UNIVERSIDADE FEDERAL DO CEARÁ

CAMPUS QUIXADÁ

CURSO DE GRADUAÇÃO EM ENGENHARIA DE COMPUTAÇÃO

PAULO RICARDO DA SILVA LOPES

VERIFICAÇÃO DO NÍVEL DE ACESSIBILIDADE DE FERRAMENTAS PARA DESENVOLVIMENTO DE SOFTWARE POR UM DESENVOLVEDOR BAIXA VISÃO COM A FERRAMENTA DE AMPLIAÇÃO DE TELA

QUIXADÁ

2023

PAULO RICARDO DA SILVA LOPES

VERIFICAÇÃO DO NÍVEL DE ACESSIBILIDADE DE FERRAMENTAS PARA DESENVOLVIMENTO DE SOFTWARE POR UM DESENVOLVEDOR BAIXA VISÃO COM A FERRAMENTA DE AMPLIAÇÃO DE TELA

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

Orientador: Prof. Dr. Antônio Joel Ramiro de Castro

Coorientador: Prof. Marcelo Martins da Silva

QUIXADÁ

2023

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

L855v Lopes, Paulo Ricardo da Silva.
Verificação do nível de acessibilidade de ferramentas para desenvolvimento de software por um desenvolvedor Baixa Visão com a ferramenta de ampliação de tela / Paulo Ricardo da Silva Lopes. – 2023.
57 f. : il. color.

Trabalho de Conclusão de Curso (graduação) – Universidade Federal do Ceará, Campus de Quixadá, Curso de Engenharia de Computação, Quixadá, 2023.
Orientação: Prof. Dr. Antônio Joel Ramiro de Castro.
Coorientação: Prof. Marcelo Martins da Silva.

1. Acessibilidade. 2. Software-Desenvolvimento. 3. Transtornos da Visão. 4. Baixa visão. I. Título.
CDD 621.39

PAULO RICARDO DA SILVA LOPES

VERIFICAÇÃO DO NÍVEL DE ACESSIBILIDADE DE FERRAMENTAS PARA DESENVOLVIMENTO DE SOFTWARE POR UM DESENVOLVEDOR BAIXA VISÃO COM A FERRAMENTA DE AMPLIAÇÃO DE TELA

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

Aprovada em: ____/____/_____.

BANCA EXAMINADORA

Prof. Dr. Antônio Joel Ramiro de Castro (Orientador)
Universidade Federal do Ceará (UFC)

Prof. Marcelo Martins da Silva (Coorientador)
Universidade Federal do Ceará (UFC)

Prof. Me. Thiago Werlley Bandeira da Silva
Universidade Federal do Ceará (UFC)

Prof. Dr. José Gadelha da Silva Filho
Faculdade de Educação, Ciências e Letras do Sertão
Central (FECLESC/UECE)

Dedico este trabalho a minha avó, Maria Margarida Baraúna da Silva (*in memorian*), e minha mãe, Maria Denize Braúna da Silva, e minha tia, Dina de Sousa Lopes Virginio, e meu tio, Jason Virginio, que me amaram e foram meu porto seguro me acolhendo com seu amor e carinho.

AGRADECIMENTOS

Com amor, agradeço primeiramente a Deus por me dar forças todos os dias para nunca desistir. Com enorme prazer e carinho, agradeço à minha avó, Maria Margarida Baraúna da Silva (*in memorian*), e minha mãe, Maria Denize Braúna da Silva, que foram exemplos de mulheres guerreiras e de pessoas boas, me ensinando sempre que a honestidade e o trabalho duro serão sempre as melhores armas para ter uma boa vida.

Com amor, agradeço à minha tia, Dina de Sousa Lopes Virginio, e meu tio, Jason Virginio, por me acolherem quando mais precisei com seu amor e carinho e me ajudar em momentos difíceis.

Agradeço com amor à minha avó, Leontina de Sousa Lopes, que fez parte do meu crescimento e do crescimento dos meus tios, cuidando e acolhendo.

Agradeço também a todo o corpo docente da Universidade Federal do Ceará - campus Quixadá, que dedicaram esforços no ensino de nós, estudantes. Agradeço à professora Dra. Andréia Libório que não mediu esforços para equipar o campus com um equipamento leitor de textos impressos para eu conseguir ouvir os livros da biblioteca.

Agradeço ao professor Dr. Paulo de Tarso que sempre se mostrou preocupado com o ensino dos seus alunos e dedicou algumas horas de sua semana para tirar as minhas dúvidas.

Agradeço à todos os professores que fizeram parte dessa jornada e agradeço aos colaboradores do campus Quixadá que sempre foram atenciosos com todos os estudantes.

Agradeço a banca por dedicarem o seu tempo e sua atenção ao meu trabalho, agradeço principalmente ao professor Dr. Joel Castro e ao professor Marcelo Martins que acreditaram em mim e me auxiliaram a completar esse ciclo super importante na minha vida.

Agradeço à meu amigo, Ederson Abreu, que foi o primeiro a me aconselhar e apoiar a ir para Quixadá, me acolhendo com seus colegas universitários em sua casa.

Com carinho, agradeço aos meus companheiros de universidade Henrique Rocha, Ruan Derlan, Mateus Pedrosa, Lucas Nobre, Camila Diógenes, Beatriz Precebes e Mateus Torres que fizeram parte de muitas vitórias e conquistas em todos esses anos.

Agradeço aos meus companheiros do Programa de Educação Tutorial - Tecnologia da Informação Robertty Freitas, Mateus Lima, Paulo Miranda e Letícia Saraiva por serem uma inspiração como estudantes e por serem humildes sempre. Agradeço à tutora do PET-TI Carla Bezerra pela oportunidade e apoio que me deu na bolsa.

Agradeço aos bolsistas de iniciação acadêmica Johnny Marcos, João Paulo, Robert

Cabral, Mateus Sousa e Davi Gomes que foram meus monitores em disciplinas e por toda a sua paciência ao me ensinar.

Com enorme carinho, agradeço aos meus amigos Will Borges, Liviane Libório, Débora Libório e Liliane Angelim por me darem força em momentos difíceis e me darem conselhos desde antes da universidade.

Agradeço aos meus amigos Afrânio Martins e Silvia Martins por me apoiarem em momentos difíceis e por também fazer parte de momentos felizes.

Agradeço do fundo do meu coração à todos vocês, meus amigos e familiares, vocês fazem parte da minha vida e fizeram parte dessa jornada.

E por fim, agradeço a mim memso por nunca ter pensado em desistir nem por um segundo da universidade, mesmo com todas as dificuldades e limitações que tive no decorrer dos semestres.

"Para as pessoas sem deficiência, a tecnologia torna as coisas mais fáceis. Para as pessoas com deficiência, a tecnologia torna as coisas possíveis."

(Mary Pat Radabaugh)

RESUMO

De acordo com estatísticas de um relatório publicado em 2012 pela Organização Mundial da Saúde (OMS), mais de 1 bilhão de pessoas no mundo todo possuem algum tipo de deficiência, e cerca de 253 milhões possuem algum grau de deficiência visual. Mais recentemente, em 2019, a OMS divulgou dados de um novo relatório indicando que o número de pessoas com deficiência visual aumentou para 2,2 bilhões de pessoas. No Brasil, de acordo com o Censo de 2010 do Instituto Brasileiro de Geografia e Estatística (IBGE), cerca de 46 milhões de pessoas se declararam com algum grau de deficiência, onde cerca de 8,64 milhões são pessoas com deficiência visual. Diante dessa realidade, a inclusão social e o contínuo desenvolvimento de ferramentas de acessibilidade desempenham um papel fundamental no auxílio dessas pessoas que enfrentam vários desafios, principalmente a dificuldade de acesso à informação no mundo digital e a entrada no mercado de trabalho do mundo da tecnologia da informação e comunicação. Portanto, este trabalho está direcionado para as ferramentas de ampliação nativas dos sistemas operacionais *Windows* e Linux, com o objetivo de verificar o nível de acessibilidade de ferramentas utilizadas no desenvolvimento de *software* ao utilizar as ferramentas de ampliação. As ferramentas de desenvolvimento de *software* neste trabalho forma o *Visual Studio Code* (VSCode) e o *Postman*. Foi desenvolvido uma API simples utilizando o editor de código VSCode e para testar a API foi utilizado o *Postman*. No desenvolvimento da API foi feita a verificação do nível de acessibilidade observando como a ferramenta de ampliação se comportava ao navegar no VSCode e ao utilizar os seus recursos, como um desenvolvedor faria no mundo real, instalando *plugins*, criando pastas no projeto, navegando no projeto com a tecla “tab”, pesquisando arquivos, utilizando terminal integrado e digitando o código da API. No *Postman*, a verificação também foi repetir um cenário do mundo real, um em que o desenvolvedor precisa criar *collections*, pastas, variáveis de ambiente, requisições e navegar com a tecla “tab”. Por fim, foram identificados pontos fortes e pontos fracos nas duas ferramentas de ampliação dos dois sistemas operacionais.

Palavras-chave: acessibilidade; ferramenta de ampliação; desenvolvimento de software; desenvolvedores com deficiência visual; baixa visão.

ABSTRACT

According to statistics from a report published in 2012 by the World Health Organization (WHO), over 1 billion people worldwide have some form of disability, with approximately 253 million having some degree of visual impairment. More recently, in 2019, the WHO released data from a new report indicating that the number of people with visual impairments has increased to 2.2 billion. In Brazil, according to the 2010 Census conducted by the Brazilian Institute of Geography and Statistics (IBGE), around 46 million people declared having some degree of disability, with approximately 8.64 million being visually impaired individuals. In light of this reality, social inclusion and the continuous development of accessibility tools play a crucial role in assisting individuals facing various challenges, particularly in terms of accessing information in the digital world and entering the information technology and communication job market. Therefore, this work focuses on the native magnification tools of the Windows and Linux operating systems, with the aim of assessing the accessibility level of tools used in software development when utilizing magnification features. The software development tools used in this work are Visual Studio Code (VSCode) and Postman. A simple API was developed using the VSCode code editor, and the Postman tool was utilized to test the API. In the API development process, the accessibility level was examined by observing how the magnification tool behaved while navigating within VSCode and utilizing its features, simulating the actions of a real-world developer such as installing plugins, creating project folders, navigating the project using the "tab" key, searching for files, using the integrated terminal, and typing API code. In Postman, the assessment also involved replicating a real-world scenario where a developer needs to create collections, folders, environment variables, requests, and navigate using the "tab" key. Finally, strengths and weaknesses were identified in the magnification tools of both operating systems.

Keywords: accessibility; magnification tool; software development; visually impaired developers; low vision.

LISTA DE QUADROS

LISTA DE ABREVIATURAS E SIGLAS

OMS	Organização Mundial de Saúde
DV	Deficiência Visual
IBGE	Instituto Brasileiro de Geografia e Estatística
BV	Baixa Visão
VS	Visão Subnormal
TA	Tecnologia Assistiva
EUA	Estados Unidos da América
UFRJ	Universidade Federal do Rio de Janeiro
NVDA	*NonVisual Desktop Access*
JAWS	*Job Access with Speech*
PDF	*Portable Document Format*
GUI	Interface Gráfica do Usuário
DDVs	Desenvolvedores com Deficiência Visual
VSCode	*Visual Studio Code*
SO	Sistema Operacional
API	*Application Programming Interface*
IA	Inteligencia Artificial
IdC	Internet das Coisas
TIC	Tecnologia da Informação e Comunicação
OCR	Reconhecimento Óptico de Caracteres
WCAG	*Web Content Accessibility Guidelines*
W3C	*World Wide Web*
HTTP	*Hyper Text Transfer Protocol*
URL	*Uniform Resource Locator*
JSON	*JavaScript Object Notation*
XML	*eXtensible Markup Language*
STS	*Spring Tools Suite*
OA	Objetos de Aprendizagem
IDE	Ambiente de Desenvolvimento Integrado
RS	Revisão Sistemática

IEEE	Instituto de Engenheiros Eletricistas e Eletrônicos
DSR	*Design Science Research*
QFD	Desdobramento da Função Qualidade

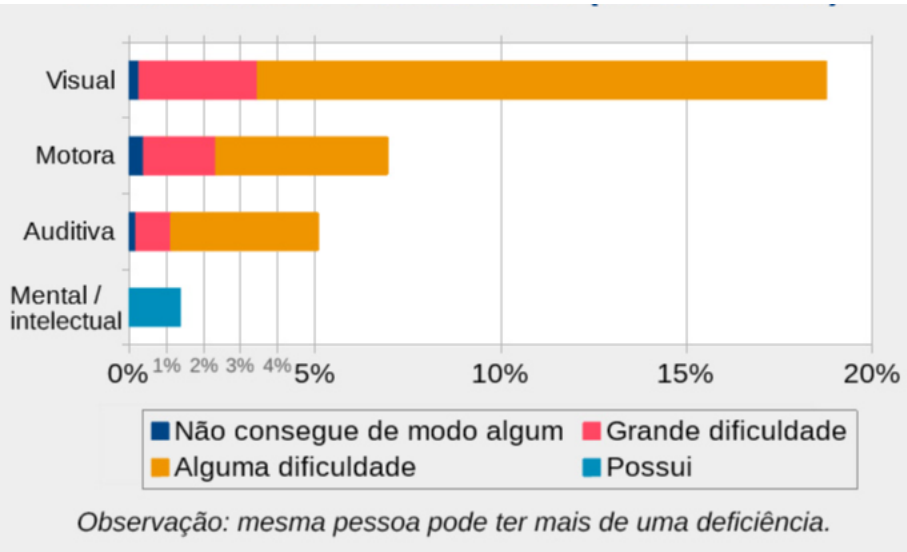
SUMÁRIO

1 INTRODUÇÃO

No ano de 2012 a Organização Mundial de Saúde (OMS) publicou um relatório indicando que no mundo todo existem mais de 1 bilhão de pessoas com algum tipo de deficiência, e desse total existem aproximadamente 253 milhões de pessoas com Deficiência Visual (DV) (OMS, 2012). Mais recentemente, no ano de 2019, a OMS divulgou um novo relatório indicando que o número de pessoas com DV aumentou para 2,2 bilhões de pessoas no mundo, sendo que em torno de 1 bilhão de casos poderiam ter sido evitados (OMS, 2019).

No Brasil, de acordo com o Censo de 2010 do Instituto Brasileiro de Geografia e Estatística (IBGE), quase 46 milhões de pessoas se declararam com algum grau de deficiência, podendo ser visual, auditiva, motora, mental ou intelectual, como pode ser visto na Figura 1, onde aproximadamente 8,64 milhões são pessoas com DV (IBGE, 2010).

Figura 1 – Porcentagem da população, por tipo e grau de deficiência



Fonte: IBGE (2010).

O termo DV diz respeito ao espectro que vai da Baixa Visão (BV), também conhecido como Visão Subnormal (VS), até a cegueira total, estas pessoas necessitam de equipamentos para auxiliar em algumas atividades simples, como subir uma escada ou ler um letreiro, estes equipamentos são conhecidos como Tecnologia Assistiva (TA) (IBGE, 2010).

Este termo TA, que vem do inglês *assistive technology* introduzido por meio de legislação nos Estados Unidos da América (EUA), ainda é algo recente e é usado para identificar toda uma biblioteca de recursos e serviços que ajudam a fornecer ou ampliar as capacidades funcionais de pessoas com deficiência e, assim, promover vida independente e inclusão, sendo

possível porque a evolução tecnológica está caminhando em direção de tornar as nossas vidas cada vez mais fáceis já que de forma constante estamos usando cada vez mais ferramentas que foram desenvolvidas especificamente para favorecer e simplificar as atividades do nosso dia (BERSCH, 2008).

Mesmo que o seu termo seja atual, as TAs são tecnologias bem antigas, e se observarmos a história é possível encontrar TAs como alguns mecanismos/instrumentos utilizados como forma de auxílio para pessoas fazerem alguma atividade, como o ábaco que é um instrumento usado para contar, provavelmente é o primeiro dispositivo mecânico criado para indivíduos com DV (SEBOLD; PEDROSA, 2020).

Citando o conceito ADA (*Americans with Disabilities Act*), Hussey e Cook (1995) definem TA como "uma ampla gama de dispositivos, serviços, políticas e práticas destinadas a aliviar problemas funcionais vivenciados por pessoas com deficiência".

De forma complementar, de acordo com Sebold e Pedrosa (2020) a TA é definida como um dispositivo ou providências que sejam capazes de prover algum resultado funcional para o usuário, e oferecer possibilidades de realizar tarefas nas quais sem a TA seriam difíceis ou impossíveis. Alguns exemplos de TA são:

- **A muleta**: que serve para dar apoio à pessoas com algum tipo de deficiência motora.
- **Aparelho de amplificação sonora**: para pessoas com deficiência auditiva.
- **Carros adaptados**: para pessoas que tiveram membros amputados.
- **Leitor de tela**: para pessoas com cegueira total utilizarem aparelhos eletrônicos com *Touch Screen* ou utilizarem computadores pessoas.
- **Ampliação de tela**: para pessoas com BV utilizarem aparelhos eletrônicos com *Touch Screen* ou utilizarem computadores pessoas.

Existem TAs para atividades que seriam impossíveis de exercer dependendo da deficiência, por exemplo: como seria possível uma pessoa com DV ler uma notícia em um site na *internet* utilizando um computador ou *smartphone* sem o leitor de tela, seria algo inviável (SOUSA, 2018).

Porém, com o avanço da tecnologia, as TAs também evoluíram e assim surgiram ferramentas que fazem a leitura da tela, ou seja, lê os elementos e textos da tela de um computador, ferramenta bastante utilizada por pessoas com cegueira total, ou a ampliação da tela, ferramenta bastante utilizada por quem tem BV (HERSH; JOHNSON, 2008).

Uma destas ferramentas que possibilitam uma pessoa com DV utilizar um computa-

dor é o *Dosvox*, que é constituído de um sistema com leitor de tela com sintetizadores de voz que faz a leitura dos elementos na tela do computador (FÁTIMA *et al.*, 2021). O *Dosvox* foi desenvolvido pela Universidade Federal do Rio de Janeiro (UFRJ) sendo uma tecnologia totalmente nacional e o primeiro sistema comercial a sintetizar textos em voz na Língua Portuguesa (SALES *et al.*, 2019).

Outras ferramentas mais conhecidas que fazem a leitura da tela são o *NonVisual Desktop Access* (NVDA) (NVDA, 2023) e o *Job Access with Speech* (JAWS) (JAWS, 2023), lendo: ícones no computador, arquivos *Portable Document Format* (PDF), arquivos de texto e ícones em páginas *Web*. Estas ferramentas são tão poderosas que possibilitam ao usuário estudar e trabalhar utilizando o computador, porém essas ferramentas têm uma certa diferença na questão de acesso à elas, onde o NVDA é um *software open source* e gratuito, o JAWS é pago (SUSANTO; NANDA, 2018).

O NVDA por ser um leitor de tela de código aberto, qualquer pessoa pode acessar seu código fonte e contribuir para sua evolução (NVDA, 2023). Atualmente, ele é compatível com os principais *softwares* de edição de texto e planilhas, navegadores *Web*, *e-mail* e mensagens instantâneas, além disso, por se tratar de uma ferramenta gratuita, tem se tornado cada vez mais popular entre os DV, ampliando sua difusão (VARGHESE; RATHNASABAPATHY, 2021).

Por outro lado, o JAWS é um leitor de tela comercial, com um custo elevado que exige uma licença específica para utilização, e a sua popularidade é decorrente da sua alta compatibilidade com praticamente todos os *softwares* disponíveis no mercado, permitindo total autonomia e independência para as pessoas com deficiência visual, e é compatível com *Windows* (JAWS, 2023).

Desse modo, é possível afirmar que a escolha entre o NVDA e o JAWS depende das necessidades individuais de cada usuário, onde o NVDA é mais acessível para aqueles que não têm condições financeiras de adquirir uma licença do JAWS, enquanto o JAWS se apresenta como uma ferramenta completa e altamente compatível com uma variedade de *softwares*, então, de maneira geral, ambos os leitores de tela permitem aos deficientes visuais o uso pleno das ferramentas tecnológicas e a inclusão digital (SILVEIRA; BEILER, 2012).

Tais ferramentas possibilitam pessoas com DV ingressar no mercado de trabalho em profissões que seriam consideradas impossíveis para eles exercerem, tal como a profissão de desenvolvedor de *software*. Além do desenvolvimento de *software*, Nascimento *et al.* (2022) observou que o profissional desenvolvedor também precisa utilizar outras ferramentas que fazem

parte do ciclo do desenvolvimento, tais como:

- Console de serviços, como o *Firebase*.
- Soluções de testes de *Web Services*, como o *Postman*.
- *Softwares* de gestão de equipe como o Jira e o Azure DevOps.
- *Softwares* de comunicação, como o *Slack*, *Discord* e o *Microsoft Teams*.

Para Pinto *et al.* (2019) as ferramentas precisam ter uma Interface Gráfica do Usuário (GUI) com alguns aspectos que são considerados principais, como: *design* do menu, ícones e *layout* da tela. Os menus possibilitam escolhas dos usuários de comandos ou opções de comandos e estes menus podem ser do tipo suspenso (*dropdown*), instantâneo (*pop-up*) ou de diálogo simples, os ícones podem ser identificadores de objetos, ações ou coisas e devem ser projetados de maneira que o usuário identifique de maneira fácil o seu significado para auxiliar na comunicação com o objetivo de sintetizar a informação, e o *layout* da tela deve dividir as tarefas nas diversas telas e como é apresentada as telas individuais (PREECE, 2005).

Uma outra TA já citada é a ampliação de tela, também conhecida como *zoom*, que auxilia usuários com BV a utilizar o computador ou *smartphone* retirando a necessidade de aumentar o tamanho da fonte das letras, evitando "quebrar" o *layout* das páginas *Web*, tendo em vista que com o *zoom* o usuário conseguirá ver todos os elementos da tela maiores, inclusive os textos.

Neste contexto, observando a dificuldade dos Desenvolvedores com Deficiência Visual (DDVs), este trabalho apresenta os resultados de uma análise feita com um desenvolvedor de *software* com BV para identificar as dificuldades utilizando as ferramentas *Visual Studio Code* (VSCode) e *Postman*, e se é possível um desenvolvedor com BV trabalhar com estas ferramentas com o auxílio da ferramenta *zoom*.

1.1 Objetivos

O objetivo geral deste trabalho é verificar e avaliar o nível de acessibilidade das ferramentas de desenvolvimento de *software* para um profissional desenvolvedor com BV, utilizando a ferramenta de ampliação de tela como auxílio, a fim de identificar as dificuldades encontradas e validar a adequação dessas ferramentas para uso diário por parte desse profissional. Como objetivos específicos estão:

a. Identificar quais requisitos tornam uma ferramenta de desenvolvimento de *software* acessível para pessoa com BV.

b. Identificar dificuldades para um desenvolvedor com BVs que utiliza as ferramentas de desenvolvimento com a ampliação de tela.

c. Desenvolver um *software* utilizando as ferramentas de desenvolvimento e de ampliação de tela no Sistema Operacional (SO) *Windows* 10 e Ubuntu 20.04 (com a versão do ambiente gráfico GNOME 3.36).

d. Avaliar se é possível um desenvolvedor com BV trabalhar com as ferramentas com pouca limitação.

2 FUNDAMENTAÇÃO TEÓRICA

Nesta seção, são apresentados os fundamentos que levaram este trabalho a ser desenvolvido, as características principais das tecnologias abordadas e exemplos de uso.

Inicia-se na seção 2.1, onde é abordado a DV de uma maneira geral, apontando algumas de suas características, impacto e dificuldades de uma pessoa com DV.

Em seguida, temos a seção 2.2 que traz um pouco da evolução e alguns exemplos de TA.

Logo na seção 2.3 o leitor irá encontrar um pouco sobre acessibilidade em geral e acessibilidade para DDVs.

Por fim, na seção 2.4 temos o exemplo de algumas ferramentas que um desenvolvedor com BV poderia utilizar para desenvolver *Application Programming Interface* (API)s.

2.1 Deficiência Visual

A DV é uma condição que afeta milhões de pessoas em todo o mundo e a OMS estima que cerca de 285 milhões de pessoas sofrem de algum tipo de DV, das quais 39 milhões são cegas e 246 milhões têm baixa visão (MEDEIROS, 2016). A DV pode ser causada por várias razões, incluindo problemas congênitos, lesões, doenças e envelhecimento, portanto ela pode afetar significativamente a qualidade de vida das pessoas, incluindo a capacidade de se comunicar, trabalhar, se locomover e realizar tarefas diárias (CASTRO *et al.*, 2008).

Existem diversas estratégias de intervenção para ajudar as pessoas com DV a melhorar sua qualidade de vida, uma das principais estratégias é a reabilitação visual, que inclui exercícios para fortalecer os músculos oculares, treinamento para melhorar a percepção espacial e adaptação do ambiente para facilitar a locomoção e o desempenho de tarefas diárias (WOLF; TAGLIETTI, 2019).

Outra estratégia importante é a utilização de TA, que inclui dispositivos e *softwares* que ajudam as pessoas com DV a se comunicar, navegar na *internet*, ler e escrever, entre outras atividades, e alguns exemplos de TA são as lupas eletrônicas, leitores de tela, teclados em braile e *softwares* de reconhecimento de voz (BERSCH, 2008).

A TA tem desempenhado um papel cada vez mais importante na melhoria da vida das pessoas com DV, este é um campo em rápido desenvolvimento, e muitos dispositivos e programas foram criados para ajudar as pessoas com DV a se comunicar, se locomover e realizar tarefas

diárias (NASCIMENTO *et al.*, 2015). Entre esses dispositivos temos as bengalas inteligentes, que usam sensores para ajudar os usuários a detectar obstáculos (INÁCIO; RAPHAEL, 2021), e os sistemas de reconhecimento de voz, que permitem que os usuários controlem seus dispositivos eletrônicos sem precisar tocá-los (MOURA, 2022).

Além disso, a tecnologia permite que as pessoas com DV tenham acesso às informações que normalmente eles não teriam, e um exemplo de tecnologia que permite tal feito são os leitores de tela, que convertem texto em fala ou Braille, permitindo que pessoas com DV acessem informações em livros, sites e outros tipos de documentos (FRIZZERA *et al.*, 2019). A acessibilidade dos serviços de transporte público também melhorou, com muitos ônibus e metrôs agora equipados com anúncios de áudio e em Braille para ajudar as pessoas com DV a se locomoverem com mais facilidade (BAGATINI, 2019).

No entanto, ainda há desafios significativos que precisam ser abordados no que diz respeito à DV. A acessibilidade de sites e aplicativos ainda é uma preocupação importante, com muitos deles não sendo totalmente acessíveis para pessoas com DV (MELO, 2021). Também é necessário fornecer mais treinamento e suporte para aqueles que trabalham com pessoas com DV, incluindo médicos, enfermeiros e professores, para que possam entender melhor as necessidades desses indivíduos e prestar cuidados adequados (SANTO; OLIVEIRA, 2018).

Outra área de pesquisa importante é o desenvolvimento de tecnologias mais avançadas para ajudar as pessoas com DV. Por exemplo, a tecnologia de visão artificial está sendo usada para ajudar as pessoas com DV a reconhecer objetos e rostos, e as próteses visuais estão sendo desenvolvidas para ajudar as pessoas cegas a recuperar algum grau de visão (CESARO *et al.*, 2019). À medida que a tecnologia continua a evoluir, é provável que novas soluções sejam desenvolvidas para ajudar as pessoas com DV a superar os desafios que enfrentam.

Por fim, a conscientização sobre a DV e a inclusão de pessoas com DV na sociedade também é essencial. É importante que as pessoas entendam as necessidades e desafios enfrentados por aqueles com DV e trabalhem para criar um ambiente mais inclusivo e acessível para todos.

2.2 Tecnologia Assistiva

A TA é uma área do conhecimento que se dedica a criar soluções tecnológicas para pessoas com deficiências ou dificuldades de acesso à informação, mobilidade e comunicação (GARDIN, 2014). A TA abrange uma grande variedade de dispositivos, sistemas e equipamentos que podem ser utilizados em diferentes contextos, como educação, trabalho, lazer, saúde e bem-

estar, onde o objetivo é promover a inclusão social e a autonomia dessas pessoas, permitindo que elas participem de atividades cotidianas e tenham uma vida mais independente e satisfatória (NASCIMENTO *et al.*, 2015).

A TA tem suas raízes na antiguidade, quando já se criavam instrumentos para auxiliar pessoas com DV, como o ábaco, ou deficiência motora, como as muletas, no entanto, foi somente no século vinte que o conceito de TA foi definido e começou a ser desenvolvido de forma mais sistemática (SEBOLD; PEDROSA, 2020).

O movimento pelos direitos das pessoas com deficiência, que ganhou força nas décadas de 1960 e 1970, foi um grande impulso para o desenvolvimento da TA, e a partir desse movimento, as pessoas com deficiência passaram a exigir mais acesso e igualdade de oportunidades, o que levou ao desenvolvimento de soluções tecnológicas para atender às suas necessidades (AMORIM *et al.*, 2019).

Em vista que nos anos 1980 e 1990, com o avanço da tecnologia digital, eletrônica, computação e da robótica, a TA se expandiu e se sofisticou, oferecendo cada vez mais opções de acessibilidade e interação, e hoje é possível encontrar desde simples lupas e bengalas até próteses robóticas, sistemas de reconhecimento de voz e aplicativos para *smartphones* (BERSCH, 2008). Nos anos 2000, com o surgimento de tecnologias como a Inteligencia Artificial (IA) e a Internet das Coisas (IdC), a TA tem se expandido ainda mais, com o desenvolvimento de soluções cada vez mais sofisticadas e integradas (COSTA *et al.*, 2022).

Hoje em dia, a TA é uma área em constante evolução, com o desenvolvimento de soluções cada vez mais acessíveis, eficazes e integradas às necessidades dos usuários. A TA é uma área que oferece grande potencial para melhorar a qualidade de vida e a inclusão social de pessoas com deficiência ou com dificuldades de acesso à informação, mobilidade e comunicação (GARDIN, 2014).

Os benefícios da TA são muitos e variados, tanto para as pessoas com deficiência como para a sociedade em geral. Alguns exemplos incluem o aumento da independência e da autoestima dos usuários, a ampliação das oportunidades de trabalho e educação, a redução do isolamento social e da discriminação, a melhoria da qualidade de vida e da saúde mental, além do potencial de aumentar a produtividade e a criatividade nas empresas (SANTOS, 2021).

Outro benefício da TA é a possibilidade de melhorar a acessibilidade e a mobilidade das pessoas com deficiência, permitindo que elas possam se locomover e realizar atividades cotidianas com mais facilidade e segurança, utilizando equipamentos como bengalas eletrônicas,

próteses robóticas e exoesqueletos que podem proporcionar mais independência e autonomia para pessoas com deficiência física ou visual (INÁCIO; RAPHAEL, 2021).

Em resumo, a tecnologia assistiva é uma ferramenta poderosa para promover a inclusão e a igualdade de oportunidades para as pessoas com deficiência. Ela pode melhorar a qualidade de vida (CASTRO *et al.*, 2008), a educação , o trabalho (NASCIMENTO *et al.*, 2015), a comunicação, a mobilidade e a interação social dos usuários, permitindo que eles participem ativamente da sociedade (INÁCIO; RAPHAEL, 2021).

Apesar dos avanços e benefícios, a TA ainda enfrenta desafios e limitações que precisam ser superados. Um dos principais desafios é o acesso aos dispositivos e equipamentos, que podem ser caros e de difícil distribuição em países em desenvolvimento, como, por exemplo, o leitor de tela JAWS que requer uma licença (JAWS, 2023), ou as bengalas inteligentes que são equipamentos mais modernos e de não fácil acesso (INÁCIO; RAPHAEL, 2021).

Além disso, há questões de padronização, compatibilidade e segurança que precisam ser consideradas na elaboração e uso da TA (PINTO *et al.*, 2019). Também é importante lembrar que a TA não pode resolver todos os problemas e limitações das pessoas com deficiência, e que é necessário abordar questões sociais e culturais para promover a inclusão plena e continuar investindo em pesquisa e desenvolvimento de tecnologias cada vez mais eficazes e acessíveis, além de garantir que as soluções existentes sejam atualizadas e adaptadas às necessidades dos usuários.

A TA é uma área fundamental para a inclusão e a acessibilidade das pessoas com deficiência, e tem um grande potencial para transformar suas vidas e as da sociedade em geral. Ao longo da história, a TA evoluiu e se diversificou, oferecendo uma ampla variedade de soluções para atender às necessidades específicas de cada usuário, utilizando tecnologias mais simples como as muletas para pessoas com deficiência física, ou um pouco mais avançadas como aparelhos auditivos para pessoas com deficiência auditiva (SEBOLD; PEDROSA, 2020) ou até mesmo **softwares** leitores de tela para pessoas com DV (NVDA, 2023). A evolução e a disseminação da TA são fundamentais para garantir que todas as pessoas tenham a possibilidade de participar plenamente da vida em sociedade, sem barreiras.

2.3 Acessibilidade

A acessibilidade em geral envolve a eliminação de barreiras físicas, sociais e culturais por isso é uma questão fundamental para garantir a inclusão e a igualdade de oportunidades para

todas as pessoas, independentemente de suas habilidades e limitações, portanto, no contexto da Tecnologia da Informação e Comunicação (TIC), a acessibilidade se torna ainda mais relevante, pois a maioria das atividades cotidianas envolve o uso de dispositivos eletrônicos, como computadores ou *smartphones*, e *softwares*, como ferramentas para edição de texto ou planilhas (SCHAFHAUZER; SILVA, 2022).

Desenvolvedores de *software* têm a responsabilidade de garantir que seus produtos sejam acessíveis para todos, incluindo pessoas com DV (PINTO *et al.*, 2019). Neste sentido, a TA desempenha um papel fundamental, permitindo que as pessoas com DV acessem e interajam com a tecnologia de forma independente e autônoma (ALVES, 2016). Este seção tem como objetivo explorar o tema da acessibilidade em geral e da acessibilidade para desenvolvedores com DV.

2.3.1 Acessibilidade para desenvolvedores com Deficiência Visual (DDVs)

A acessibilidade é um conceito que visa garantir que pessoas com deficiência possam usufruir dos mesmos direitos e oportunidades que as demais pessoas, por isso quando se trata de desenvolvedores com DV, refere-se à disponibilização de TAs e ferramentas que permitam que esses desenvolvedores possam produzir *software* com a mesma qualidade e eficiência que os demais desenvolvedores (PINTO *et al.*, 2019).

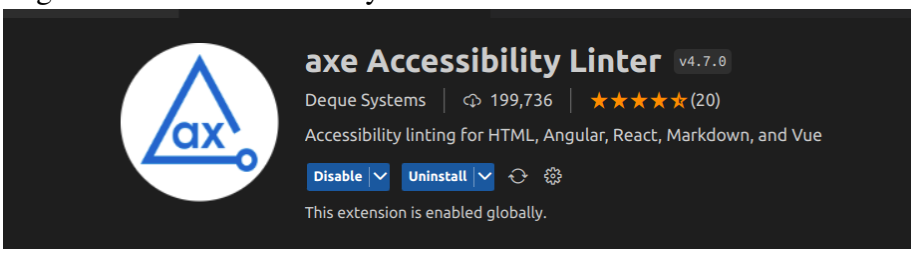
O surgimento das TAs, como *softwares* de leitura de tela, ampliadores de tela, sintetizadores de voz, entre outros, na década de 80 com o desenvolvimento dos primeiros sistemas de leitura de tela, tem sido fundamental para a evolução da acessibilidade, possibilitando que pessoas com DV (pessoas totalmente cegas) possam utilizar recursos, como computadores e *smartphones* (SCHAFHAUZER; SILVA, 2022).

Com o avanço da tecnologia, surgiram novas ferramentas e recursos que ampliaram as possibilidades de acessibilidade para pessoas com DV, como *softwares* de Reconhecimento Óptico de Caracteres (OCR), que permitem a conversão de textos impressos em formatos digitais, permitindo que pessoas com DV consigam “ler” documentos impressos que não estejam escritos em braile (MEMON *et al.*, 2020).

Deste modo é perceptível que a acessibilidade também tem sido cada vez mais valorizada na área de desenvolvimento de *software*, com a inclusão de diretrizes de acessibilidade em padrões e normas internacionais, como o *Web Content Accessibility Guidelines* (WCAG) e a norma ISO/IEC 40500 (RODRÍGUEZ *et al.*, 2019). Isso tem impulsionado a criação de

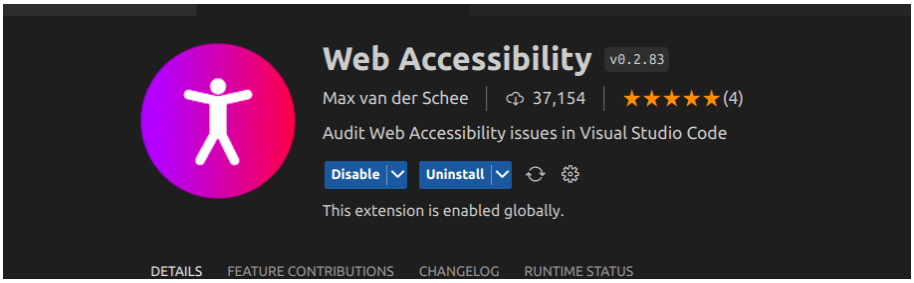
ferramentas e recursos específicos para desenvolvedores, como *plugins* e extensões para editores de código que permitem a detecção de problemas de acessibilidade em tempo real, como o *Axe Accessibility Linter*, Figura 2, e o *Web Accessibility*, Figura 3, encontrados no VSCode.

Figura 2 – Axe Accessibility Linter



Fonte: Elaborada pelo Autor.

Figura 3 – Web Accessibility



Fonte: Elaborada pelo Autor.

Portanto, a acessibilidade para DDVs traz alguns benefícios, tanto para os próprios desenvolvedores como para os usuários finais dos produtos desenvolvidos. Um destes benefícios é permitir que esses profissionais tenham acesso a tecnologias que antes eram inacessíveis para eles, como computadores e *smartphones*, e que esses desenvolvedores criem e testem produtos que atendam às suas necessidades específicas, sem que isso represente uma barreira para eles (CAMPOS *et al.*, 2013).

Além disso, a acessibilidade para DDVs pode levar a uma maior inovação e criatividade no desenvolvimento de produtos, como *softwares* leitores de texto mais bem evoluídos ou *softwares* de reconhecimento de objetos que podem ser usados em óculos que podem auxiliar pessoas cegas ou servir para melhorar a automação de carros inteligentes (LIMA *et al.*, 2022), e com isso poder trazer novas ideias e soluções para os desafios enfrentados pela indústria de tecnologia.

Apesar dos benefícios da acessibilidade para DDVs, ainda há desafios e limitações a serem superados. Um dos principais desafios é a falta de conscientização e conhecimento por parte dos desenvolvedores em relação à acessibilidade de suas aplicações.

Inúmeras vezes os desenvolvedores não estão cientes das necessidades específicas dos usuários com DV e, consequentemente, não criam interfaces adequadas para eles, uma tarefa complexa de se desenvolver especialmente quando se trata de plataformas complexas, como aplicativos móveis e páginas da *Web* com muitos recursos interativos, e para mitigar esse tipo de problema os desenvolvedores devem analisar os requisitos que tornam as suas interfaces acessíveis (PINTO *et al.*, 2019).

Além disso, o pouco conhecimento em padrões e diretrizes claras de acessibilidade pode dificultar ainda mais o processo de desenvolvimento de interfaces acessíveis, estas diretrizes de acessibilidade podem variar dependendo da plataforma e podem ser vagas ou contraditórias, dificultando o desenvolvimento de interfaces que atendam a todas as necessidades dos usuários com DV, entretanto, seguir as diretrizes do *World Wide Web* (W3C) pode ser um bom caminho para padronizar a acessibilidade das aplicações (FERREIRA JÚNIOR; PEREIRA, 2020).

Infelizmente, em alguns projetos a acessibilidade ainda é muitas vezes vista como um requisito opcional ou secundário em vez de uma necessidade essencial, podendo levar a um financiamento e prioridade baixas de recursos para a criação de interfaces acessíveis.

Observou-se que a acessibilidade para DDVs é um tema de extrema importância para a inclusão e igualdade no setor de tecnologia, em vista disso, é fundamental que haja um aumento na conscientização da importância da acessibilidade em geral, para garantir que todos possam ter acesso à tecnologia de forma igualitária, portanto, é importante que as empresas de TIC adotem práticas de acessibilidade e ofereçam recursos e ferramentas para os DDVs. Ao levar em consideração as necessidades dos DDVs, as soluções criadas tendem a ser mais abrangentes e adequadas para diferentes perfis de usuários, independente de suas habilidades e limitações, democratizando o acesso à tecnologia e promovendo a inclusão social (GARDIN, 2014).

2.4 Ferramentas de Desenvolvimento

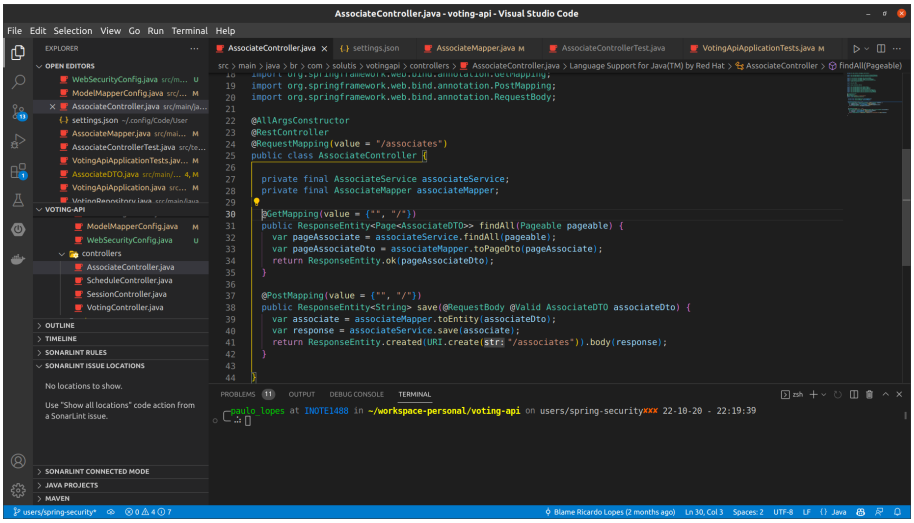
Nesta seção serão abordadas algumas das possíveis ferramentas que podem ser utilizadas por DDVs, o VSCode, o *Postman* e a ferramenta de ampliação (*zoom*). O VSCode é um editor de código de fonte aberta e multiplataforma desenvolvido pela *Microsoft*, que oferece diversas funcionalidades, como extensibilidade, integração com Git, *debug* e terminal integrado (VSCODE, 2022). O *Postman*, por sua vez, é uma ferramenta de colaboração, comumente usada como um cliente *Web*, que permite aos desenvolvedores testar, documentar e compartilhar APIs (POSTMAN, 2022). E por último, as ferramentas de ampliação são essenciais para

desenvolvedores com BV, permitindo que eles ajustem o tamanho do texto e das imagens em suas telas, tornando a leitura e a visualização mais confortáveis.

2.4.1 Editor Visual Studio Code

O VSCode é um editor de código-fonte da empresa *Microsoft* e possui compatibilidade com os SOs *Windows*, Ubuntu e macOS. Ele possui *IntelliSense* (capacidade de agilizar a boa escrita do código, possui coloração diferente nas palavras no código e função autocompletar), função de depuração de código, seu código-fonte foi versionado com o Git e possui diversos *Plugins* que auxiliam ao usuário desenvolver e testar códigos de maneira mais rápida e fácil (VSCODE, 2022). Na Figura 4 podemos ver como é a visão de um desenvolvedor utilizando o VSCode.

Figura 4 – Janela do VSCode



Fonte: Elaborada pelo Autor.

Como é possível observar, a janela do VSCode possui alguns botões do lado esquerdo, possui menus na sua parte superior (*file*, *edit*, *selection*, etc), um terminal integrado na parte centro inferior e o código para edição fica no centro do editor.

2.4.1.1 Botões auxiliares do lado esquerdo do editor

O primeiro botão visto na Figura 4 representa os arquivos do projeto. Ao clicar, ele exibe todos os arquivos do projeto caso eles não estejam sendo exibidos e quando arquivos são modificados ele mostra um aviso visual indicando quantos arquivos foram modificados mas não foram salvos.

O segundo botão é uma lupa e sua funcionalidade é de fazer uma pesquisa por um termo ou frase em todos os arquivos do projeto podendo deixar a pesquisa específica por palavra ou frase completa e aceitando até mesmo expressões regulares, com isso facilitando a busca.

O terceiro botão representa as modificações que estão sendo observadas por alguma ferramenta de versionamento de código, como, por exemplo, o Git ou o Mercurial.

O quarto botão é o de depuração, nele são exibidos as variáveis e seus conteúdos e os *breakpoints* adicionados no código. Um *breakpoint* é um ponto que o desenvolvedor pode marcar para o *software* parar ao chegar nesse ponto, facilitando ao desenvolvedor na depuração.

E o quinto é o das extensões, é nesse local que é possível pesquisar por *plugins* para facilitar no desenvolvimento de um *software*.

E dependendo das extensões instaladas, podem aparecer outros botões como: botões para visualizar os testes, visualizar *containers* ou imagens Docker, entre outros.

2.4.1.2 Terminal integrado

O editor VSCode também possui um terminal integrado, possibilitando ao desenvolvedor digitar linhas de comando para executar tarefas ou navegar nos diretórios. Também é possível inicializar novos terminais e alternar entre eles.

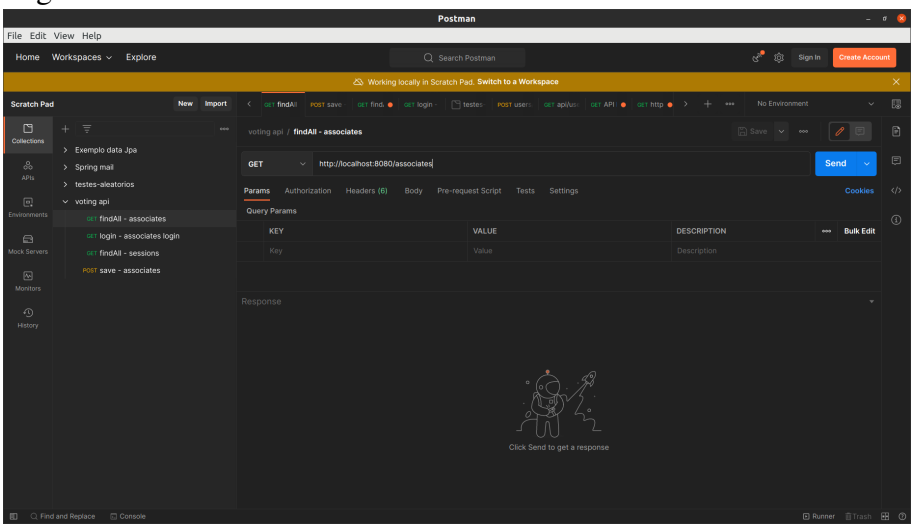
A utilização do terminal facilita ao desenvolvedor executar comandos sem a necessidade de utilizar o *mouse* para clicar em botões, o que pode aumentar a produtividade razoavelmente.

2.4.2 Postman

O *Postman* é uma ferramenta para cliente *Web* e onde comumente é utilizada para testar as rotas de uma API, é utilizada por mais de 20 milhões de desenvolvedores e pode ser instalada nos SOs *Windows*, Ubuntu e macOS. O *Postman* também é capaz de armazenar e gerenciar especificações de API, coleções de requisições, casos de teste, documentação, métricas e compartilhar essas informações com o seu time através dos espaços de trabalho de forma *online*.

Como é possível observar na Figura 5, o *Postman* possui diversos botões e campos de texto para o desenvolvedor elaborar seus testes da sua API no seu espaço de trabalho, porém todos esses elementos podem facilitar o trabalho de alguns desenvolvedores mas pode atrapalhar DDVs caso esses elementos não possuam um mínimo de acessibilidade.

Figura 5 – Janela do Postman



Fonte: Elaborada pelo Autor.

2.4.2.1 Campos auxiliares do lado esquerdo da ferramenta

O primeiro campo que podemos observar na Figura 5 é o campo *Collections*, onde é possível criar nossas coleções. Uma coleção é um conjunto de requisições com configurações variadas para o desenvolvedor testar a resposta da sua API à uma determinada requisição. As requisições são compostas de: os verbos *Hyper Text Transfer Protocol* (HTTP) (*GET*, *POST*, *PUT*, *DELETE*, *PATCH*, etc) e *Uniform Resource Locator* (URL), porém em alguns casos é necessário enviar *tokens* de autorização, objetos para cadastro no servidor em formato de *JavaScript Object Notation* (JSON) ou *eXtensible Markup Language* (XML), entre outros campos e funcionalidades.

Postman possui uma documentação bastante rica e através dela é possível aprender como usá-lo, caso o desenvolvedor seja um iniciante. Possui suporte a diversos tipos de *tokens* (no caso da API exigir uma autorização de acesso por *token*), também permite o envio de arquivos em anexo para testes de serviços onde a sua API necessite enviar algum anexo, como, por exemplo, em serviços de *e-mail*. Uma ferramenta muito poderosa que serve para auxiliar desenvolvedores nos testes de suas APIs.

2.4.3 Ferramenta de ampliação

Uma outra ferramenta adotada por DDVs, em particular os com BV, na hora de trabalhar com desenvolvimento de *softwares* é a ferramenta de ampliação. Esta é uma ferramenta encontrada nativamente nos mais populares sistemas de computadores (SO), porém cada

SO possui a sua implementação e possíveis configurações e comportamentos diferentes desta ferramenta.

2.4.3.1 Ferramenta de aplicação no Ubuntu

A ferramenta de ampliação pode ser encontrada nas configurações do sistema, no menu de acesso universal, com o nome de *zoom*, onde também é possível encontrar outras ferramentas de acessibilidade como leitor de tela e alto contraste. O *zoom* pode ser ativado ou desativado no seu menu, mas deste modo não é muito funcional, onde a melhor opção é utilizar os atalhos de teclado. O atalho de teclado padrão para ativar e desativar o *zoom* é a combinação das teclas “Alt + Super + 8” no Ubuntu podendo ser modificado nas configurações dos atalhos de teclado.

O *zoom* possui uma ampliação padrão podendo ser incrementada ou decrementada com o atalho padrão de teclado “Alt + Super + =” ou “Alt + Super + -”, com isso podendo regular a dimensão do *zoom* a depender do nível de BV do usuário.

O comportamento do *zoom* depende da configuração escolhida, ele pode seguir o cursor do *mouse* mantendo o cursor fixo no centro da região ampliada acompanhando o cursor do *mouse*, deixar o conteúdo fixo e navegar com o cursor ao redor. Além de ser possível também usar o *zoom* em modo tela cheia, ou apenas no topo da tela, ou apenas na parte de baixo da tela, ou na parte esquerda ou direita da tela.

Adicionalmente, podemos observar o comportamento diferente do foco do *zoom* quando o cursor está em aplicações diferentes. Um exemplo de um comportamento de falta de acessibilidade é quando o *zoom* está ativado e o desenvolvedor digita algum texto utilizando a ferramenta *Spring Tools Suite* (STS), essa ação faz com que o foco do *zoom* mude sua posição para o canto superior esquerdo, impossibilitando ao desenvolvedor de acompanhar o que está digitando. Entretanto, quando o desenvolvedor digita comandos no terminal, o foco do *zoom* acompanha o cursor de texto, facilitando na produtividade ao digitar *shell script* no terminal. Porém, majoritariamente o seu comportamento em outras aplicações é manter a sua posição parada, não seguindo o cursor.

2.4.3.2 Ferramenta de aplicação no Windows

A ferramenta de ampliação no *Windows* é encontrada no menu de acessibilidade e tem o comportamento bastante parecido com a ferramenta do Ubuntu, porém com algumas

diferenças. O atalho padrão para ativar o *zoom* é “Tecla *Windows* + =” ou “Tecla *Windows* + +” (tecla + do teclado numérico), onde a “Tecla *Windows*” é a mesma tecla “Super” no Ubuntu. Para desativar o *zoom* o atalho padrão é “Tecla *Windows* + Esc”, logo é possível observar que as combinações de teclas para ativar e desativar o *zoom* são diferentes, onde no Ubuntu a mesma combinação de teclas que ativa o *zoom* é a mesma que o desativa.

Uma grande diferença é que, diferente do *zoom* do Ubuntu, no *Windows* existe uma configuração onde o foco do *zoom* costuma seguir sempre o cursor, facilitando ao usuário digitar textos e não ter a necessidade de usar o *mouse* para corrigir a posição do foco do *zoom* quando estiver digitando.

Outra peculiaridade no *zoom* do *Windows* é a configuração que habilita o seu foco seguir o foco do cursor ao pressionar a tecla “tab”.

Por fim, também é possível configurar se, ao ativar o *zoom*, as cores da tela ficaram invertidas, onde for preto ficará branco, onde for branco ficará preto, o azul ficará vermelho e assim por diante.

2.4.3.3 Comparativo entre as ferramentas de ampliação

Após utilizar as duas ferramentas nos diferentes sistemas, conclui-se que as duas possuem prós e contras, onde o *zoom* no Ubuntu apresenta alguns *bugs* (*bug* é como é conhecido um erro em um *software*) dependendo da aplicação em que o usuário esteja utilizando, a ferramenta no *Windows* não demonstrou tal problema. Porém, o fato de ser possível alterar o atalho de teclado de maneira simples para ativar e desativar o *zoom* no Ubuntu é um ponto muito forte porque deixa mais cômodo para o usuário, um exemplo disso é salvar o atalho como “Super + z” facilitando ativar e desativar o *zoom* com apenas uma mão.

Entretanto, no *Windows* o *zoom* segue o foco do cursor facilitando ao usuário digitar textos mais longos sem a necessidade de tirar as suas mãos do teclado para mover o *mouse* para centralizar o foco do *zoom* onde estiver o cursor.

Por fim, as duas ferramentas cumprem a sua função e qualquer usuário com BV irá conseguir ler os textos na tela do computador, o que pode variar são as configurações e alguns comportamentos que podem ou não atrapalhar dependendo da aplicação em uso.

3 TRABALHOS RELACIONADOS

Nesta seção são apresentados os trabalhos que vieram a servir como apoio e contribuíram para o desenvolvimento desta pesquisa. Serão mostradas as semelhanças e diferenças com relação à proposta apresentada neste trabalho.

Na seção 3.1 é apresentado um trabalho onde o autor identifica as dificuldades de DDVs brasileiros com o desenvolvimento de *software* ao utilizar ferramentas comuns no dia de um desenvolvedor.

O trabalho na seção 3.2 pesquisa a acessibilidade na Engenharia de *Software* buscando analisar dezenas de artigos e classificar técnicas de acessibilidade na *Web*.

Na seção 3.3 é apresentado um trabalho onde o público alvo do autor foram as pessoas com BV, onde ele buscou propor requisitos de projeto dos elementos da interface gráfica do Objetos de Aprendizagem (OA) do *software* HyperCAL^{3D}.

Por fim, na seção 3.4, é apresentado qual a semelhança e o adicional deste trabalho corrente com os três trabalhos supracitados.

Adicionalmente, notou-se que a ideia de acessibilidade é algo bastante amplo e requer bastante atenção, onde é possível identificar pouca ou a falta de acessibilidade em algumas ferramentas usadas no trabalho ou nos estudos na vida de um desenvolvedor de *software* com deficiência visual.

3.1 Programando às cegas: investigando a acessibilidade de ambientes de desenvolvimento de software

Em Nascimento *et al.* (2022) o autor identificou as dificuldades dos DDVs e quais as estratégias usadas pelos mesmos para transpor as barreiras no âmbito do desenvolvimento em geral procurando sempre lançar novos requisitos ou para a criação de novos recursos de acessibilidade ou para aprimorar ainda mais os requisitos existentes, chegando ao ponto que essas investigações geraram melhorias para alguns Ambiente de Desenvolvimento Integrado (IDE).

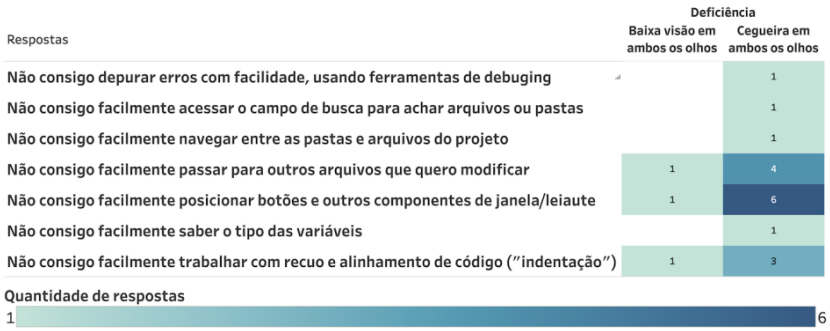
Neste contexto, o trabalho propôs uma pesquisa com DDVs brasileiros e fez uma análise de ferramentas que fazem parte do ciclo de desenvolvimento de *software*. Como primeira etapa desta pesquisa foi aplicado um questionário para dez DDVs e também foram avaliadas quatro ferramentas *Web* (Jira, *Postman*, Bugzilla, *Firebase*). O questionário foi dividido em

duas partes, a primeira continha perguntas sobre o perfil do desenvolvedor (nome, *e-mail*, etc) e também com algumas questões sobre a sua atuação profissional (TAs que usava, área em que atua, etc), e a segunda parte estava dividida em função das dificuldades de acessibilidade e as etapas do ciclo de desenvolvimento de um *software*. Concluiu-se que eles, os DDVs encontram algumas dificuldades ao trabalhar com desenvolvimento, tais como:

- Dificuldade em escrever, revisar e compreender os códigos.
- Dificuldades ao posicionar elementos gráficos em um *layout*.
- Dificuldade em alterar entre arquivos para edição.
- Dificuldade ao navegar entre as pastas de um projeto.
- Dificuldade em lembrar o tipo de variável no meio do código.
- Dificuldade ao depurar erros no código.

Na Figura 6 é possível visualizar algumas das respostas obtidas na pesquisa.

Figura 6 – Dificuldades ao escrever o código da aplicação.



Fonte: Nascimento *et al.* (2022).

O trabalho de Nascimento *et al.* (2022) mostrou que o ferramental envolvido no ciclo de desenvolvimento de *software* ainda possui diversas barreiras para os DDVs e tais barreiras apareceram em diversos contextos de atuação, como no mercado de trabalho, no meio científico ou em projetos pessoais, em vista que os DDVs precisam acessar os mesmos recursos que outros desenvolvedores têm a disposição para que sejam incluídos de forma satisfatória as metodologias atuais do desenvolvimento de *software*.

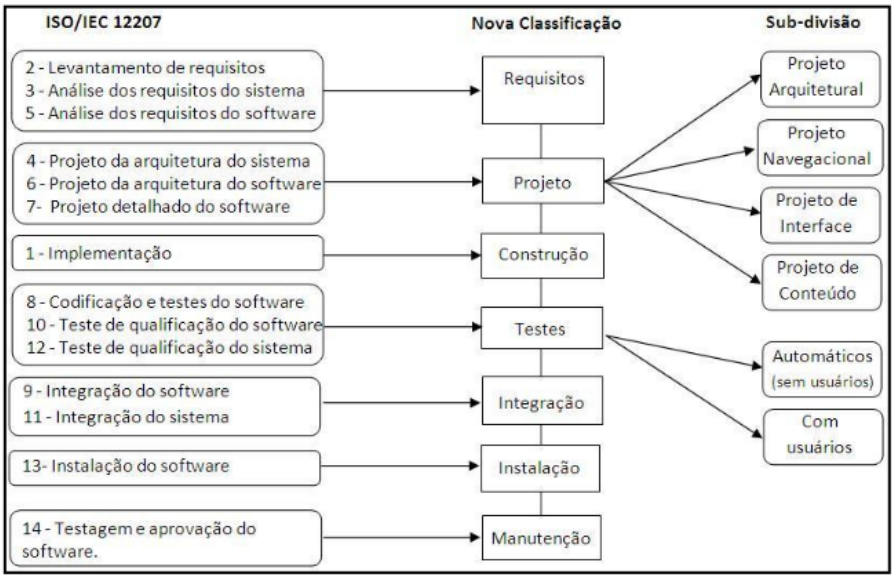
Um diferencial entre o trabalho de Nascimento *et al.* (2022) e este, é que neste a pesquisa será sobre a existência ou não existência de acessibilidade que um desenvolvedor de *software* com BV irá encontrar ao desenvolver um *software* utilizando somente a ampliação de tela nativa do SO, utilizando um editor de código-fonte para desenvolver e debugar o código e um cliente *Web* para os testes de desenvolvedor.

3.2 Uma Revisão Sistemática sobre a inserção de Acessibilidade nas fases de desenvolvimento da Engenharia de Software em sistemas Web

Em Dias *et al.* (2010) é apresentado a ideia de acessibilidade na *Web* como a possibilidade de qualquer pessoa, que possua um agente (*software* ou *hadware* que recupera e serializa conteúdo *Web*), possa entender e interagir com conteúdos da *Web*. O objetivo deste trabalho foi efetuar um levantamento e classificação de técnicas para acessibilidade na *Web* com base em um grupo de processos de Engenharia da Norma ISO/IEC 12207, utilizando procedimentos de Revisão Sistemática (RS). A RS é a produção da síntese completa de trabalhos publicados sobre um tema específico e se utiliza de um processo aberto e bem definido.

Foram analisados 65 artigos e estes foram classificados como técnica de apoio a um ou mais processos de uma Nova Classificação, como mostra na Figura 7.

Figura 7 – Classificação realizada a partir da NORMA ISO/IEC 12207.



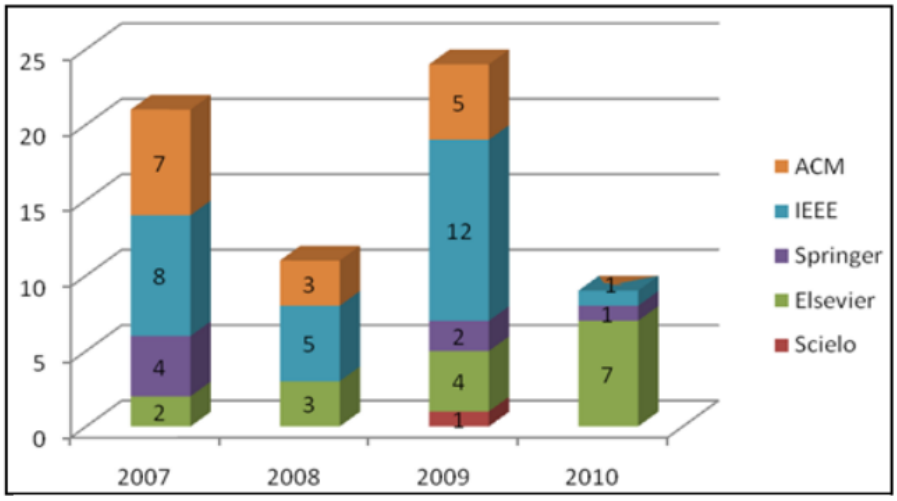
Fonte: Dias *et al.* (2010).

Os trabalhos foram classificados nas seguintes fases: requisitos, projeto, construção, testes, integração, instalação e manutenção. Constatou-se que não há nenhuma pesquisa sobre acessibilidade que apoia o processo de instalação de *software*, bem como os processos de projeto arquitetural, construção e manutenção foram os que tiveram menos trabalhos envolvidos. O processo que mais obteve trabalhos foi o de teste com usuário.

Como resultado, pode-se verificar um crescimento de trabalhos à respeito de acessibilidade na *Web*, como é possível observar na Figura 8, que mostra as publicações em cada portal de busca. O portal da Instituto de Engenheiros Eletricistas e Eletrônicos (IEEE) é a que

lidera em publicações.

Figura 8 – Ano de publicação dos artigos selecionados.



Fonte: Dias *et al.* (2010).

O trabalho apresentado possui interesse em acessibilidade na *Web* e este tema vem crescendo a cada ano desde 2007, portanto, é natural que acessibilidade para *Web* seja maior a cada ano e que a acessibilidade é essencial para uma boa experiência de usuário.

Como visto, o trabalho de Dias *et al.* (2010) é um estudo sobre trabalhos que têm o foco principal a acessibilidade na *Web*, indicando que o tema está bem forte na literatura, e o diferencial do trabalho proposto é que este experimento não busca epenas verificar contribuições para o tema mas também contribuir com o ponto de vista de um desenvolvedor com BV se as ferramentas utilizadas no mercado de trabalho estão buscando manter o seu *software* acessível.

3.3 Requisitos de Projeto de Interfaces Gráficas de Objetos de Aprendizagem Acessíveis para Usuários com Baixa Visão

O trabalho de Pinto *et al.* (2019) possui os resultados de uma pesquisa que teve como objetivo propor requisitos de projeto dos elementos da interface gráfica de OA digitais que concedem acessibilidade aos usuários com BV, onde a proposta da pesquisa foi a aplicação do estudo da interface gráfica do OA do *software* HyperCAL^{3D}.

A metodologia desse trabalho teve como base o método de condução da *Design Science Research* (DSR) e tal método busca, a partir de um determinado problema real, a construção e avaliação de artefatos que solucionem determinada situação, e o desenvolvimento da pesquisa seguiu os seguintes procedimentos:

- Fundamentação teórica a partir de pesquisa bibliográfica.
- Revisão Sistemática da Literatura.
- Análise dos dados da pesquisa bibliográfica.
- Avaliação de similares através da ferramenta de análise sincrônica.
- Entrevista com usuários.
- Aplicação do método Desdobramento da Função Qualidade (QFD).
- Teste com usuários.

Como resultados o autor encontrou algumas barreiras para usuários com BV no acesso à GUI e agrupou tais barreiras em sete tópicos:

- Contraste inadequado entre plano de fundo e primeiro plano.
- Uso de fonte serifada, decorada ou com sombra.
- Conteúdos que não podem ser redimensionados, quebram ou desconfiguram ao serem ampliados.
- Acúmulo de botões e menus em um único ponto.
- Não identificação de conteúdos dinâmicos.
- Interface gráfica poluída visualmente.
- Falta de padronização da interface.

Logo, após uma análise foi levantada a necessidade de cada barreira somado aos princípios, diretrizes e orientações de acessibilidade para usuários com BV, chegando a elaborar 17 requisitos para à acessibilidade de interface para usuários com BV, onde vai de apresentar um visual mais simples e limpo até possibilitar ampliação.

Neste contexto, baseado nos requisitos apresentados por Pinto *et al.* (2019) que visa bons *layouts* da tela (melhor escolha de cores, posicionamentos e títulos intuitivos) e a possibilidade de ampliar os objetos para uma melhor visualização de um usuário com BV, temos o diferencial que neste trabalho abordou uma maneira diferente de um usuário com BV usar um *software*, onde, mesmo que o usuário encontre um *software* que não atenda aos requisitos supracitados, não há a necessidade de forçar os objetos na tela a aumentarem de tamanho pois o usuário estará utilizando a ferramenta de ampliação de tela. Como visto nos resultados na seção 5, não seria possível forçar o aumento de tamanho dos objetos dos *softwares* utilizados no experimento (exceto a fonte do texto) e sem a ampliação de tela seria inviável um usuário com BV trabalhar de maneira fluida com estas ferramentas comuns no mundo do desenvolvimento de *software*, por isso houve a necessidade dessa outra abordagem.

3.4 Comparativo dos trabalhos

Quadro 1 – Comparação entre os trabalhos

Trabalho	**Utilizou a ferramenta de ampliação nativa do Sistema Operacional**	**Pesquisa voltada para o público Baixa Visão**	**Fez o experimento em mais de um Sistema Operacional**	**Simulou o fluxo completo de um profissional desenvolvedor de *software***
Nascimento *et al.* (2022)	Não	Sim	Sim	Sim
Dias *et al.* (2010)	Não	Sim	Sim	Não
Pinto *et al.* (2019)	Não	Sim	Não	Sim
Trabalho Proposto	Sim	Sim	Sim	Sim

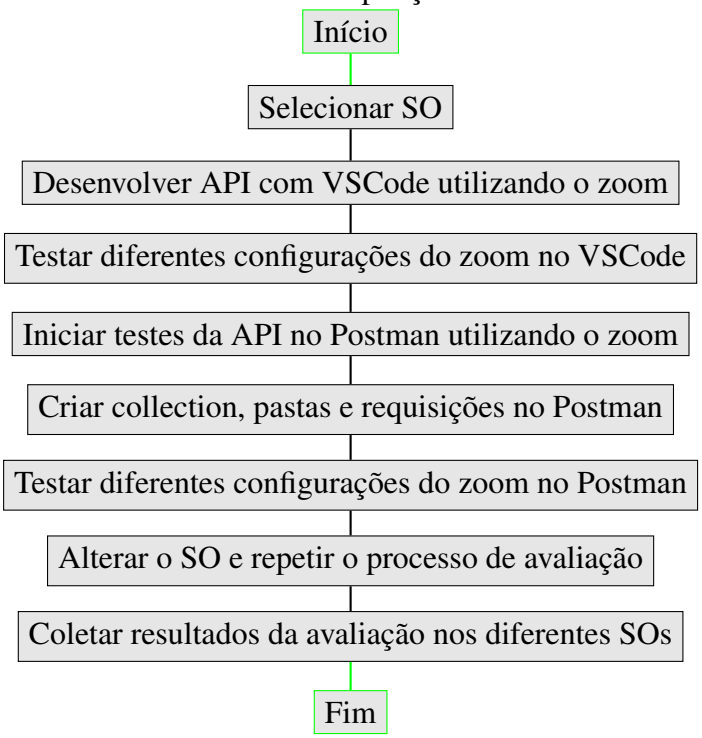
Fonte: Elaborado pelo Autor.

Quando se trata de buscar os elementos de acessibilidade em ferramentas que são utilizadas por DDVs todos os trabalhos fazem isso com clareza, e este trabalho procurou observar as técnicas e requisitos abordados nos três trabalhos citados com o diferencial que este experimento foca no usuário com BV, na utilização de uma TA gratuita e de fácil acesso e vivenciar o fluxo completo do desenvolvimento de um *software* com duas ferramentas de desenvolvimento utilizadas de forma comum no dia de um desenvolvedor em dois SOs diferentes.

4 PROCEDIMENTOS METODOLÓGICOS

Este trabalho tem como objetivo avaliar as ferramentas VSCode e Postman para desenvolvimento de *software* utilizando a ferramenta de ampliação utilizada por usuários com BV, identificando as limitações e dificuldades que um profissional com BV pode encontrar dependendo do SO e da ferramenta de desenvolvimento.

Figura 9 – Diagrama de procedimentos metodológicos para desenvolver e testar API com ferramentas de ampliação de diferentes SOs.



Fonte: Elaborada pelo Autor.

Iniciando na seção 4.1, é feito uma análise do comportamento do *zoom* ao utilizar o VSCode nos SOs Ubuntu e *Windows*, para avaliação de acessibilidade neste editor de código-fonte e uma comparação para identificar se existe diferença de comportamento em cada SO.

Em seguida, na seção 4.2, são feitos testes de usabilidade fazendo requisições em uma API, desenvolvida utilizando o VSCode, criada pelo próprio autor para verificar o comportamento do *zoom* nos dois SOs citados anteriormente, com o intuito em verificar a acessibilidade e identificar as possíveis dificuldades que os DDVs com BV podem ter ao utilizar o *Postman*.

Por fim, temos a seção 4.3 onde é apresentado os paradigmas de como foi feita a avaliação e os resultados são coletados e apresentados em Resultados (5), na seção 5.3.

4.1 VSCode como ferramenta acessível para edição de código-fonte

Para obter uma melhor compreensão da solução, o experimento foi dividido em duas etapas: o desenvolvimento da API no Ubuntu, e o desenvolvimento no *Windows*. Foi desenvolvido uma aplicação utilizando apenas o VSCode, como editor de código-fonte, e o *zoom*, como ferramenta de auxílio para pessoa com BV para poder ser testado a acessibilidade utilizando as ferramentas supracitadas.

4.1.1 Desenvolvendo nos diferentes SOs

Foi implementado uma API utilizando a linguagem de programação Java e o *framework Spring Boot* e seu desenvolvimento foi todo criado no editor de código-fonte VSCode com o auxílio do *zoom* para ser possível o desenvolvedor com BV conseguir ver os códigos, os botões e as listas de menus. Ao decorrer da implementação da API, foi testado o comportamento do *zoom* ao clicar em algum menu do editor, ao percorrer as opções com a tecla "tab" e ao digitar o código.

Em seguida foi observado o nível de dificuldade do usuário ao instalar novas extensões no editor e se o mesmo apresentou algum tipo de comportamento que impeça o usuário utilizar a ferramenta com o *zoom*.

Também foi testado clicar em botões que novas extensões criam na interface do editor e como se comporta o terminal integrado comparado ao terminal nativo do SO.

Todos os testes foram feitos em dois diferentes SOs, o Ubuntu e o *Windows*, onde foi avaliado as diferentes configurações do *zoom* nos diferentes SOs, onde foi observado um número maior de configurações para o *zoom* no *Windows* comparado ao *zoom* do Ubuntu. Porém, cada *zoom* possui sua peculiaridade e foram obtidos resultados diferentes na avaliação, como pode ser visto na seção 5.

4.2 Postman como ferramenta acessível para testes em uma API

Postman é uma ferramenta muito poderosa utilizada por profissionais da TIC para testar uma API através de requisições HTTP simulando um cliente *Web*.

Após a instalação nos dois SOs utilizados neste trabalho, foi utilizado o *zoom* como uma ferramenta auxiliar para o usuário com BV conseguir navegar no *Postman* e ler as respostas das requisições.

Foram criadas *collections*, pastas e requisições para testar a API desenvolvida, e no decorrer dos testes as configurações do *zoom* foram alternadas para avaliar a experiência do usuário nos diferentes SOs, identificando alguns comportamentos diferentes de cada *zoom* em cada configuração.

Os testes foram realizados com base em critérios específicos para garantir a acessibilidade da aplicação. Os critérios utilizados para avaliar a acessibilidade foram baseados em diretrizes e padrões de acessibilidade, como as estabelecidas pelo WCAG e pelas melhores práticas de *design* inclusivo. Esses critérios envolvem aspectos como contraste de cores, estrutura lógica dos elementos, rótulos descritivos para campos e controles, e uso adequado de alternativas textuais para elementos visuais.

Para testar a acessibilidade, foram executadas diversas atividades e cenários, incluindo:

- Navegação pela interface do *Postman*: Foi testado a capacidade de navegar pelos diferentes botões e guias da aplicação, utilizando atalhos de teclado e interagindo com os elementos de interface.
- Criação de *collections* e requisições: Foi verificado se as ações de criação e configuração de *collections* e requisições eram acessíveis, garantindo que os campos e opções estivessem adequadamente identificados e que as interações fossem claras.
- Configuração da ferramenta de ampliação do *Windows* (*zoom*): Foram ajustadas diferentes configurações da ferramenta de ampliação do *Windows* para simular a experiência de um usuário com BV. Foi testado se a ampliação afetava a usabilidade do Postman, verificando se o conteúdo permaneceu legível e se os elementos de interface foram redimensionados corretamente.

Essas atividades e cenários foram executados em múltiplas iterações, permitindo que fosse possível revisar e refinar a acessibilidade da aplicação com base nos resultados dos testes, documentando as falhas identificadas. No *Windows* foram feitos mais testes devido ao seu *zoom* possuir um número maior de diferentes configurações, o que não necessariamente seja algo bom, como pode ser visto nos Resultados na seção 5.

4.2.1 Testes utilizando o Postman nos SOs

Foi testado a criação de novas *collections* e a criação de pastas na *collection* para uma melhor organização das requisições. Uma *collection* nada mais é do que uma coleção, um

conjunto, um grupo de pastas ou requisições. A *collection* pode ser vista como um espaço de trabalho.

O *Postman* também permite a criação de um ambiente para podermos organizar nossas variáveis de ambiente e facilitar a troca de valores de *host*, *tokens*, etc. Após toda a configuração do ambiente foram testadas algumas requisições com verbos HTTP. Observe que toda essa personalização foi testada para observarmos o comportamento do *Postman* ao ser utilizado com o *zoom*.

Por fim, foi testado exportar e importar uma *collection* através de um arquivo JSON para observar se o procedimento é ou não algo simples para alguém que tenha dependência do *zoom* para a utilização da ferramenta.

4.3 Avaliação

Foram realizadas avaliações das ferramentas por uma pessoa de um grupo específico de usuários, um DDVs com BV que possui acuidade visual entre 15% e 10%, com o objetivo de verificar a usabilidade das ferramentas de desenvolvimento de *software* e identificar possíveis problemas ou melhorias ao utilizá-las com o *zoom*. Esse perfil permitiu obter *insights* valiosos tanto sobre a acessibilidade das ferramentas quanto sobre a experiência do usuário em geral.

Durante a avaliação, foram estabelecidos critérios de usabilidade e acessibilidade, baseados em diretrizes reconhecidas internacionalmente, como as estabelecidas pelo WCAG e as melhores práticas de *design* inclusivo. Esses critérios abrangiam aspectos como a clareza das informações apresentadas, a facilidade de navegação com o *zoom* ativado e o uso de legendas para elementos visuais e botões.

Os cenários de avaliação foram projetados para abranger diversas configurações do *zoom* e fluxos de interação das ferramentas. Eles incluíram desde tarefas simples, como a realização de uma busca de arquivos no VSCode, até atividades mais complexas, como a criação de *collections* e requisições com parâmetros. Os cenários foram elaborados de forma a refletir situações reais de uso e permitir uma análise abrangente da experiência do usuário em diferentes contextos.

Para a avaliação no SO Ubuntu, foi utilizado a única configuração disponível do *zoom*, onde o *zoom* fica fixo com o cursor do *mouse*, e modificado o atalho de teclado de ativar/desativar o *zoom* para as teclas “Super + z” para melhorar a na experiência de usuário, já que o atalho padrão é “Alt + Super + 8” obrigando ao usuário a usar as duas mãos para

ativar/desativar o *zoom*, fazendo com que sempre seja necessário retirar a mão do *mouse* ao navegar nas telas.

Após alterar o atalho de teclado, foi avaliado o comportamento do *zoom* no VSCode ao criar pacotes, criar arquivos, buscar arquivos, instalar extensões, editar o código e digitar no terminal integrado. Todas essas ações foram executadas de forma exaustiva simulando um ambiente real, onde o DDVs precisará usar o VSCode para desenvolver a sua API, criando novas *features* e refatorando códigos para uma manutenção, quando necessário.

Com o *Postman* no Ubuntu a configuração do *zoom* continuou a mesma padrão, apenas com a alteração do atatalho do teclado e foi executados diversas ações como: criar *collections*, requisições, pastas, variáveis de ambiente, alternar o ambiente e analisar a resposta recebida verificando o *body* e o *headers*.

Após toda a avaliação do *zoom* no Ubuntu, foi feita a avaliação do *zoom* no SO *Windows*. No *Windows* o *zoom* possui uma variedade de configurações e no decorrer da criação da API as configurações foram alternadas para a avaliação de usabilidade e experiência de usuário, identificando pontos positivos e negativos de cada configuração nas ferramentas VSCode e *Postman*. Os critérios avaliados foram os mesmos nos dois SOs, e os testes no *Windows* foram feitos exaustivamente como no Ubuntu.

Ao combinar cenários representativos com um perfil diversificado de usuário e critérios de avaliação bem estabelecidos, foi possível obter uma visão abrangente das ferramentas e identificar aspectos a serem aprimorados, tanto em termos de usabilidade quanto de acessibilidade, como pode ser visto na seção 5.3.

5 RESULTADOS

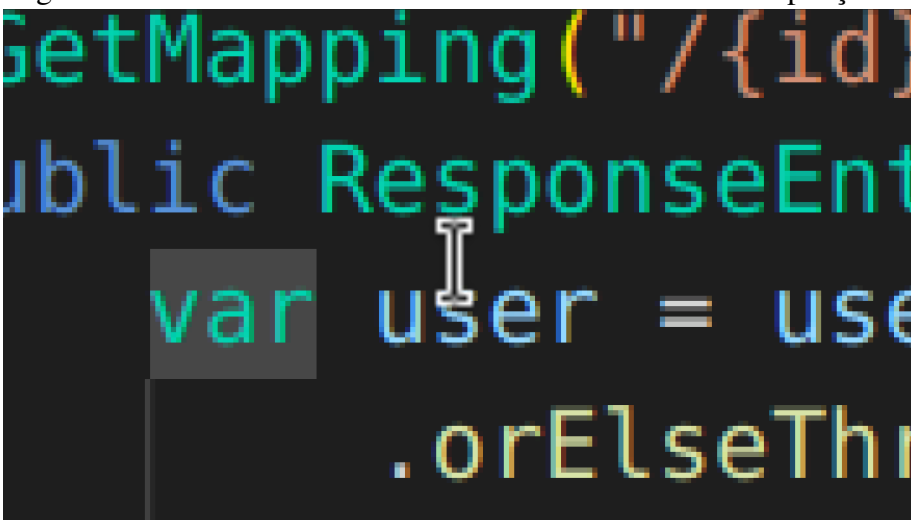
Nesta seção são apresentados os resultados do experimento deste trabalho, onde os dados do resultado foram divididos em seções por SO, na qual a seção 5.1 apresenta os resultados obtidos utilizando o SO Ubuntu, e na seção 5.2 são os resultados obtidos utilizando o SO *Windows*.

5.1 Resultados do Experimento no Ubuntu

O SO Ubuntu 20.04 foi utilizado no experimento para uma comparação com o experimento feito no SO *Windows*, possibilitando um ponto de vista diferente em cada SO do usuário.

5.1.1 Ferramenta de ampliação no VSCode

Figura 10 – Visão do VSCode utilizando a ferramenta de ampliação



Fonte: Elaborada pelo Autor.

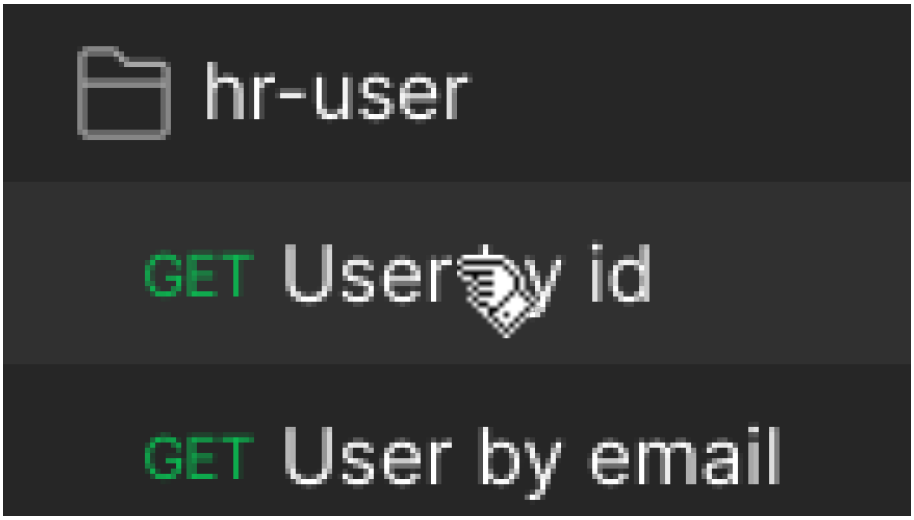
Na utilização do VSCode foram obtidos bons resultados, o *zoom* se comportou sem *bugs* e sempre fixo no ponto onde era deixado, porém este comportamento cria a necessidade do usuário ficar movimentando o cursor do *mouse* para conseguir acompanhar o que estava digitando no editor, o que não é algo incômodo devido ao fato de ser possível movimentar o cursor através do *touch pad* do *laptop*, e o mesmo fica bem próximo das mãos quando o usuário está digitando.

Entretanto, será visto que, diferente do *Windows*, no Ubuntu não foi encontrado

opções de configurações para o usuário escolher se quer que o foco do *zoom* acompanhe o cursor do *mouse*, foco do teclado, etc. Tais opções poderiam dar uma melhor experiência de usuário para o desenvolvedor com BV, onde a opção do foco no cursor do texto poderia ajudar na hora de digitar os códigos e não haver mais a necessidade de mover o cursor do *mouse* manualmente sempre para corrigir o foco do *zoom*.

5.1.2 Ferramenta de ampliação no Postman

Figura 11 – Visão das pastas do Postman utilizando a ferramenta de ampliação.



Fonte: Elaborada pelo Autor.

No *Postman* o resultado foi bem parecido com o do VSCode, o foco do *zoom* ficou fixo ao clicar com o *mouse* nos elementos da aplicação e ao navegar nos elementos com a tecla "tab". Um ótimo comportamento sem *bugss*, como, por exemplo, a mudança de posicionamento do foco do *zoom*.

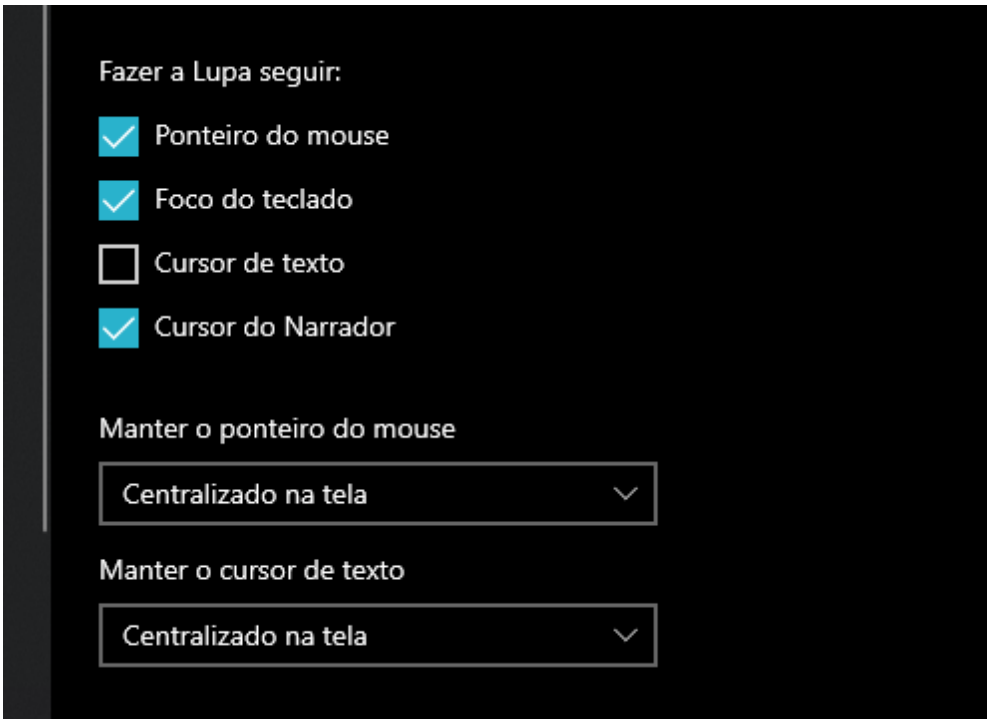
Foi criado uma *collection*, criados requisições e variáveis de ambiente, logo em seguida foram feitos testes de requisição à uma API simples que foi criada utilizando o VSCode e foi verificado que o comportamento do *zoom* e do *Postman* seguiu de forma que não impactou de forma negativa na experiência do usuário, demonstrando que um desenvolvedor com BV conseguiria utilizar esta ferramenta com o *zoom* sem problemas.

5.2 Resultados do Experimento no Windows

Nesta seção são apresentados os resultados do experimento da utilização da ferramenta de ampliação do *Windows*, ao utilizar as ferramentas para desenvolvimento de *softwares*, o VSCode e o *Postman*.

Um detalhe importante é que a ferramenta de ampliação do *Windows* possui algumas configurações diferentes, alterando alguns comportamentos do *zoom* ao ser utilizado. Na Figura 12 é possível observar uma das várias configurações possíveis que alteram o comportamento do *zoom*.

Figura 12 – Opções de comportamento do cursor nas configurações do zoom.



Fonte: Elaborada pelo Autor.

5.2.1 Ferramenta de ampliação no VSCode

Como visto na Figura 12, é possível modificar o comportamento da lupa ao utilizar o *zoom*, porém uma dessas opções é inviável de desativar, a opção "Ponteiro do *mouse*" no menu "Fazer a lupa seguir", esta opção é a que faz o *zoom* acompanhar o cursor do *mouse* ao mover o *mouse*, com essa opção desativada o quandro do foco do *zoom* ficará fixo mesmo o usuário movendo o *mouse*.

Ao testar diferentes combinações, foi visto que o usuário pode ter uma ótima experi-

ência baseado na combinação escolhida. Por exemplo, com a opção “Cursor do texto” é possível forçar o *zoom* a ficar sempre no centro do cursor do texto, o cursor do texto é a barra vertical que representa o local onde o próximo caractere será impresso na tela, facilitando ao usuário ao digitar não precisar corrigir o foco do *zoom* com o *mouse*.

A opção “foco do teclado” faz com que o foco do *zoom* acompanhe o foco do teclado. Por exemplo, o foco do *zoom* acompanha o foco do teclado ao teclar a tecla “tab”, ou seguir o foco quando o usuário altera uma janela com o atalho de teclado “alt + tab”.

Notou-se que, ao desativar todas as opções, exceto a opção “Ponteiro do *mouse*”, o *zoom* do *Windows* irá se comportar de forma idêntica ao do Ubuntu.

5.2.2 Ferramenta de ampliação no Postman

O *Postman* foi utilizado de forma semelhante no Ubuntu e *Windows*. Foi criado uma *collection*, algumas requisições e variáveis de ambiente, onde obtivemos resultados diferentes para cada opção das configurações do menu “Fazer a lupa seguir”.

5.2.3 Resultados do comportamento da zoom no VSCode e Postman

Cada posição do Quadro 2 marcada com um “x” representa a opção ativa no momento do experimento.

Quadro 2 – Resultado das diferentes configurações do zoom no VSCode e Postman

RESULTADOS	Ponteiro do *mouse*	Foco do teclado	Cursos de texto	Cursos do Narrador
Resultado A	x			
Resultado B	x	x		
Resultado C	x	x	x	
Resultado D	x	x	x	x
Resultado E	x	x		x
Resultado F	x		x	
Resultado G	x		x	x
Resultado H	x			x

Fonte: Elaborado pelo Autor.

- **Resultado A**: O *zoom* se comportou ficando fixo, independente do teclado, apenas mudando a sua posição com o movimento do *mouse*, comportamento semelhante ao do *zoom* do Ubuntu.
- **Resultado B**: O *zoom* seguiu o foco do teclado, focando nas abas abertas ao digitar o comando “alt + tab”, ou ao navegar nos ícones com a tecla “tab”, porém esse comportamento no Postman não foi satisfatório por manter o foco em uma posição não correta com relação

ao ícone.

- **Resultado C**: Nesta configuração, foi observado também que o foco do *zoom* busca sempre manter o cursor do texto dentro do quadro visual que o *zoom* cria ao ser ativado, porém não busca manter o cursor do texto centralizado, fato esse que poderia ou não melhorar a experiência do usuário.
- **Resultado D**: Com esta configuração não foi observado nenhuma diferença comparado com a configuração do Resultado C. Suponhamos que a opção "Cursor do Narrador" somente fará alguma diferença caso o leitor de tela esteja ativado.
- **Resultado E**: Como descrito no Resultado D, ativar a última opção não fará diferença, logo esse resultado foi semelhante ao Resultado B.
- **Resultado F**: Nesta configuração o *zoom* não irá mais focar nos ícones onde o teclado estiver em foco, porém irá continuar acompanhando o cursor de texto.
- **Resultado G**: Resultados idênticos ao Resultado F.
- **Resultado H**: Resultados idênticos ao Resultado A.

5.2.4 Discussão e análise dos resultados

De forma geral, esta avaliação nos mostrou que a acessibilidade com o *zoom* em ambos os SOs não é perfeita mas proporciona uma boa experiência de usuário e possibilita que os DDVs trabalhem com desenvolvimento de *software*. No Ubuntu foi observado que o *zoom* possui menos diferentes configurações, o que pode não satisfazer alguns usuários, porém a sua maior simplicidade e a possibilidade de alterar o atalho de ativar/desativar o *zoom* é realmente algo bem vantajoso. Entretanto, as muitas configurações diferentes do *zoom* no *Windows* pode agradar mais usuários pois, mesmo não podendo alterar a tecla de atalho para ativar/desativar o *zoom*, ainda assim possui recursos que auxiliam mais na hora da digitação.

Estes resultados foram obtidos testando o *zoom* de forma prática como ferramente auxiliar para conseguir utilizar as ferramentas de desenvolvimento de *software*. Foi testado a navegação nas ferramentas com funcionalidades nativas do teclado com a tecla "tab" e atalhos de teclado das próprias ferramentas para avaliar o comportamento do *zoom* em cada configuração, como visto anteriormente.

5.3 Resultados da avaliação

Após a avaliação foram obtidos alguns resultados diferentes para cada SO, bem como para cada configuração do *zoom* no *Windows*.

Iniciando com os resultados da avaliação no SO Ubuntu, observou-se que, independente de qual ferramenta estivesse sendo utilizada (VSCode ou *Postman*), o *zoom* permaneceu imóvel sempre que o cursor do *mouse* esteve imóvel independente de qual ação era executada, como:

- No VSCode:
  - Ao pesquisar por outros arquivos o projeto da API
  - Ao digitar o código no editor.
  - Ao instalar novas extensões.
  - Ao utilizar o terminal integrado.
  - Ao criar novas pastas ou arquivos.
  - Ao navegar no projeto com a tecla “tab”.
- No *Postman*:
  - Ao criar novas *collections*.
  - Ao criar novas requisições.
  - Ao configurar novas requisições.
  - Ao criar novas variáveis de ambiente.
  - Ao enviar as requisições e receber a resposta.
  - Ao navegar no projeto com a tecla “tab”.

Concluindo que, apesar do *zoom* no Ubuntu não ter configurações para diferentes comportamentos, ele é uma ótima opção para quem não tem problemas em utilizar o *mouse* para corrigir a sua posição na tela. Sabendo disse, o mais indicado seria utilizar o *zoom* do Ubuntu em *laptops* já que é possível mover o cursor do *mouse* com o *touchpad*, facilitando ao corrigir o foco do *zoom* quando necessário e continuar com as mãos no teclado.

De outra forma, foi observado pontos diferentes na avaliação do *zoom* do *Windows* devido as suas diversas configurações possíveis. No *Windows*, como pôde ser visto na subseção 5.2.3, existem várias configurações diferentes e podemos fazer combinações com elas para avaliar o comportamento do *zoom* em diferentes ações nas ferramentas.

Como pode ser visto no Quadro 2, foram obtidos oito resultados, porém alguns são foram idênticos à outros. Iniciando com o **Resultado A** temos a configuração mais simples do

zoom, que possui o comportamento idêntico ao do *zoom* do Ubuntu. Com esta configuração não foram detectados problemas no comportamento do *zoom*, o seu foco permaneceu sempre onde o cursor do *mouse* estava ao pesquisar arquivos, digitar código, instalar novas extensões e utilizar o terminal integrado. Nesta configuração a única diferença do *zoom* do *Windows* para o do Ubuntu é que no Ubuntu é possível alterar o atalho de teclado para ativar/desativar o *zoom*.

No **Resultado B** é adicionado mais uma opção de comportamento do *zoom* ficando ativas as opções "Ponteiro do *mouse*" e "Foco de teclado", com essa configuração o *zoom* agora segue o foco do teclado, ou seja, o foco segue ao usar a tecla "tab" ou "Alt + tab". Foi identificado um problema ao navegar com o "tab" no *Postman*, o foco do *zoom* não ficava posicionado nos botões corretamente fixando o foco em um local da ferramenta. Concluiu-se que no *Postman* é inviável navegar com o "tab" quando estiver com o *zoom* ativo, algo que deve ser melhorado por quem desenvolve e mantém o *Postman*. Já no VSCode, o comportamento foi como o esperado, com o foco seguindo o foco do teclado navegando com o "tab".

No **Resultado C**, além das opções do **Resultado B**, temos também a opção "Cursor do texto". Com esta opção o foco do *zoom* segue o ponteiro de digitação de texto, removendo a necessidade de corrigir a posição do foco do *zoom* movendo o *mouse*. Esta é uma boa opção para o desenvolvedor quando está digitando o código no VSCode, mas é importante lembrar que o foco não fica sempre centralizado na tela, o foco se move no decorrer que o cursor de texto chega ao fim da área em foco, porém o mais ideal seria se o foco do *zoom* ficasse sempre centralizado com o cursor de texto, isso é um ponto que melhoraria a usabilidade.

Por fim, a avaliação concluiu que, dependendo da ação, algumas configurações podem melhorar a usabilidade e outras podem piorar, logo as ferramentas de ampliação avaliadas ainda precisam melhorar em alguns aspectos, mas esses pontos negativos não impedem que os DDVs com BV trabalhem com desenvolvimento de *software*. Os resultados não abordados são os resultados idênticos aos já apresentados, conferir no Quadro 2.

6 CONCLUSÕES E TRABALHOS FUTUROS

A deficiência é algo que atinge milhões de pessoas e encontrar profissionais que se sintam incluídos sempre foi algo raro, porém a acessibilidade veio se demostrando estar sempre em constante evolução junto com a TIC, incluindo cada vez mais os usuários no mercado de trabalho, inclusive na área de desenvolvimento de *software*.

Neste trabalho foi apresentado a experiência de um usuário com BV sobre como é atuar com algumas ferramentas de desenvolvimento de *software* e o quão acessível estas ferramentas são para serem usadas com o *zoom* nativo dos SOs *Windows* e Ubuntu. Foi identificado que cada SO possui a sua peculiaridade em seu *zoom*, onde cada *zoom* possui opções diferentes de configuração e um comportamento um pouco diferente dependendo da configuração e da ferramenta em que o usuário está navegando.

Após a avaliação foi observado que um desenvolvedor com BV irá conseguir utilizar as ferramentas utilizadas para desenvolvimento de *software* com o auxílio do *zoom*. O usuário deve apenas identificar qual a configuração que mais lhe atende, podendo ser a mais simples, que o *zoom* fica fixo e apenas se movimenta com o movimento do cursor do *mouse*, ou uma configuração com mais recursos, onde o *zoom* se movimenta também baseado no cursor de texto ou tecla “tab”. Independente do SO utilizado, o usuário com BV irá conseguir utilizar com maestria com configuração simples as ferramentas avaliadas, porém irá encontrar alguns *bugs* caso ative as outras funcionalidades de navegação do *zoom*.

De forma complementar, observou-se que o comportamento do *zoom* no *Windows* ainda precisa melhorar quando algumas opções de navegação estão ativadas, notou-se que no *Postman* o *bug* da navegação com tecla “tab” é o pior, pois fica inviável navegar nos ícones apenas com o “tab”, o foco do *zoom* fica travado em um ponto e não segue o cursor de navegação.

Entretanto, notou-se que utilizando o *Windows* o usuário poderá ativar uma das suas poções de navegação do *zoom* que acompanha o cursor de digitação de texto, o que torna a experiência muito agradável. Por outro lado, no Ubuntu existe a opção de modificar as teclas de atalho para ativar e desativar o *zoom*, o que pode melhorar na experiência do usuário, em vista que o usuário pode escolher teclas que facilitem ativar e desativar o *zoom* com uma única mão. Por fim, observou-se uma pequena diferença no tempo de ativação e desativação do *zoom*, onde no Ubuntu é praticamente instantâneo, no *Windows* existe um breve período de tempo de espera podendo chegar a mais ou menos 1 segundo.

Como trabalho futuro, pretende-se fazer avaliações com as mesmas ferramentas

de desenvolvimento de *software* mas utilizando os leitores de tela do *Windows* e Ubuntu para identificar o nível de acessibilidade das ferramentas para os DDVs que são completamente cegos.

REFERÊNCIAS

ALVES, G. **Tecnologia Assistiva: ampliando habilidades e diminuindo limitações**. Porto Alegre - RS, 2016. Monografia (Graduação em Psicologia) - Universidade Federal do Rio Grande do Sul.

AMORIM, J. F. G. de; RAFANTE, H. C.; CAIADO, K. R. M. A organização política das pessoas com deficiência no brasil e suas reivindicações no campo educacional. **Revista Educação Especial**, Universidade Federal de Santa Maria, Santa Maria - RS, v. 32, p. 1–26, 2019.

BAGATINI, C. **Dispositivo que auxilia pessoas com deficiência visual a utilizar o transporte público**. Porto Alegre - RS, 2019.

BERSCH, R. **Introdução à tecnologia assistiva**. Porto Alegre: CEDI, v. 21, 2008.

CAMPOS, M. de B.; SÁNCHEZ, J.; SOUZA, T. C. de. Acessibilidade na web no brasil: percepções dos usuários com deficiência visual e desenvolvedores web. **Proceedings of the Nuevas Ideas en Informática Educativa TISE**, [S.l: s.n], p. 325–333, 2013.

CASTRO, S. S. d.; CÉSAR, C. L. G.; CARANDINA, L.; BARROS, M. B. A.; ALVES, M. C. G. P.; GOLDBAUM, M. Deficiência visual, auditiva e física: prevalência e fatores associados em estudo de base populacional. **Cadernos de Saúde Pública**, SciELO Public Health, [S.l: s.n], v. 24, n. 8, p. 1773–1782, 2008.

CESARO, S.; MUSSI, A.; SILVA, L.; SILVA, F.; CARVALHO, J.; SANTOS, M.; NAYAK, R.; NARAYANAN, A.; TRIPATHY, A. Eyespace: projeto colaborativo de um óculos para pessoas com deficiência visual no cenário internet of things. **SIMPÓSIO BRASILEIRO DE TECNOLOGIA DA INFORMAÇÃO E COMUNICAÇÃO NA CONSTRUÇÃO**, [S.l: s.n], v. 2, p. 1–5, 2019.

COSTA, A. C. C. dos S.; FONSECA, L. C. C.; ROSA, M. d. F. L. Acessibilidade urbana facilitada por aplicativos para pcds: uma revisão sistemática e literatura. **Research, Society and Development**, [S.l: s.n], v. 11, n. 11, p. e08111133296–e08111133296, 2022.

DIAS, A. L.; FORTES, R. P. de M.; MASIERO, P. C.; GOULARTE, R. Uma revisão sistemática sobre a inserção de acessibilidade nas fases de desenvolvimento da engenharia de software em sistemas web. **In IHC**, São Carlos - SP, p. 39–48, 2010.

FÁTIMA, E. A.; SANTINELLO, J.; MAMCASZ, L. V. V.; CARVALHO, R. S. D. S. de. O uso do dosvox no ensino de ciências por professores em formação inicial. **Tecné, Episteme y Didaxis: TED**, [S.l: s.n], p. 979–987, 2021.

FERREIRA JÚNIOR, J. H. F. J.; PEREIRA, A. A. de S. Análise e soluções de acessibilidade web para deficientes visuais utilizando os padrões w3c. **Revista Científica UNIFAGOC-Multidisciplinar**, [S.l: s.n], v. 4, n. 2, 2020.

FRIZZERA, A.; SONDERMANN, D.; SIMÕES, A.; KOEHLER, A. O leitor de tela e a criação de materiais digitais acessíveis a pessoas com deficiência visual. In: **Incluir é possível: desmistificando barreiras no processo de ensino-aprendizagem.**, [S.l: s.n], 2019. Disponível em: http://biblioteca.incaper.es.gov.br/digital/handle/123456789/3508. Acesso em 01 jun. 2023.

GARDIN, S. **Tecnologias assistivas no ensino superior: um estudo de caso na UFSM**. Universidade Federal de Santa Maria, Santa Maria - RS, 2014.

HERSH, M. A.; JOHNSON, M. A. **Assistive technology for visually impaired and blind people**. [*S. l.*]: Springer, 2008. v. 1.

HUSSEY, S.; COOK, A. Assistive technologies: Principles and practice.. st. louis, mo: Mosby. **Year Book, Inc**, p. 22, 1995.

IBGE, I. B. d. G. e. E. **Pessoas com deficiência**. 2010. Disponível em: https://educa.ibge.gov.br/jovens/conheca-o-brasil/populacao/20551-pessoas-com-deficiencia.html. Acesso em: 20 de mar. 2023.

INÁCIO, C. D. da S.; RAPHAEL, G. P. Bengala eletrônica para deficientes visuais. **Revista Alomorfia**, [S.l: s.n], v. 5, n. 1, p. 220–234, 2021.

JAWS, J. A. W. S. **is the world's most popular screen reader, developed for computer users whose vision loss prevents them from seeing screen content or navigating with a mouse**. 2023. Disponível em: https://www.freedomscientific.com/products/software/jaws/. Acesso em: 20 mar. 2023.

LIMA, G.; DANTAS, H.; YOSHIKAWA, G.; NAKAYAMA, L.; SOUSA, L. **Óculos inteligente para deficientes visuais com visão computacional embarcada para detecção de objetos**. São Paulo - SP, 2022.

MEDEIROS, A. R. R. **Noção corporal, lateralidade e estruturação espaço-Temporal na deficiência visual: estudo comparativo entre praticantes e não praticantes de Goalball**. [S.l: s.n], 2016.

MELO, M. E. S. R. **Acessibilidade para deficientes visuais: uma abordagem prática da lei brasileira de inclusão**. Florianópolis - SC, 2021.

MEMON, J.; SAMI, M.; KHAN, R. A.; UDDIN, M. Handwritten optical character recognition (ocr): A comprehensive systematic literature review (slr). **IEEE Access**, IEEE, [S.l: s.n], v. 8, p. 142642–142668, 2020.

MOURA, G. G. **Protótipo de automação residencial para tecnologia assistiva utilizando reconhecimento de voz e controle através de aplicativo móvel**. Universidade Federal de Santa Maria, Santa Maria - RS, 2022. Disponível em: http://repositorio.ufsm.br/handle/1/24002. Acesso em 01 jun. 2023.

NASCIMENTO, F. L. do; BARBOSA, P. L. S.; VIANA, W. Programando às cegas: investigando a acessibilidade de ambientes de desenvolvimento de software. In: SBC. **Anais do VII Workshop sobre Aspectos Sociais, Humanos e Econômicos de Software**. [*S. l.*], 2022. p. 1–10.

NASCIMENTO, R. A. L. d. *et al.* **O Impacto dos recursos de tecnologia assistiva na educação e inclusão da pessoa com deficiência visual**. Universidade Federal da Grande Dourados, Dourados - MS, 2015.

NVDA, N. D. A. **We believe that every Blind + Vision Impaired person Deserves the right to freely easily access a computer!** 2023. Disponível em: https://www.nvaccess.org/. Acesso em: 20 mar. 2023.

OMS, O. M. d. S. **Relatório Mundial Sobre a Deficiência**. 2012. Disponível em: https://apps.who.int/iris/bitstream/handle/10665/44575/9788564047020_por.pdf. Acesso em: 20 mar. 2023.

OMS, O. M. d. S. **Relatório global sobre cegueira**. 2019. Disponível em: https://news.un.org/pt/story/2019/10/1690122. Acesso em: 20 mar. 2023.

PINTO, K. C. B.; SILVA, R. P. d.; TEIXEIRA, F. G. Requisitos de projeto de interfaces gráficas de objetos de aprendizagem acessíveis para usuários com baixa visão. **Educação gráfica. v. 23, n. 1 (abr. 2019), p. 96-115**, [S.l: s.n], 2019. Disponível em: http://hdl.handle.net/10183/203797. Acesso em 05 jun. 2023.

POSTMAN. **Build APIs together**. 2022. Disponível em: https://www.postman.com/. Acesso em: 20 mar. 2023.

PREECE, J. Design de interação: além da interação homem-computador/trad. **Viviane Possamai. Porto Alegre - RS**, 2005.

RODRÍGUEZ, L.; ANTEPARA, J.; BRAGANZA, L. Web accessibility analysis of the universities and public polytechnic schools of guayaquil applying the nte inen iso/iec 40500: 2012 standard. **Espirales Revista Multidisciplinaria de investigación**, [S.l: s.n], v. 3, n. 27, p. 59–77, 2019.

SALES, G. M.; SOUSA, L. P. A. de; MACEDO, M. S. Tecnologia inclusiva em biblioteca universitária: uma proposta de uso. In: **Anais do 28° Congresso Brasileiro de Biblioteconomia, Documentação e Ciência da Informação-FEBAB**. [*S. l.*: *s. n.*], 2019. v. 28.

SANTO, C. U. N. D. E.; OLIVEIRA, A. A. D. **Um olhar sobre o ensino de ciências e biologia para alunos deficientes visuais**. São Mateus - ES, 2018.

SANTOS, A. **Os benefícios de utilizar a tecnologia assistiva/acessibilidade na inclusão de deficientes na área da tecnologia**. [S.l: s.n], 2021.

SCHAFHAUZER, L. M. B.; SILVA, C. M. da. Análise da acessibilidade dos serviços de tecnologia da informação e comunicação do tribunal de justiça de pernambuco (tjpe) para servidores com deficiência visual. **Research, Society and Development**, [S.l: s.n], v. 11, n. 3, p. e43711326859–e43711326859, 2022.

SEBOLD, W.; PEDROSA, S. M. P. d. A. Tecnologia assistiva: uma introdução. **Revista Educação e Cultura Contemporânea**, [S.l: s.n], v. 17, n. 51, p. 111–134, 2020.

SILVEIRA, C. da; BEILER, A. **Análise comparativa dos softwares leitores de tela utilizando o ambiente virtual de aprendizagem Moodle**. Porto Alegre - RS, 2012.

SOUSA, R. P. d. **A importância da tecnologia assistiva para inclusão de alunos com deficiência visual no ensino fundamental brasileiro**. São Carlos - SP, 2018. Monogragia (Especialização em Educação e Tecnologias: Produção e Uso de Tecnologias para Educação da Universidade Federal de São Carlos) - Universidade Federal de São Carlos. Disponível em: https://edutec.ead.ufscar.br/tccs/b17add24fd3e507b1af350f9f377ded8.pdf. Acesso em 01 jun. 2023.

SUSANTO, S.; NANDA, D. S. Teaching and learning english for visually impaired students: an ethnographic case study. **English Review: Journal of English Education**, [S.l: s.n], v. 7, n. 1, p. 83–92, 2018.

VARGHESE, S. T.; RATHNASABAPATHY, M. Assistive technology for low or no vision. In: SPRINGER. **Information Management and Machine Intelligence: Proceedings of ICIMMI 2019**. [*S. l.*], 2021. p. 199–202.

VSCODE, V. S. C. **Code editing. Redefined.** 2022. Disponível em: https://code.visualstudio.com/. Acesso em: 20 mar. 2023.

WOLF, C. M.; TAGLIETTI, M. Exercícios oculares na insuficiência de convergência: Série de casos. **Revista Sociedade Portuguesa de Oftalmologia**, [S.l: s.n], v. 43, n. 1, 2019.