*, 2008) e no *AcessibiWeb*

(Souza; Dutra, 2024), sendo selecionadas após estudo nas prototipações do que seria necessário e o que seria interessante ter no sistema para torná-lo mais acessível.

Foi priorizado a utilização de tags HTML semânticas, que já possuem *roles* (papéis) bem definidos. Porém, em alguns cenários foi necessário a utilização dos papéis definidos pelo WAI-ARIA.

A aplicação possui diversos ícones, seja eles com o intuito de criar contexto para definir uma ação, seja somente para ter um indicativo visual extra para auxiliar no significado do elemento. Nesse sentido, para ícones de ação, que em sua grande maioria se encontram ou em botões, ou em *links*, optou-se pelo uso do atributo *aria-label* para trazer sentido a usuários de leitores de tela (servindo como o nome acessível do elemento) e *aria-hidden* para o leitor de tela desconsiderar o ícone. Com relação aos ícones somente para auxílio visual, optou-se por utilizar também o atributo *aria-hidden*. Como tratado previamente nesse trabalho, esses atributos se enquadram em *states* e *properties* de *roles* do WAI-ARIA. Eles aparecem em diversos elementos/componentes do sistema, e para testar suas funcionalidades foi utilizada a ferramenta NVDA, uma Tecnologia Assistiva *open source* amplamente usada por pessoas com deficiência.

Todas as imagens foram incluídas com o atributo *alt,* atributo específico de imagem para trazer uma descrição caso ela não seja renderizada e também para leitores de tela. Por esse motivo, foi pensado para que quando um usuário cadastrar uma diretriz com uma imagem, se torna obrigatório que ele também informe uma descrição para ela.

Além da aplicação ser responsiva, ela também não depende de orientação de tela para utilizar qualquer funcionalidade do sistema. Todos os elementos se adequam ao tamanho da tela, e o usuário não precisa rotacionar ela para continuar visualizando o conteúdo.

O idioma das páginas foi definido como `pt-BR` no atributo *lang* da *tag* HTML. E cada página teve seu título definido utilizando o *metadata* do *Next.js*. O *metadata* é um objeto que pode ser definido em um arquivo de *layout* ou de página, do lado do servidor. No caso dessa aplicação o título foi definido de forma estática, porém o *Next.js* possibilita também a geração de *metadata* dinâmica (Vercel, 2024).

Todos os *inputs* de texto da aplicação possuem alternativa de escrita por voz. Para essa implementação foi utilizada a *Web Speech* API que provê um serviço de

reconhecimento de voz que pode ser acessado pela interface *SpeechRecognition* (Mozilla Foundation, 2024). Essa API não necessita de instalação e é de fácil integração, porém nem todas as funcionalidades estão disponíveis em todos os navegadores, portanto a compatibilidade não é 100%.

Com relação à navegação, nos pontos **Acesso de conteúdo por mais de um meio (menu, *search*...)**, **Lógica sequencial e navegação intuitiva**, **Mostrar ao usuário visualmente o ponto focal de onde ele está** e ***Breadcrumb*.** O sistema possui diversos meios para acessar um conteúdo, seja por menu como por pesquisa como por *breadcrumb* e ele consegue saber exatamente onde está pelos pontos focais tanto na navegação como no *breadcrumb* (Figuras 12 e 13).

Figura 12 — Vários meios para navegação e busca de dados

Fonte: elaborado pela autora (2025).

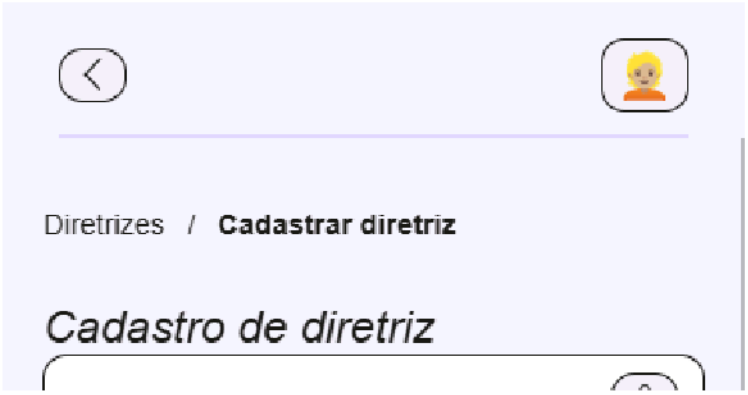
Figura 13 — Uso de *breadcrumb*

Fonte: elaborado pela autora (2025).

O usuário ainda tem acesso a um botão de pesquisa para buscar por

conteúdos. Ele também tem várias opções de filtros para buscar por diretrizes (exemplo acima). Ao clicar em criar diretriz, é redirecionado para a tela de cadastro de diretriz, que possui um *breadcrumb,* no qual o usuário consegue visualizar o local onde está e pode voltar para a tela em que estava. Ainda, para telas menores, o usuário possui acesso a um botão também para voltar para a página anterior (Figura 14).

Figura 14 — Navegação para página anterior



Fonte: elaborado pela autora (2025).

Buscou-se em todos os botões com escrita utilizar uma linguagem clara e direta, indicando exatamente a ação/funcionalidade/significado do botão. Essa mesma preocupação se deu também para *Headings* e *Labels.* Nas figuras anteriores tem-se a tela de cadastro de diretriz, que possui um *heading* (título para a página) com propósito claro, informando que o usuário irá realizar um cadastro de diretriz. Os inputs possuem *labels* ocultos para serem lidos por leitores de tela, referenciado com o nome de classe *sr-only*, uma técnica que utiliza o CSS para esconder o elemento da tela, mas ainda ser visível para Tecnologias Assistivas.

Ao verificar o contraste entre primeiro plano e segundo plano foi utilizada a ferramenta *WebAIM Contrast Checker*[7] (Figura 15).

[7] https://webaim.org/resources/contrastchecker/

Figura 15 — *WebAIM Contrast Checker*

Fonte: WebAIM Contrast Checker (2025).

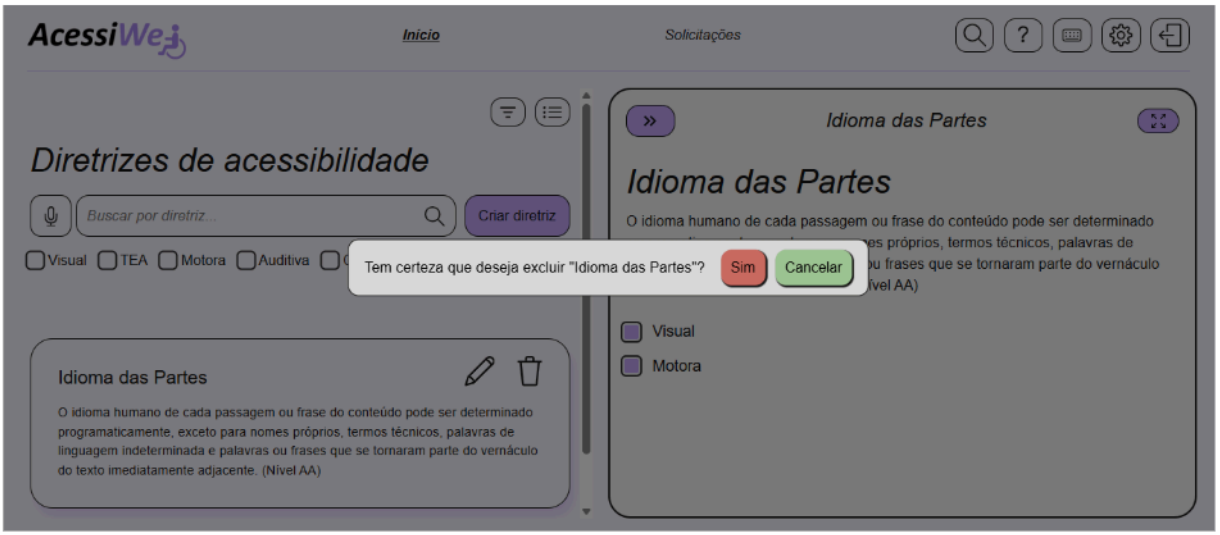
Como pode ser visto na imagem acima, para as cores escolhidas passarem no teste do contraste (usuários não terão problema para ler o texto ou algum tipo de dificuldade em visualizar a diferenciação entre primeiro e segundo plano), é necessário que ela ganhe *Pass* (aprovação) em todos os requisitos.

Sobre artifícios visuais, procurou-se fazer utilização da tríade cor, texto e elemento. Para mensagens de erro de entrada em formulário foi utilizado um texto explicativo e um ícone de X, ambos em vermelho. Para *modals* de confirmação com ação “crítica” como exclusão de dados foi utilizado o vermelho para o botão que confirma a ação e verde para o botão que cancela a ação. Como pode ser visto nas Figuras 16 e 17.

Figura 16 — Retorno de dados de entrada inválidos

Fonte: elaborado pela autora (2025).

Figura 17 — *Modal* de *delete*



Fonte: elaborado pela autora (2025).

Em todos os locais em que se utilizou algum tipo de lista, foi informado um identificador para cada item, além de o seu conteúdo ser único.

Para a acessibilidade por teclado, é importante que os comandos sejam informados de alguma forma para o usuário, que eles não conflitam com comandos

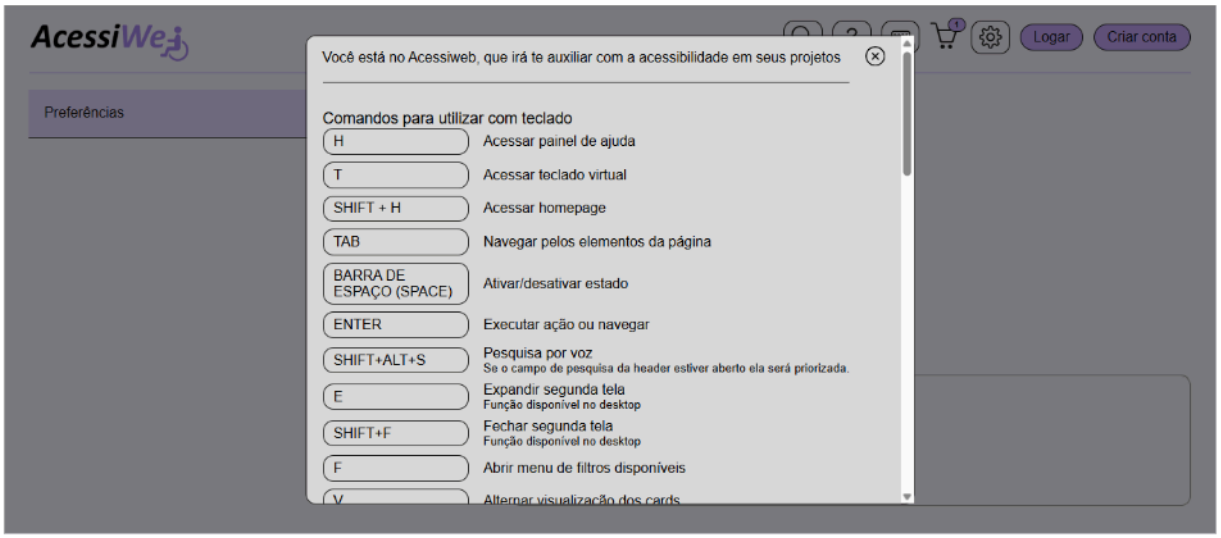
do sistema operacional (Windows, Linux, MacOS, etc.) ou do navegador (*Chrome*, *Firefox*, *Edge*, *Safari*, etc.), em que pode ser acessado na própria documentação na *internet*, e se possível utilizar a propriedade *aria-keyshortcuts*.

Também há a existência do atributo global *accesskey* no qual não é indicada a utilização pela MDN (Mozilla Foundation, 2024), em que se for utilizada precisa ser com cautela.

No sistema foi utilizado tanto o atributo *aria-keyshortcuts* como o *title* (servindo como uma *tooltip* para informar ao usuário que o componente pode ser acessado via teclado e qual comando utilizar).

Foi criado um *modal* que pode ser acionado por meio de um botão com um ponto de interrogação, indicando ajuda. Nele não somente o usuário tem acesso a quais comandos (nativos e customizados) ele pode utilizar por teclado, mas também por voz (Figura 18).

Figura 18 — Menu de ajuda



Fonte: elaborado pela autora (2025).

É importante frisar que mesmo que o usuário consiga navegar por todos os elementos com a tecla *tab* e utilizar outros comandos nativos para executar alguma ação ou navegar, foram criados comandos personalizados para mais rápido acesso. Por exemplo, se ele quiser fazer uma pesquisa rápida por comando de voz, ao invés de ter que navegar até o elemento, pode utilizar um comando rápido.

Além disso, foi incluído também um teclado virtual para utilizar com dispositivos *desktop*. Para todo e qualquer *input* o usuário consegue ter acesso

rápido ao teclado (comando T ou botão) e digitar com ele (Figura 19).

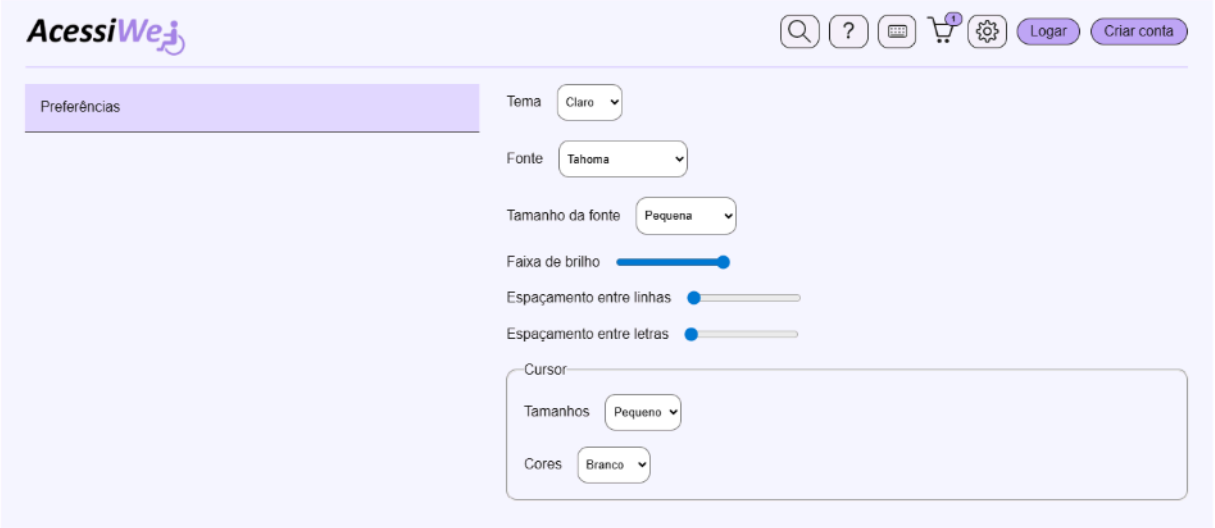
Figura 19 — Teclado Virtual

Fonte: elaborado pela autora (2025).

É relevante ressaltar o atributo *tabindex* que auxilia a tornar elementos que por padrão não conseguem ser focados pela tecla *tab.* No sistema foi utilizado em elementos como `li`. Em conjunto com *keydown*, uma ação que era possível somente com o *click,* é forçado que ao teclar *enter* a ação também seja executada (utilizando uma condição para verificar qual tecla foi acionada). Com o CSS foi possível estilizar para quando os elementos estiverem em foco mostrar um retorno muito claro para o usuário do elemento no qual ele está focando com o *tab.* Nos elementos com *tabindex*, foi utilizada a propriedade *focus-visible* para isso.

Ainda, o usuário possui alternativas de estilização e acessibilidade, como ajustar o estilo e tamanho da fonte, brilho, contraste, dentre outros aspectos (Figura 20).

Figura 20 — Preferências



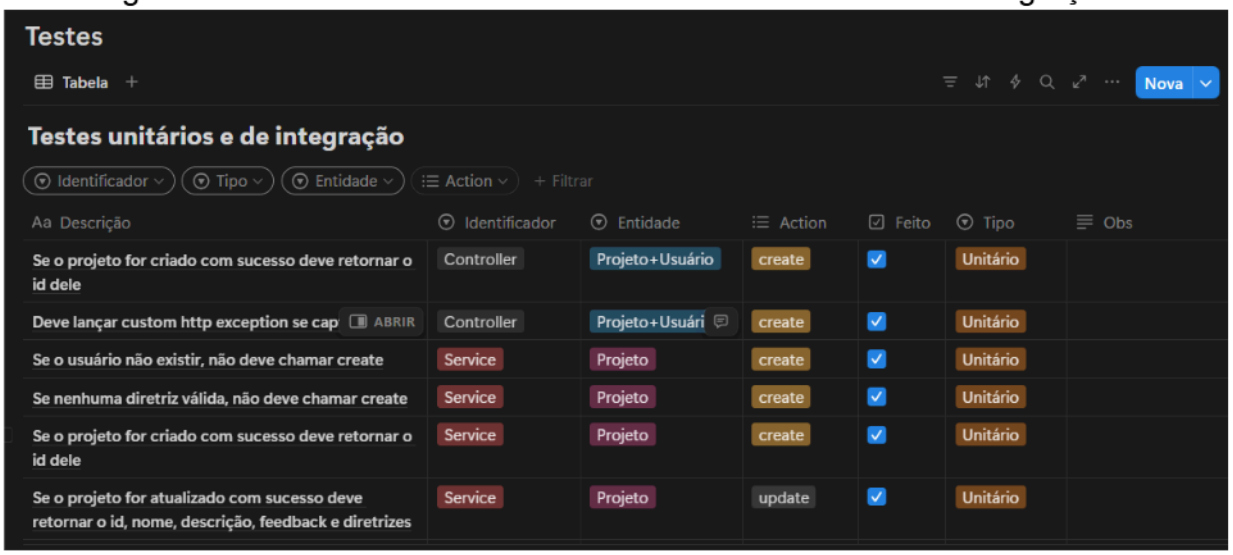
Fonte: elaborado pela autora (2025).

Em conjunto com o desenvolvimento foram realizados os testes unitários e de integração que serão abordados a seguir.

4.5.4 Testes

No TDD também foi incluída uma seção para testes, especificando quais testes unitários e de integração precisavam ser realizados na parte da API (Figura 21). Para testes do fluxo geral da API foi utilizado o *Postman* e sua *collection* também foi incluída no documento.

Figura 21 — Tabela de casos de uso de testes unitários e de integração



Testes

Tabela + Nova

Testes unitários e de integração

Identificador | Tipo | Entidade | Action | + Filtrar

Aa Descrição	Identificador	Entidade	Action	Feito	Tipo	Obs
Se o projeto for criado com sucesso deve retornar o id dele	Controller	Projeto+Usuário	create	☑	Unitário	
Deve lançar custom http exception se cap ABRIR	Controller	Projeto+Usuári	create	☑	Unitário	
Se o usuário não existir, não deve chamar create	Service	Projeto	create	☑	Unitário	
Se nenhuma diretriz válida, não deve chamar create	Service	Projeto	create	☑	Unitário	
Se o projeto for criado com sucesso deve retornar o id dele	Service	Projeto	create	☑	Unitário	
Se o projeto for atualizado com sucesso deve retornar o id, nome, descrição, feedback e diretrizes	Service	Projeto	update	☑	Unitário	

Fonte: elaborado pela autora (2025).

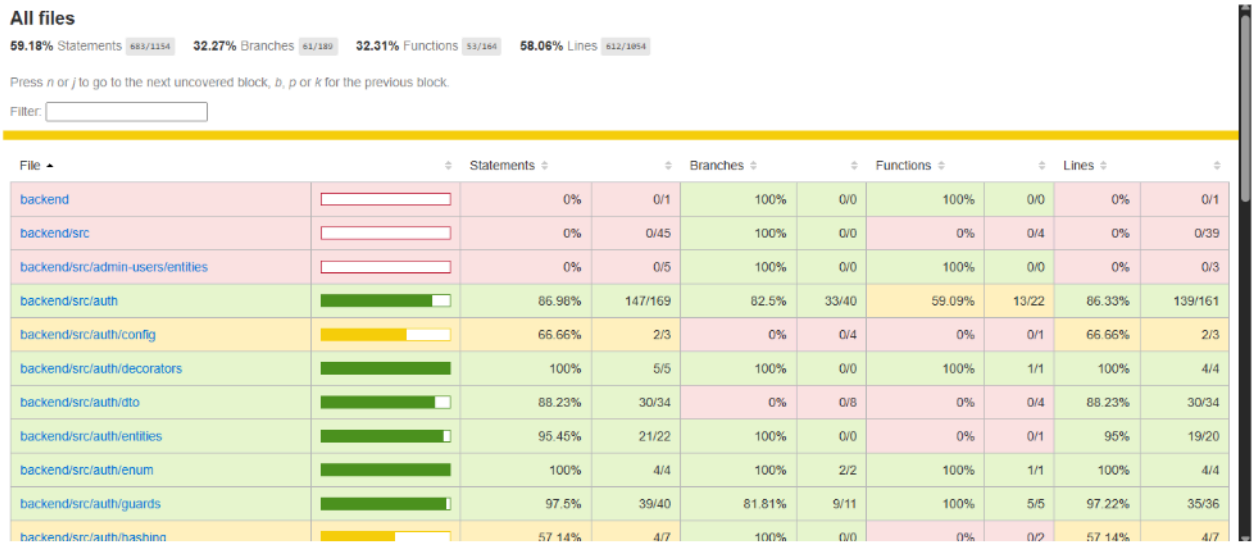
Para a API foram definidos diversos casos de uso, tanto de testes unitários como de testes de integração. Teve-se como base a premissa de que não é necessário cobrir 100% um sistema com testes, mas sim, realizar testes úteis que irão cobrir as partes do sistema que mais precisam de garantia de funcionamento (Valente, 2020).

Os testes adicionam uma camada de confiabilidade, servindo como uma ferramenta para reduzir risco de erros, não se trata de um método que irá excluir completamente as chances de algo acontecer. Nesse sentido, é importante não somente testar cenários positivos, mas pensar nas possíveis ações e consequências que poderiam ocorrer no sistema (Valente, 2020).

Procurou-se testar as principais entidades do sistema como Projeto, Diretriz, Usuário, etc., cobrindo as *Controllers* e Services. Porém, também realizaram-se testes unitários para outras partes do sistema como *Guards* e utilitários.

O relatório da cobertura de testes da API (Figura 22), que consegue gerar pela própria CLI do *Jest*, foi gerado com o comando `jest --coverage`. Para isso foi criada uma pasta denominada *coverage*, para ter acesso ao arquivo HTML do relatório.

Figura 22 — Cobertura de testes da API



All files

59.18% Statements 683/1154 **32.27%** Branches 61/189 **32.31%** Functions 53/164 **58.06%** Lines 612/1054

Press *n* or *j* to go to the next uncovered block, *b*, *p* or *k* for the previous block.

Filter:

File	Statements		Branches		Functions		Lines	
backend	0%	0/1	100%	0/0	100%	0/0	0%	0/1
backend/src	0%	0/45	100%	0/0	0%	0/4	0%	0/39
backend/src/admin-users/entities	0%	0/5	100%	0/0	100%	0/0	0%	0/3
backend/src/auth	86.98%	147/169	82.5%	33/40	59.09%	13/22	86.33%	139/161
backend/src/auth/config	66.66%	2/3	0%	0/4	0%	0/1	66.66%	2/3
backend/src/auth/decorators	100%	5/5	100%	0/0	100%	1/1	100%	4/4
backend/src/auth/dto	88.23%	30/34	0%	0/8	0%	0/4	88.23%	30/34
backend/src/auth/entities	95.45%	21/22	100%	0/0	0%	0/1	95%	19/20
backend/src/auth/enum	100%	4/4	100%	2/2	100%	1/1	100%	4/4
backend/src/auth/guards	97.5%	39/40	81.81%	9/11	100%	5/5	97.22%	35/36
backend/src/auth/hashing	57.14%	4/7	100%	0/0	0%	0/2	57.14%	4/7

Fonte: elaborado pela autora (2025).

O processo de testagem se mostrou um dos mais importantes. O teste permite capturar falhas de implementação de uma função, ao entender qual o objetivo dela e como ela deve ser implementada. Isso permitiu realizar alterações em funções para torná-las mais seguras, o que sem o teste inicial seria identificado em um teste manual ou até mesmo em produção.

Pelo relatório (Figura 22), consegue-se ter um vislumbre de uma cobertura razoável de testes, mas um número baixo de *branches* testadas (diferentes cenários). Com o seguimento do desenvolvimento da aplicação, os testes devem ser melhorados, cuidando para não testar somente cenários positivos. Também realizar uma cobertura maior.

Ainda, foi utilizada a ferramenta *Postman* para testes manuais, ao fazer uma requisição para a API e simular testes *End-2-End*.

4.6 Implantação

O *front-end* foi hospedado[8] na *Vercel* (Vercel, 2024), pela facilidade e rapidez de integração com *Next.js* e *GitHub*. Esta plataforma permite disponibilizar a aplicação gratuitamente com configurações mínimas, proporcionando um processo de *deploy* mais automatizado.

Para o *back-end*, foi utilizado o serviço *Render*[9] para hospedar a API. Embora

[8] https://acessiweb.vercel.app/
[9] https://render.com/

o *Render* também ofereça hospedagem para o banco de dados, seu plano gratuito possui um limite de tempo, funcionando como um período de teste até a exclusão automática do banco. Por essa razão, optou-se pelo *Supabase*[10], uma plataforma especializada em *PostgreSQL*, para gerenciar o banco de dados. Essa escolha manteve a arquitetura simplificada e reduziu a complexidade operacional do projeto.

Essas ferramentas atendem adequadamente às necessidades atuais do projeto. No entanto, com o crescimento da aplicação e o aumento das demandas de desempenho do servidor, será necessário avaliar alternativas mais robustas que possam garantir escalabilidade e tempos de resposta otimizados. Adicionalmente, com o crescimento do projeto, também será interessante implementar pipelines de CI/CD (*Continuous Integration and Continuous Deployment*), incluindo testes automatizados mais abrangentes e ambientes de *staging*, para garantir que bugs não sejam introduzidos em produção.

[10] https://supabase.com/

5 CONCLUSÃO

Este projeto se baseou em etapas sequenciais que nortearam o desenvolvimento de um sistema *web* acessível que teve como base uma página *web* estática intitulada *AcessibiWeb*. Essa página reuniu diretrizes de acessibilidade advindas de uma revisão da literatura por meio das diferentes deficiências: motora, auditiva, visual, cognitiva e neural, e TEA. O objetivo do sistema foi transformar essa página em um sistema *web* para centralizar e organizar essas diretrizes, facilitando o acesso a informações essenciais e servir de incentivo para criar experiências digitais mais inclusivas e alinhadas aos padrões de acessibilidade digital; dado que, com a crescente evolução da sociedade permeada pelas bases tecnológicas não pode ser excludente, e sim, permitir que a maior gama de usuários possam fazer proveito dessas ferramentas para auxiliar em seu cotidiano.

Para isso, foram levantados requisitos funcionais e não funcionais a fim de compreender o comportamento do sistema e a viabilidade de implementação das diretrizes de acessibilidade. Essa etapa foi fundamental para modelar o sistema e prototipar interfaces.

No entanto, a necessidade de um maior detalhamento e aprofundamento em certos aspectos de implementação conduziu à elaboração de um Documento de Especificação Técnica do Sistema. Ele conferiu significativa fluidez ao processo de desenvolvimento, com um planejamento mais preciso, maior controle e o refinamento contínuo das ideias.

Durante a fase de desenvolvimento, o principal desafio residiu em converter as diretrizes de acessibilidade em implementações eficazes. Embora o projeto não almeja a acessibilidade em sua totalidade, dada a limitação temporal e a amplitude do tema, o foco foi garantir a aplicação efetiva das diretrizes selecionadas, visando tornar o sistema genuinamente mais acessível.

Adicionalmente, a concepção e implementação de casos de teste representaram um obstáculo significativo. A configuração inicial dos testes de integração, em particular, demandou considerável esforço e aprendizado.

O projeto foi construído a partir de uma abordagem interdisciplinar, integrando conhecimentos de Engenharia de Software, Interação Humano-Computador, Programação *Web* e Banco de Dados. Essa integração permitiu tratar questões teóricas e práticas relevantes ao desenvolvimento de sistemas inclusivos e voltados

ao bem-estar social, em que o foco principal foi na inclusão de PcD.

O projeto culminou em um sistema *web* substancialmente mais acessível. O usuário poderá acessar suas funcionalidades por meio de comandos de teclado, bem como inserir informações via comandos por voz, ou teclado virtual. Adicionalmente, a plataforma disponibiliza alternativas de cores, tipos e tamanhos de fontes, espaçamento entre linhas, dentre outros recursos adaptáveis. Nesse sentido, a aplicação transcende a mera manipulação de dados, estabelecendo-se como uma ferramenta inclusiva e acessível a um público diversificado.

Sendo assim, este projeto permitiu à pessoa desenvolvedora aplicar conhecimentos em um contexto prático, contribuindo para seu crescimento profissional e para o desenvolvimento de soluções reais voltadas à sociedade. A proposta resultou em uma aplicação, intitulada *AcessiWeb,* que organiza e exemplifica diretrizes de acessibilidade em um sistema acessível, com potencial de aplicação em diferentes cenários de desenvolvimento de *software*.

Como continuidade, está prevista a aplicação de um questionário baseado no Modelo de Aceitação de Tecnologia (*Technology Acceptance Model*), a fim de avaliar quanto à acessibilidade e aos fluxos de interação com o sistema. Com isso, espera-se validar as escolhas feitas e aprimorar ainda mais a solução desenvolvida para garantir sua efetividade e usabilidade por diferentes públicos.

Para pesquisas futuras, sugere-se a análise da utilização da plataforma e os dados obtidos por meio dela, para observar e fazer um levantamento das deficiências e diretrizes mais abordadas nos projetos.

REFERÊNCIAS

AGÊNCIA SENADO. Senado aprova novo Símbolo Internacional de Acessibilidade. **Senadonoticias,** Brasília, 29 abr. 2025. Disponível em: https://www12.senado.leg.br/noticias/materias/2025/04/29/senado-aprova-novo-simbolo-internacional-de-acessibilidade. Acesso em: 12 jun. 2025.

ALMEIDA, Cinthia Carvalho; CARDOSO, Ariston de Lima. **Acessibilidade em materiais didáticos digitais para estudantes com deficiência visual.** 2023. 30 p. Dissertação (Mestrado) – Programa de Pós-Graduação em Educação Científica, Inclusão e Diversidade, Universidade Federal do Recôncavo da Bahia. Feira de Santana, 2023. Disponível em: http://ri.ufrb.edu.br/jspui/handle/123456789/3992. Acesso em: 5 nov. 2024.

AMADEU, Claudia Vicci; SILVA, Jorge Luiz da; MANOCHIO-PINA, Marina Garcia. Inclusão digital e suas relações com o empoderamento, a qualidade de vida e o bem-estar. **Aletheia**, Canoas, RS, v. 55, n. 1, p. 207–223, 2022. DOI 10.29327/226091.55.1-11. Disponível em: http://pepsic.bvsalud.org/scielo.php?script=sci_arttext&pid=S1413-03942022000100012&lng=pt&nrm=iso. Acesso em: 5 nov. 2024.

BARBOSA, Simone Diniz Junqueira *et al*. **Interação humano-computador e experiência do usuário.** Rio de Janeiro: Ed. do autor, 2021.

BASTOS, Luana Arrial *et al*. As pessoas com deficiência e a acessibilidade digital nos dias de hoje no brasil. **Revista Tecnologia e Sociedade**, v. 19, n. 58, p. 212–228, 2023. DOI 10.3895/rts.v19n58.16170. Disponível em: https://periodicos.utfpr.edu.br/rts/article/view/16170. Acesso em: 5 nov. 2024.

BAUMGARTNER, Stefan. **TypeScript in 50 lessons.** Freiburg: Smashing Media AG, 2020.

BOURQUE, Pierr; FAIRLEY, Richard E. (ed.). **SWEBOK V3.0:** guide to the software engineering body of knowledge. Piscataway, NJ: IEEE, 2014. Disponível em: https://ieeecs-media.computer.org/media/education/swebok/swebok-v3.pdf. Acesso em: 29 abr. 2025.

BRASIL. Ministério do Planejamento, Orçamento e Gestão. Secretaria de Logística e Tecnologia da Informação. **eMAG modelo de acessibilidade em governo eletrônico. Versão 3.1.** Brasília: Ministério do Planejamento, Orçamento e Gestão, 2014. Disponível em: https://www.gov.br/governodigital/pt-br/acessibilidade-e-usuario/acessibilidade-digital/eMAGv31.pdf. Acesso em: 5 nov. 2024.

BRASIL. **Lei 13.146 de 6 de Julho de 2015.** Institui a lei brasileira de inclusão da pessoa com deficiência (estatuto da pessoa com deficiência). 2015. Disponível em: https://www.planalto.gov.br/ccivil_03/_ato2015-2018/2015/lei/l13146.htm. Acesso em: 5 nov. 2024.

CALDWELL, Ben; COOPER, Michael; REID, Loretta Guarino; VANDERHEIDEN, Gree. **Web content accessibility guidelines (WCAG) 2.0.** Cambridge: W3C, 2008. Disponível em: https://www.w3.org/TR/2008/REC-WCAG20-20081211/. Acesso em: 5 nov. 2024.

CARMO, Paloma; DUARTE, Felipe; GOMES, Ana Bárbara. **Inclusão digital como política pública: Brasil e América do Sul em perspectiva.** Belo Horizonte: Instituto de Referência em Internet e Sociedade, 2020.

CARVALHO, Thiago Leite e. **Orientação a objetos:** aprenda seus conceitos e suas aplicabilidades de forma efetiva. São Paulo: Casa do Código, 2016.

COELHO, Rosália; SOUSA, Jenny. Inclusão social: a communitas no centro do desenvolvimento comunitário. **Revista Interdisciplinar em Cultura e Sociedade (RICS)**, São Luís, MA, v. 10, n. 1, p. 36–53, 2024. DOI 10.18764/2447-6498.v10n1.2024.3. Disponível em: https://periodicoseletronicos.ufma.br/index.php/ricultsociedade/article/view/23867. Acesso em: 5 nov. 2024.

COSTA, Maria Izabel Sanches; IANNI, Aurea Maria Zöllner. A dialética do conceito de exclusão/inclusão social. *In*: COSTA, M.I.S., and IANNI, A.M.Z. **Individualização, cidadania e inclusão na sociedade contemporânea:** uma análise teórica. 1. ed. São Bernardo do Campo: UFABC, 2018, p. 75–101. Disponível em: https://books.scielo.org/id/sysng/pdf/costa-9788568576953.pdf. Acesso em: 5 nov. 2024.

DIGGS, Joanmarie; NURTHEN, James; COOPER, Michael; MACLEOD, Carolyn. **Accessible rich internet applications (WAI-ARIA) 1.2**. Cambridge: W3C, 2023. Disponível em: https://www.w3.org/TR/wai-aria/. Acesso em: 5 nov. 2024.

FERREIRA SOBRINHO JUNIOR, Durval; SONZA, Andréa Poletto; FERNANDES, Woquiton Lima. Concepção e desenvolvimento de um sistema web para avaliação de acessibilidade de sites e geração automatizada de selo. **Revista Diálogos e Perspectivas em Educação Especial**, Marília, SP, v. 11, n. 3, p. e0240035, 2024. DOI 10.36311/2358-8845.2024.v11n3.e0240035. Disponível em: https://revistas.marilia.unesp.br/index.php/dialogoseperspectivas/article/view/16402. Acesso em: 5 nov. 2024.

FORBES. Menos de 1% dos sites brasileiros são considerados acessíveis, diz pesquisa. **Forbes Brasil**, São Paulo, 28 jul. 2021. Disponível em: https://forbes.com.br/forbesesg/2021/07/menos-de-1-dos-sites-brasileiros-sao-considerados-acessiveis-diz-pesquisa/. Acesso em: 5 nov. 2024.

FRAZ, Joeanne Neves *et al.* Tecnologia assistiva: produtos e serviços disponíveis na internet. **Pesquisa Brasileira em Ciência da Informação e Biblioteconomia**, João Pessoa, PB, v. 16, n. 2, p. 70–84, 2021. Disponível em: https://periodicos.ufba.br/index.php/revistaici/article/view/35225/20736. Acesso em: 5 nov. 2024.

GIL, Antônio Carlos. **Como elaborar projetos de pesquisa.** 4. ed. São Paulo: Atlas,

2002.

GUERRA, Camila Guedes; CARNEIRO, Fernando Henrique Fogaça; SANTAROSA, Lucila Maria Costi. Perspectivas de usuários surdos sobre um ambiente digital acessível: análises sobre os princípios de acessibilidade à web. **EaD em Foco**, Rio de Janeiro, RJ, v. 12, n. 2, p. e1854, 2022. DOI 10.18264/eadf.v12i2.1854. Disponível em: https://eademfoco.cecierj.edu.br/index.php/Revista/article/view/1854. Acesso em: 5 nov. 2024.

IBGE. **Pessoas com deficiência:** 2022 / IBGE, coordenação de pesquisas por amostra de domicílios. Rio de Janeiro: IBGE, 2023. Disponível em: https://biblioteca.ibge.gov.br/index.php/biblioteca-catalogo?view=detalhes&id=2102013. Acesso em: 5 nov. 2024.

MARIA, Marina. **Orientações gerais sobre acessibilidade e inclusão para profissionais de comunicação.** Rio de Janeiro: Fiocruz/ICICT, 2020. 41 p. Disponível em: https://www.arca.fiocruz.br/handle/icict/43374. Acesso em: 5 nov. 2024.

META PLATFORMS. **React:** the library for web and native user interfaces. Disponível em: https://react.dev/. Acesso em: 5 nov. 2024.

MOZILLA FOUNDATION. **MDN Web Docs.** Disponível em: https://developer.mozilla.org/en-US/. Acesso em: 5 nov. 2024.

MYSLIWIEC, Kamil. **NestJS:** a progressive node.js framework. Disponível em: https://nestjs.com/. Aceso em: 5 nov. 2024.

PARREIRA JÚNIOR, Walteno Martins. **Apostila engenharia de software.** [2010]. Disponível em: http://www.waltenomartins.com.br/ap_es_v1.pdf. Acesso em: 5 nov. 2024.

PRESSMAN, Roger S.; MAXIM, Bruce R. **Engenharia de software:** uma abordagem profissional. 9. ed. Porto Alegre: AMGH, 2021.

RAGNEDDA, Massimo; RUIU, Maria Laura; ADDEO, Felice. The self-reinforcing effect of digital and social exclusion: the inequality loop. **Telematics and Informatics**, Oxford, v. 72, p. 101852, 2022. DOI 10.1016/j.tele.2022.101852. Disponível em: https://www.sciencedirect.com/science/article/pii/S0736585322000855. Acesso em: 5 nov. 2024.

RANGEL, Leonardo Augusto de Oliveira. A inclusão das pessoas com deficiência visual na sociedade: a carência de acessibilidade digital e os reflexos na educação e no trabalho, segundo o estatuto da pessoa com deficiência. **Brazilian Journal of Development**, São José dos Pinhais, v. 9, n. 1, p. 1135–1154, 2023. DOI 10.34117/bjdv9n1-080. Disponível em: https://ojs.brazilianjournals.com.br/ojs/index.php/BRJD/article/view/56018. Acesso em: 5 nov. 2024.

ROZA, Rodrigo Hipólito. O papel das tecnologias da informação e comunicação na atual sociedade. **Ciência da Informação**, Brasília, v. 49, n. 1, p. 67–75, 2020. DOI 10.18225/ci.inf.v49i1.4755. Disponível em: https://revista.ibict.br/ciinf/article/view/4755. Acesso em: 5 nov. 2024.

SANFELICE, Gustavo Roese; RENNER, Jacinta Sidegum. Qualidade de vida — aproximações conceituais. *In*: SANFELICE, G.R.; BASSANI, P.S. (org.). **Diversidade cultural e inclusão social.** 1. ed. Novo Hamburgo: Universidade Feevale, 2020, p. 7-17. Disponível em: https://www.feevale.br/institucional/editora-feevale/diversidade-cultural-e-inclusao-social. Acesso em: 5 nov. 2024.

SILVA, Edna Lucia da; MENEZES, Estera Muszkat. **Metodologia da pesquisa e elaboração de dissertação**. 3. ed. Florianópolis: UFSC/PPGEP/LED, 2001.

SOLER, João Vinícius Alves; FARINA, Renata Mirella; FLORIAN, Fabiana. A relevância da acessibilidade intuitiva: como o foco na usabilidade no design de interfaces impacta o usuário. **Revista Interface Tecnológica**, v. 18, n. 2, p. 194–207, 2021.

SOMMERVILLE, Ian. **Engenharia de software**. 9. ed. São Paulo: Pearson Prentice Hall, 2011.

SOUZA, Camilla; DUTRA, Taynara. AcessibiWeb — uma revisão sistemática da literatura para identificar, categorizar e divulgar diretrizes de acessibilidade web. *In*: ENCONTRO NACIONAL DE COMPUTAÇÃO DOS INSTITUTOS FEDERAIS (ENCOMPIF), 11., 2024, Brasília. **Anais [...]**. Porto Alegre: Sociedade Brasileira de Computação, 2024. p. 34–41.

THE POSTGRESQL GLOBAL DEVELOPMENT GROUP. **PostgreSQL:** the world's most advanced open source database. Disponível em: https://www.postgresql.org/. Acesso em: 5 nov. 2024.

TYPEORM. **TypeORM:** amazing ORM for TypeScript and JavaScript. Disponível em: https://typeorm.io/. Acesso em: 5 nov. 2024.

VALENTE, Marco Tulio. **Engenharia de software moderna**: princípios e práticas para desenvolvimento de software com produtividade. Belo Horizonte: ASERG/DCC/UFMG, 2020.

VERCEL. **The react framework for the web.** Disponível em: https://nextjs.org/. Acesso em: 5 nov. 2024.

THE OPENJS FOUNDATION. **Jest.** Disponível em: https://jestjs.io/. Acesso em: 5 nov. 2024.

WACKER, Mike. **Just Say No to More End-to-End Tests.** [2015]. Disponível em: https://testing.googleblog.com/2015/04/just-say-no-to-more-end-to-end-tests.html. Acesso em: 12 maio 2025.

WAZLAWICK, Raul Sidnei. **Metodologia de pesquisa para ciência da computação**. 6. ed. Rio de Janeiro: Elsevier, 2009.

WORLD WIDE WEB CONSORTIUM. **World wide web consortium.** Disponível em: https://www.w3.org/. Acesso em: 5 nov. 2024.