UNIVERSIDADE DE CAXIAS DO SUL
ÁREA DO CONHECIMENTO DE CIÊNCIAS EXATAS E ENGENHARIAS

GABRIEL GALLINA MOSCONE

NOVA INTERFACE RESPONSIVA PARA O SISTEMA EDUCACIONAL WEBALGO: REDESIGN FOCADO EM USABILIDADE E ACESSIBILIDADE

BENTO GONÇALVES

2025

GABRIEL GALLINA MOSCONE

NOVA INTERFACE RESPONSIVA PARA O SISTEMA EDUCACIONAL WEBALGO: REDESIGN FOCADO EM USABILIDADE E ACESSIBILIDADE

Trabalho de Conclusão de Curso apresentado como requisito parcial à obtenção do título de Bacharel em Ciência da Computação na Área do Conhecimento de Ciências Exatas e Engenharias da Universidade de Caxias do Sul.

Orientador: Prof. Dr. Leonardo Pellizzoni

BENTO GONÇALVES

2025

GABRIEL GALLINA MOSCONE

NOVA INTERFACE RESPONSIVA PARA O SISTEMA EDUCACIONAL WEBALGO: REDESIGN FOCADO EM USABILIDADE E ACESSIBILIDADE

Trabalho de Conclusão de Curso apresentado como requisito parcial à obtenção do título de Bacharel em Ciência da Computação na Área do Conhecimento de Ciências Exatas e Engenharias da Universidade de Caxias do Sul.

Aprovado em 25/11/2025

BANCA EXAMINADORA

_____

Prof. Dr. Leonardo Pellizzoni
Universidade de Caxias do Sul - UCS

_____

Prof. Dra. Elisa Boff
Universidade de Caxias do Sul - UCS

_____

Prof. Me. Alexandre Erasmo Krohn Nascimento
Universidade de Caxias do Sul - UCS

"O sonho é que leva a gente para frente. Se a gente for seguir a razão, fica aquietado, acomodado."

Ariano Suassuna

RESUMO

No aprendizado da programação é essencial que sejam realizadas atividades práticas para reforçar a teoria. A Universidade de Caxias do Sul (UCS) utiliza um interpretador da linguagem C desenvolvido por professores e chamado de WebAlgo. O software conta com uma lista predefinida de exercícios que auxilia na aprendizagem dos alunos e permite que os professores acompanhem o andamento dos estudantes. Inicialmente, o WebAlgo foi desenvolvido na versão Web, porém, com a descontinuação do suporte ao Java nos navegadores, passou a ser Desktop, utilizando listas duplamentes encadeadas e posteriormente convertido para Código de Três Endereços (C3E) em uma aplicação *Web*, voltada para o layout do *computador*. Pesquisas, como o Pesquisa Nacional por Amostra de Domicílios (PNAD), indicam que os usuários utilizam prioriatariamente o celular para acessar a internet, com uma taxa crescente, e poucos possuem acesso a um computador, limitando a utilização do sistema a uma parcela de pessoas. Tendo em vista esse cenário, o presente trabalho tem como proposta ampliar o acesso ao WebAlgo *Web* aos tablets e celulares e aumentar a acessibilidade. Para isso, será utilizado no trabalho técnicas e frameworks, garantindo uma componentização, responsividade e implementar funcionalidades de responsabilidade, como ajuste de tema e tamanho da fonte. Todos os códigos desenvolvidos estarão disponíveis em um repositório público do GitHub.

Palavras-chave: Responsividade. Acessibilidade. Ensino. Bootstrap. Programação.

ABSTRACT

In programming education, it is essential to carry out practical activities to reinforce theoretical knowledge. The University of Caxias do Sul (UCS) uses an interpreter for the C language developed by professors, called WebAlgo. The software includes a predefined list of exercises that support students' learning and allow professors to monitor their progress. Initially, WebAlgo was developed as a web-based application; however, with the discontinuation of Java support in browsers, it was converted into a desktop version using doubly linked lists, and later migrated to a C3E web application designed primarily for desktop layouts. Research, such as the PNAD, indicates that users primarily access the internet through mobile phones, with an increasing rate, while few have access to a computer, thus limiting the system's use to a smaller portion of people. Considering this scenario, the present work aims to expand access to the WebAlgo web platform for tablets and mobile devices, enhancing accessibility. To achieve this, techniques and frameworks will be employed to ensure componentization, responsiveness, and the implementation of accessibility features such as theme adjustment and font size customization. All the developed code will be available in a public GitHub repository.

Keywords: Responsiveness. Accessibility. Teaching. Bootstrap. Programming.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ALGORITMOS

LISTA DE ABREVIATURAS E SIGLAS

ABERGO Associação Brasileira de Ergonomia
ABNT Associação Brasileira de Normas Técnicas
C3E Código de Três Endereços
CSS *Cascading Style Sheets*
DOM *Document Object Model*
GDB *GNU Debugger*
HTML *Hypertext Markup Language*
HTTP *Hypertext Transfer Protocol*
IA Inteligência Artificial
IDE *Integrated Development Environment*
ISO *International Organization for Standardization*
MVC *Model-View-Controller*
PNAD Pesquisa Nacional por Amostra de Domicílios
POC Prova de Conceito
PSSUQ *Post Study System Usability Questionnaire*
px *pixels*
SEO Otimização para Mecanismos de Busca
SOA *Service-Oriented Architecture*
TI Tecnologia da Informação
UCS Universidade de Caxias do Sul
UI *User Interface*
W3C *World Wide Web Consortium*
WAI Iniciativa de Acessibilidade na Web
XML *Extensible Markup Language*

SUMÁRIO

1 INTRODUÇÃO

Professores dos cursos de Tecnologia da Informação (TI) relatam sobre a dificuldade que os alunos têm em seu primeiro contato com a programação, necessitando desenvolver uma lógica e aprender a sintaxe das linguagens (JENKINS, 2002). Com a finalidade de mitigar essas barreiras, os trabalhos recomendam que os ingressantes realizem atividades práticas para desenvolver as habilidades na programação, através de softwares e exercícios reais (JENKINS, 2002). Em conformidade com os pontos abordados desenvolveu-se ferramentas voltadas para a educação, como linguagens de programação baseadas em ícones, micromundos de aprendizagem e Sistemas de Tutores Inteligentes (GOMES; HENRIQUES; MENDES, 2008).

No contexto da Universidade de Caxias do Sul (UCS), estas ferramentas também foram desenvolvidas, dando origem ao WebAlgo - originalmente web e depois Desktop - ou seja, que necessita o acesso a um computador - voltado para o aprendizado da programação, contendo um banco de problemas que auxiliam os alunos a desenvolver os primeiros passos na programação (DORNELES; JR; ADAMI, 2010). Um software, utilizado na UCS desde 2009, desenvolvido em Java que valida automaticamente as soluções submetidas para cada problema, informando o grau de assertividade do algoritmo (DORNELES; JR; ADAMI, 2010). Vale ressaltar, que a primeira versão do WebAlgo foi *Web*, porém, com a descontinuação do suporte ao Java nos navegadores, a versão utilizada passou a ser Desktop (DORNELES; JR; ADAMI, 2010).

Ainda que o WebAlgo, versão Desktop, esteja em uso, visando auxiliar o aprendizado de programação, pesquisas evidenciam que o celular ainda é o principal meio de acesso a internet para os estudantes, sendo utilizado por 97,3% deles - desses, menos de 30% acaba tendo a possibilidade de acesso ao computador (ESTATíSTICA, 2023). Dificultando que os usuários utilizem os sites que não funcionam corretamente em dispositivos móveis e também aplicativos exclusivos de computadores, como o WebAlgo, que requer o Java para funcionar.

Com base nestes pontos, o trabalho de Susin, 2024, desenvolveu uma versão desse aplicativo totalmente Web, com um compilador de C3E, possibilitando que os estudantes conseguissem acessar e usufruir dessa ferramenta na maior parte dos dispositivos que tenha acesso a internet (SUSIN, 2024). Assim, essa ferramenta se torna mais acessível, facilitando o aprendizado dos ingressos nos cursos de Tecnologia da Informação. Destacando-se que evita a necessidade de realizar configurações no computador, como instalar o Java e principalmente, possuir um computador fora do ambiente universitário.

No entanto, vale ressaltar que o trabalho desenvolvido por (SUSIN, 2024) possui algumas possibilidades de trabalhos futuros, sendo um deles o de possibilitar a usabilidade e disponibilidade em resoluções menores, como dispositivos móveis, ampliando a quantidade de usuários do WebAlgo.

Tendo em vista os fatos supracitados, a relevância das ferramentas de programação nas disciplinas e as possibilidades de acesso, o presente trabalho se propõe a permitir o uso da ferramenta WebAlgo, através de navegadores de dispositivos móveis, como celulares ou Notebooks, sem distorcer e preservando as funcionalidades e operações do software.

1.1 QUESTÃO DE PESQUISA

Quais características que o WebAlgo, quando acessadao por celulares, tenha a capacidade de se adaptar às telas menores, possibilitando o desenvolvimento e execução dos códigos fontes em linguagem C?

1.2 OBJETIVOS

Disponibilizar a utilização do WebAlgo em dispositivos móveis com acesso a *internet*, através do navegador, seja ele Android ou iOS.

O presente trabalho tem como objetivos específicos:

- Prototipar uma interface intuitiva e amigável
- Ampliar o acesso ao sistema web para funcionar em dispositivos móveis
- Testar o sistema em múltiplos dispositivos
- Realizar pesquisas de satisfação e usabilidade

1.3 ESTRUTURA DO TRABALHO

Inicialmente, apresenta-se uma introdução do trabalho desenvolvido, com um breve contexto do que será realizado e o escopo das atividades, dividido em problema de pesquisa, questão de pesquisa e objetivos.

O segundo capítulo, tem como objetivo fundamentar teoricamente os conceitos técnicos envolvidos nesse trabalho, explicando as linguagens, termos e métodos que serão aplicados ou analisados.

Por outro lado no capítulo 3, são elencados os trabalhos relacionados, buscando uma fundamentação em artigos semelhantes dos principais temas abordados no presente trabalho e as soluções utilizadas.

O capítulo 4 elenca a proposta de solução para o problema, explicando como será feito, o motivo das decisões e o que será utilizado, como tecnologias, protótipos e metodologia.

O capítulo 5 lista brevemente os resultados obtidos na primeira parte do trabalho, elencando o sistema desenvolvido e o resultados das pesquisas.

Por último, o capítulo 6, levanta as conclusões do trabalho, indicando se foi atendido a proposta ou não e organizando ideias de trabalhos futuros.

2 FUNDAMENTOS

O software é um sistema lógico constituído por instruções, que ao ser executadas, desempenham uma função desejada (PRESSMAN; MAXIM, 2021). Por isso, para se criar um software, recomenda-se seguir uma área, como a Engenharia de Software, que consiste em um processo que reúne ferramentas, práticas e processos - que se seguidos resultam em um software de qualidade no tempo estipulado (PRESSMAN; MAXIM, 2021).

Assim sendo, a engenharia tem como meta auxiliar todos os envolvidos no desenvolvimento do software, desde as etapas iniciais até disponibilizar o sistema em produção e os testes (SOMMERVILLE, 2011; PRESSMAN; MAXIM, 2021).

O código é um pilar fundamental e é essencial ter um controle do software desenvolvido, garantindo que se tenha um histórico das versões e que seja possível trabalhar em equipe (AQUILES; FERREIRA, 2014). Assim, surgiram as ferramentas de controle de versão, como o Git, que guarda e administra as revisões dos sistemas, tendo um histórico completo para cada repositório (COSENTINO; IZQUIERDO; CABOT, 2017). Com o tempo surgiram algumas aplicações *Web* que utilizam o Git, como o GitHub, que permite hospedar e acessar pelo celular ou computador os códigos do projeto, podendo salvar códigos de variadas *frameworks*, por exemplo, Node.js, jQuery e Sprint (AQUILES; FERREIRA, 2014).

Há casos no qual é necessário manter diferentes versões de um código, seja para ter ambientes diferentes, implementar recursos novos ou até mesmo testes, para isso o Git implementa as ramificações (*branch*) (BLISCHAK; DAVENPORT; WILSON, 2016). As *branchs*, mantém registros de onde você fez a cópia e armazena as novas mudanças, permitindo que diferente programadores trabalhem ao mesmo e que o código original permaneça intacto até o momento de mesclar eles (BLISCHAK; DAVENPORT; WILSON, 2016; VALE *et al.*, 2020).

Com essas ferramentas, há a possibilidade de diversos programadores trabalharem em correções ou funcionalidades novas simultaneamente, sendo necessário mesclar elas no final do desenvolvimento (VALE *et al.*, 2020). Essas unificações podem gerar conflitos que precisam ser resolvidos para ter um código final funcional, para realizar essa mescla há as *Pull-Requests* - que são solicitações para integrar o código a uma *branch*, permitindo realizar revisões (VALE *et al.*, 2020; GHIOTTO *et al.*, 2018). Nessa proposta, há metodologias de trabalho como a *trunk-based*, na qual se prioriza trabalhar apenas em uma *branch*, criando pequenas ramificações que constantemente devem voltar para a principal (RÍOS; EMBURY; ERASLAN, 2022).

2.1 ACESSIBILIDADE E RESPONSIVIDADE

A acessibilidade, segundo a Associação Brasileira de Normas Técnicas (ABNT), se caracteriza por garantir a possibilidade de um indíviduo usufruir de alguma ferramenta ou até

localidade com segurança e liberdade, mesmo que tenha uma limitação (MANZINI, 2005). Dessa forma, com o aumento da quantidade de informação presente no meio digital, essa preocupação se mostrou necessária nessa frente também, principalmente, para deficientes visuais e auditivos (SOUZA; TABOSA, 2014). Assim sendo, surgiu-se regulamentos, como a Iniciativa de Acessibilidade na Web (WAI), iniciativa da *World Wide Web Consortium* (W3C), que desenvolve os padrões e documentações para auxiliar os desenvolvedores, buscando integrar as pessoas com defiência - como, auditiva, motora, visual ou cognitiva - com a *internet* (CORRÊA; RIBEIRO, 2015).

A Organização Mundial da Saúde estima que aproximadamente 15% da população possui algum tipo de deficiência, mostrando a relevância do tópico e podendo ampliar uma parcela significativa a quantidade de possíveis usuários (ECONOMIC; AFFAIRS, 2024). Além disso, no Brasil há leis que regulamentam a acessibilidade, como a Lei nº 13.146/2015, que determina a obrigação de aplicar a acessibilidade nas aplicações *web* (DEFICIÊNCIA, 2015).

Portanto, para garantir essa característica, segundo (SOUZA; TABOSA, 2014), há alguns conceitos essenciais nos sistemas *web*, como, as interfaces, que são meios visuais no qual o sistema pode se comunicar com o usuário. Além da ergonomia, que segundo a Associação Brasileira de Ergonomia (ABERGO), a ergonomia cognitiva se refere a interpretação e raciocínio demonstrados conforme se utiliza um sistema. Também, há tecnologias assistivias, ou seja, ferramentas que auxiliam pessoas com limitações, como: leitores e ampliadores de tela, com o objetivo de adaptar uma interface para o usuário.

Ademais, além dos conceitos, ao configurar um sistema *web* é fundamental, criar os seguintes tópicos segundo a W3C (SOUZA; TABOSA, 2014; GITHUB, 2023):

- **Título:** identifica o sistema que está sendo acessado.
- **Texto alternativo nas imagens:** garante que um leitor que não consegue visualizar a imagem entenda o que está sendo mostrado, através de uma descrição.
- **Cabeçalhos:** ajuda a organizar a página em seções, permitindo uma navegação assertiva.
- **Controle de contraste:** controle do constraste da página, adequando a visão do leitor
- **Lente de aumento:** permite alterar o tamanho dos textos e imagens, conforme a necessidade
- **Atalhos:** permite realizar funções ou a leitura, sem o uso do mouse
- **Mídias alternativas:** acesso a página através de documentos auditivos ou até mesmo vídeos.
- **Tamanho adequado:** elementos com tamanho mínimo de 24*pixels* (px), garantindo que independente da resolução tenha um tamanho adequado para ser identificado e clicado.

- ***Labels***: elementos auxiliares associados corretamente, garantindo, que a navegação da página e leitores tenham um direcionamento

Outrossim, criou-se regulamentações e padrões de acessibilidade, como ISO ou até mesmo pela W3C, desenvolvendo ferramentas para melhorar a acessibilidade. Práticas como essas buscam garantir que os sistemas não sejam construídos com estruturas que deixem o usuário confuso ou usem demasiadamente de informações gráficas (CONFORTO; SANTAROSA, 2002).

Em paralelo com a busca à acessibilidade, há a responsividade, que consiste em um sistema se adaptar a resolução do dispositivo utilizado, permitindo que todas as funcionalidades sejam acessíveis (MONTEIRO *et al.*, 2020). O termo vem sendo priorizado desde 2010, quando Ethan Marcotte, abordou a necessidade dos sites se adaptarem de forma automática as telas diferentes (GERADE, 2024).

Portanto, é necessário que os desenvolvedores, criem os sistemas, com as ferramentas que as tecnologias possuem, como o *Hypertext Markup Language* (HTML), que suporta elementos semânticos (GERADE, 2024) e a utilização do *Cascading Style Sheets* (CSS), com a estilização completa das *interfaces*, com a criação de *layouts* fluídos ou até mesmo com *media querys*, propriedade do CSS que permitem definir estilos diferentes, de acordo com a resolução (GERADE, 2024).

Ademais, ao seguir essas regras, deixando os sistemas responsivos para diferentes resoluções, há pesquisas que apontam melhorias na usabilidade e aprendizado dos usuários, principalmente em dispositivos móveis. Fato que pode ser mais agravado em momentos de crise, como a pandemia de COVID, reforçando o uso de celulares (PARLAKKILIÇ, 2022).

Com o avanço de tecnologias assistivas, também criou-se ferramentas que auxiliam os desenvolvedores a identificarem os pontos de melhoria relacionados a acessibilidade, como avaliadores semi automatizados, mesmo que ainda seja de extrema importância realizar testes com usuários reais (ARA; SIK-LANYI; KELEMEN, 2024). Como por exemplo, o *PageSpeed Insights* e o *Lighthouse*, que auxiliam a medir a *performance* e a acessibilidade do produto, indicando pontos de melhorias e uma pontuação para comparação (PANDUWIKA; SOLEHATIN, 2024).

Para otimizar o desenvolvimento, há práticas como o *mobile first*, que iniciam a codificação pelo resolução do celular e posteriormente se adapta o *layout* para as telas maiores, como computadores e tablets. Essa prática, busca garantir a velocidade e acessibilidade para os celulares, tornando acessível as funcionalidades essenciais e garantindo a otimização do carregamento (WROBLEWSKI, 2012).

Somada a essa práticas, há *frameworks* de CSS, como o Bootstrap, que simplificam a criação de *layouts* dinâmicos, oferencendo soluções prontas através de elementos pré construidos. Além dessa padronização, também se deve utilizar os elementos corretos do HTML, como o *picture* e a minimização dos arquivos compilados, garantindo um tempo menor de carrega-

mento, principalmente, para os dispositios móveis e suas conexões mais limitadas, visto que, requerem pacotes de internet, velocidade e até mesmo os recursos limitados do celular (MOHD *et al.*, 2022).

Ademais, através da análise de sites universitários e governamentais na Arábia Saudita, identificou-se que mesmo com os inúmeros avanços das tecnologias e o aumento das normas e da digitalização dos serviços, há barreiras significativas que tornam difícil o acesso aos portais por pessoas com deficiências. Esta pesquisa destaca que esses portais não seguem adequadamente as diretrizes de acessibilidade, comprometendo a inclusão, refletindo que as próprias normas não refletem 100% as necessidades dos usuários. Por isso, revela-se a importância de pesquisas e testes com usuários reais, permitindo aprimorar os critérios e promover um ambiente inclusivo (AKRAM; SULAIMAN, 2017).

Nos ambientes de modelagem é comum ser necessário a interação com o *mouse* e desenvolveram uma ferramenta de modelagem acessível, podendo ser operado exclusivamente pelo teclado. Essas abordagens de navegação, atendem usuários com deficiência motora e visual, considerando a dificuldade em usar um *mouse* com precisão. Além disso, destaca-se que grande parte das ferramentas não possuem *interfaces* intuitivas e acessíveis, possuindo poucos ou nenhum atalho de teclado. Dessa forma, a pesquisa reforça a necessidade de seguir diretrizes e um mapeamento para aprimorar a inclusão digital, salientando que é necessário mapear a acessibilidade desde o início do projeto e não como uma tarefa final (SARIOĞLU; METIN; BORK, 2025; MASRUROH *et al.*, 2022).

Com a implementação da acessibilidade, outra maneira de validar o que foi implementado, para aplicativos móveis, é através das avaliações de usuários nas lojas, como a Google Play e a App Store. Para isso, é realizado uma análise, através do processamento de linguagem natural, mineração de texto e aprendizado de máquina, com o objetivo de classificar os sentimentos e detectar os problemas da *interface*, no entanto, geralmente focados na usabilidade e deixando de lado a acessibilidade, apesar de serem termos integrados (OLIVEIRA; ELER, 2025).

Dessa forma, novas áreas vão surgindo, como a engenharia da acessibilidade, que deve ser integrada durante todo o processo de desenvolvimento de um sistema, tendo, incluindo-a como um requisito não funcional. Para isso, torna-se necessário criar métricas e ter essa visão desde a criação da ideia do produto e crucialmente com a realização de avaliações com usuários reais com alguma deficiência. Esta abordagem, garante um desenvolvimento mais eficaz, com a identificação dos pontos de acessibilidade e evitando sistemas prontos, com diversos problemas, além do que, torna-se financeiramente mais econômico desenvolver ele desde o início de uma forma acessível (ARA; SIK-LANYI; KELEMEN, 2024).

2.2 PROTOTIPAGEM

A prototipagem é uma etapa do desenvolvimento utilizada para testar, planejar e validar as ideias de um sistema, garantindo que o processo de construção seja linear, sem ter retrabalhos e que tenha uma direção do que deve ser implementado (BAUMER *et al.*, 1996; BARBOSA *et al.*, 2024). Essa técnica, ajuda a reduzir os custos e os riscos de atrasos, podendo ter representações simples ou reais do produto (SZEKELY, 1995).

Os protótipos podem ser inicialmente simples esboços com papel e caneta, ilustrando as principais funcionalidades e a estrutura do sistema (SZEKELY, 1995). No entanto, há ferramentas mais completas para a ilustração, podendo ser baseadas em modelos de classes ou nas *interfaces* (BAUMER *et al.*, 1996).

Além disso, pode-se criar *wireframes* - que são esboços visuais da estrutura do sistema - sendo classificados em dois tipos de acordo com (LEITE *et al.*, 2024):

- **Baixa fidelidade:** foco na estrutura da interface, organização do conteúdo e dos fluxos, sem detalhes, geralmente em cores neutras
- **Alta fidelidade:** incluem detalhes visuais, sendo próximos do protótipo final, sendo recomendados para *feedback* dos clientes

Para criação de interface há ferramentas como o Figma, um editor gráfico, que permite a criação de *interfaces* e protótipos de aplicações.O Figma vem sendo amplamente utilizada pela possibilidade de trabalhar simultaneamente com outras pessoas, recursos em nuvem e praticidade, permitindo criar os *wireframes* e integrações entre os sistemas (LEITE *et al.*, 2024). O Figma unifica tarefas que geralmente eram encontradas em sistemas separados, com planos gratuitos, como prototipar, criar *wireframes* e criar ilustrações (STAIANO, 2022).

Algumas ferramentas possuem foco em geração de códigos, como é o caso do Figma que gera o código que deve ser usado, em CSS, para iOS e até para o Android (STAIANO, 2022). Ademais, o Figma se destaca por possibilitar a criação de componentes no *design*, criando uma padronização para a *User Interface* (UI), que se mantém consistente em todos usos (STAIANO, 2022).

2.3 TECNOLOGIAS PARA DESENVOLVIMENTO *WEB*

A estrutura básica de uma página *web* é o HTML, possuindo uma sintaxe que define os elementos das páginas, através dessa linguagem de marcação de hipertexto (CHALLAPALLI *et al.*, 2021). Com a estrutura definida, pode-se utilizar o CSS para criar o estilo da página, adicionando cores, espaçamentos e formatação, e por fim há o JavaScript, que adiciona funcionalidades ao *frontend* (CHALLAPALLI *et al.*, 2021). O *frontend* é a parte visual do sistema, que

os usuários irão visualizar e interagir (JALOLOV, 2024). O HTML está na versão 5, permitindo a configuração dos elementos *header*, *section*, *article* e o *footer*, otimizando a hierarquia da página e consequentemente a acessibilidade (CHALLAPALLI *et al.*, 2021).

Criado em 1995, o JavaScript é uma linguagem *web* de alto nível, utiliza para definir o comportamento das páginas, não tipada (CHALLAPALLI *et al.*, 2021; FLANAGAN, 2012). O JavaScript permite realizar modificações no *Document Object Model* (DOM) e configurar eventos na página, como ao usuário clicar em um botão ou ao descer a página (FLANAGAN, 2012). Dessa forma, com a evolução do Javascript, foram sendo introduzidas novas funções, permitindo realizar funções mais complexas, com a utilização de módulos e até mesmo *arrow functions* (ZAKAS, 2016). Essas melhorias, vão de acordo com a busca pela *performance* nas páginas, com práticas como o *lazy loading*, o *cache* e a minimização dos arquivos (ZAKAS, 2016).

2.3.1 Frameworks de responsividade

O CSS é o nível inicial para estilização de páginas *web*, garantindo que a página seja amigável e responsiva, que foi criado de acordo com a W3C, possuindo três versões e atualmente estando na terceira (CHALLAPALLI *et al.*, 2021). Essa versão, introduziu diversas melhorias para a responsividade e customização da página, permitindo criar animações, transições e o *layout grid*, segregados por resoluções. Portanto, torna-se possível, criar disposições que se adaptem as diferentes telas, mantendo interações únicas e funcionais para as funcionalidades (RAO; CHAURASIA; SINGH, 2023).

A funcionalidade de *media query* possibilita transformar uma página HTML em responsiva, permitindo customizar o *layout*, conforme a dimensão do dispositivo, ou seja, pode-se aplicar regras distintas para celulares, tablets e notebooks (FRAIN, 2022). Conforme pode ser observado na Figura 1, que aplica um estilo diferente para telas que tenham no mínimo 800 px. As *media queries* funcionam com regras variadas, como, altura, largura, orientação e até mesmo pela resolução (FRAIN, 2022).

Inicialmente, os *layouts* buscavam se adaptar para resoluções fixas, acompanhando as evoluções dos dispositivos. Porém, com o crescimento exponencial de novos aparelhos tornou-se inviável essa prática e abrindo espaço para a abordagem do *layout grid* ou *flex* (MARCOTTE, 2018). O *layout grid*, permite que a tela seja dividida em linhas e colunas, definindo a porção da tela ocupada e facilitando a divisão, independente do tamanho da resolução. Por outro lado, a *flex*, trabalha com o alinhamento dos elementos, distribuindo-os com o padrão configurado pela tela, como por exemplo, o *space-between*, que distribui os itens, mantendo o mesmo espaço entre eles (FRAIN, 2022).

O *framework* é um conjunto de códigos e regras que auxiliam e agilizam o desenvolvimento do *software*, seja guiando-o ou com funções prontas (JOHNSON; FOOTE, 1988). No entanto, além do CSS puro, há *frameworks*, que podem ser divididos em Componente, ou seja,

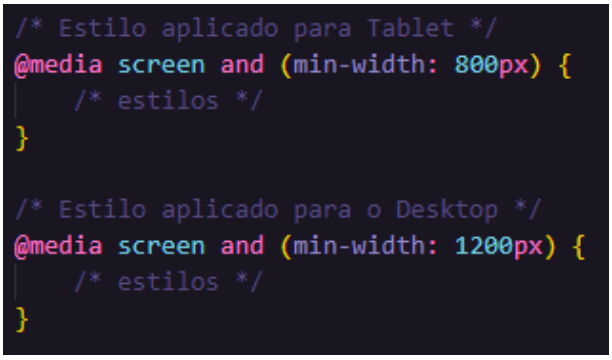


Figura 1 – Exemplo de utilização da *media query*

Fonte: Adaptado de (FRAIN, 2022)

são fáceis de se implementar, com componentes prontos, porém, com o tempo os sistemas ficaram muito semelhantes entre si. Ou em Utilidade, aplicado no nível mais baixo do código, permitindo mais flexibilidade no *design* (GERCHEV, 2022).

Um exemplo de *Framework* do tipo componente é o Bootstrap (GERCHEV, 2022), que é um projeto de código aberto, que utiliza CSS e JavaScript para entregar componentes e estilos prontos (SPURLOCK, 2013). Esse projeto, possui estilos globais, botões e até mesmo um *grid* pré definidos (SPURLOCK, 2013; GERCHEV, 2022). O Bootstrap é recomendado para projetos que o objetivo é construir o sistema rapidamente, sem muitas customizações (SPURLOCK, 2013).

Outrossim, um *framework* de utilidade é o Tailwind, que permite a criação de classes reutilizáveis, através de blocos de estilos, cobrindo as propriedades do CSS e podendo ser customizado (GERCHEV, 2022). O Tailwind entrega uma liberdade para customizar e misturar os diferentes estilos pré estabelecidos, como a "*font-bold*"e a "*italic*", que aplicam as propriedades de negrito e itálico, respectivamente, no texto (GERCHEV, 2022).

Outra abordagem que pode agilizar o desenvolvimento de sistemas responsivos é a abordagem *mobile first*, prática que orienta os desenvolvedores a inicializar o desenvolvimento pelas resoluções dos dispositivos móveis (MARCOTTE, 2017). Essa metodologia é suportada e utilizada pelos *frameworks*, como o Bootstrap e o Foundation, refletindo diretamente nas suas configurações de *layout* utilizando o *grid*, como por exemplo, a classe "*col-md-6*", que define que o elemento ocupará metada da página em dispositivos médios, como tablets (MARCOTTE, 2017; SPURLOCK, 2013; WROBLEWSKI, 2012).

No entanto, esses *frameworks* podem trazer pacotes completos dos seus componentes que pesam o tamanho total do sistema desenvolvido, mesmo que, seja utilizado apenas alguns elementos. Para resolver esse problema, há a possibilidade de importar exclusivamente os elementos utilizados ou até mesmo usar versões minimizadas dos códigos, que excluem as partes não utilizadas para diminuir o tamanho do pacote (ZAKAS, 2010). Caso essas medidas não sejam tomadas, o *site* sendo mais pesado afeta tanto o Otimização para Mecanismos de Busca (SEO) - tornando o sistema menos atrativo para as buscas - como para o carregamento geral, princi-

palmente em dispositivos móveis (SHIVAKUMAR, 2020).

Ademais, os *frameworks* auxiliam nas configurações sobre a acessibilidade da página, como por exemplo, o Bootstrap que inclui os elementos auxiliares nos componentes, direcionando o foco do teclado e auxiliando os leitores (CHALLAPALLI *et al.*, 2021; SPURLOCK, 2013). Porém, mesmo com essas ferramentas na mão, é necessário que o desenvolvedor tenha conhecimento de como configurar a estrutura da página, caso contrário, pode ter pouco contraste, navegação confusa e até mesmo uma *interface* polúida (POWELL, 2010).

Tendo em vista esses pontos, cada *framework* ou tecnologia tem seus pontos fortes e negativos, cabendo ao desenvolvedor escolher a que melhor o atende. Dessa forma, pode-se observar a Tabela 1 para identificar um breve resumo dos pontes de cada tecnologia.

Tabela 1 – Comparação entre as tecnologias para estilização

	Bootstrap	**Tailwind**	**CSS**
Velocidade de construção	Alta	Média	Baixa
Customização	Média	Alta	Muito alta
Consistência	Alta	Média	Média
Tamanho do pacote	Grande	Pequeno	Pequeno
Parâmetros de acessibilidade	Alto	Médio	Baixo

Fonte: (SPURLOCK, 2013; POWELL, 2010; GERCHEV, 2022)

2.4 USABILIDADE E APLICABILIDADE

A usabilidade é um conceito aplicado nos produtos, que buscar medir a qualidade da experiência dos utilizadores com o sistema. Nos sistemas *web* com o grande volume de informações, é necessário ter a atenção a lógica e a estruturação do projeto, garantindo uma navegação intuitiva, design claro e uma busca eficiente (TERRA *et al.*, 2008). Dessa forma, acessibilidade, usabilidade e responsivadade tornam-se termos correlatos (TERRA *et al.*, 2008).

Para garantir o modelo de qualidade dos *softwares* a *International Organization for Standardization* (ISO) define um conjunto de padrões na construção dos sistemas, como o ISO/IEC 9126 (BOTELLA *et al.*, 2004). Esse padrão define características gerais do sistema, relacionados com os atributos mensuráveis, como funcionalidades, usabilidade, acessibilidade e portabilidade - garantindo que pessoas com limitações consigam utilizar o sistema (BOTELLA *et al.*, 2004).

Para validar um *software* se deve realizar pesquisas e coletar *feedbacks* dos usuários, um exemplo de ferramenta é o PSSUQ - que consiste em um questionário com 19 perguntas (LEWIS, 2002) - listadas no Apêndice A. O PSSUQ tem como meta validar a satisfação dos usuários com o sistema com um baixo custo que é aplicado em nível internacional (ROSA *et al.*, 2015; LEWIS, 2002). Para cada pergunta o participante deve dar uma nota de 1 a 7 ao sistema,

sobre um determinado tópico, como por exemplo "Este sistema foi simples de utilizar"(ROSA *et al.*, 2015).

No entanto, o PSSUQ é uma método originado em inglês, por isso, foi necessário ter um trabalho de tradução do mesmo para o Português, permitindo sua aplicação no Brasil, validando com a participação de tradutores e investigadores (ROSA *et al.*, 2015). Com o fim do formulário, deve-se analisar as respostas, calculando a média aritmética delas e separando nas seguintes categorias segundo (LEWIS, 2002):

- **Geral:** respostas 1 a 19
- ***SysUse* (Utilidade do sistema):** respostas 1 a 8
- ***InfoQual* (Qualidade da informação):** respostas 9 a 15
- ***IntQual* (Qualidade da *interface*):** respostas 16 a 18

Para realizar esses testes, (NIELSEN, 1994) recomenda que se teste com ao menos 5 usuários, garantindo que se consiga observar 85% dos problemas de usabilidade, tendo poucas descobertas novas com o aumento da quantidade de participantes.

Outrossim, há a prática de *onboarding*, que é o processo de integrar e introduzir uma pessoa nova a um produto. Quando o usuário abre o sistema pela primeira vez, ele terá que consumir muitas informações para entender completamente o sistema, necessitando de uma curva de aprendizado para ter pleno controle da ferramenta (SANTOS *et al.*, 2025). Essa prática é essencial para garantir a retenção dos usuários, considerando que a primeira utilização do aplicativo é refletida na volta do cliente (LEE; LAM; HUI, 2025).

Para a construção de um *onboarding* intuitivo, há práticas sugeridas como: utilizar textos flutuantes que se sobrepõe, dando enfase em um conteúdo específico, permitir que o usuário navegue com autonomia, permitir que as instruções possam ser ocultadas e mostradas novamente e explique como ler e utilizar as ferramentas, como pode ser observado na Figura 2 (STOIBER, 2024).

Ademais, é de suma importância, que esses ajustes e funcionalidades não coloquem entraves para a experiência do usuário, visto que, a necessidade de muitas ações ou preenchimentos, ocasionam em um desencorajamento de um potencial cliente. Outrossim, um pilar essencial nas aplicações é a velocidade e a simplicidade, é necessário que o utilizador, consiga encontrar as funcionalidades do produto facilmente. Todas essas práticas, somadas a acessibilidade, geram um aumento na produtividade e aprendizado do usuário, garantindo que tenha o máximo de aproveitamento do sistema (RODRIGUES; BECHER, 2008).

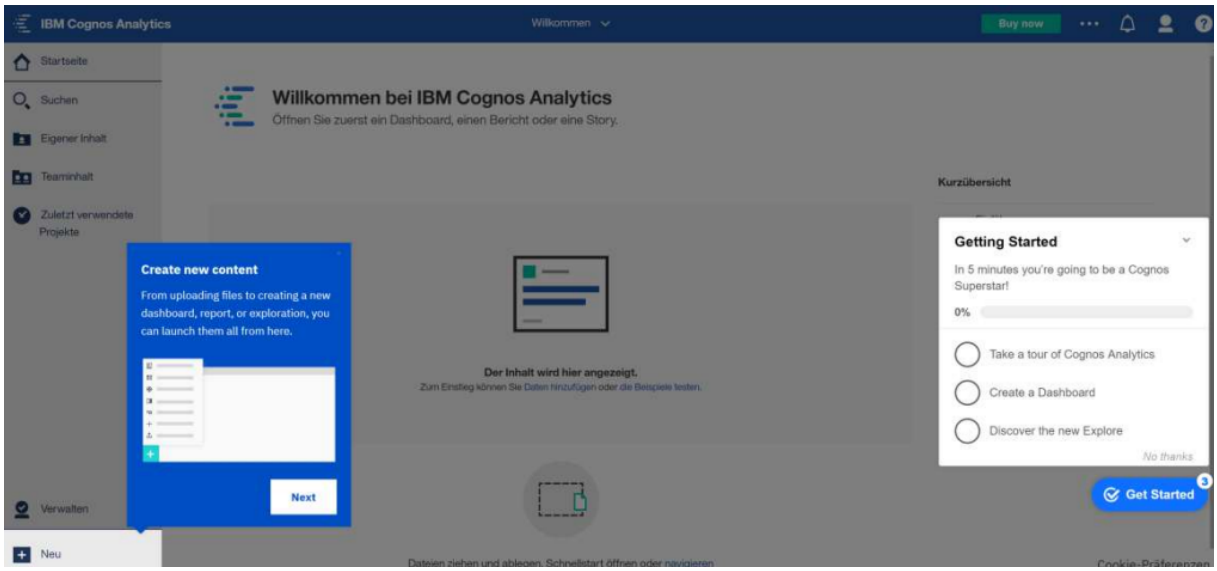


Figura 2 – Exemplo de onbording

Fonte: (STOIBER, 2024)

3 TRABALHOS CORRELATOS

Esta seção aborda e analisa projetos e trabalhos desenvolvidos com objetivos e propósitos em linha com este trabalho.

A pesquisa dos trabalhos relacionados, utilizou o ScienceDirect com palavras chaves: *responsiveness*, *web* e *mobile*, dos últimos 5 anos. Desses trabalhos foi realizado um filtro pelo *abstract*, identificando os artigos que mais tem relação com o presente trabalho.

Na mesma linha do desenvolvimento web, pode-se analisar a refatoração de códigos, abstraindo as classes, segmentando as funções maiores e renomeando métodos para serem claros, gerando um código mais reduzido e claro para os programadores (BARROZO; VINHAS; REIS, 2012). Outrossim, em aplicações web é primordial ter segurança nos sistemas, criando mecanismos de segurança para ataques como o *Cross site scripting* e o *SQL Injection* (UTO; MELO, 2009).

Ademais, é de suma importância elencar os principais pontos que devem ser observados ao desenvolver aplicações web, seguindo algumas boas práticas como: criação de componentes modulares, utilização de boas práticas e que interfaces de dados bem definidas (LOUDON, 2018). Ressaltando-se que geralmente se pode utilizar as linguagens HTML, CSS, Javascript e PHP (LOUDON, 2018).

Antes de aprofundar mais nos artigos é importante definir as principais frentes de pesquisa. Primeiramente, a seção 3.1 apresenta trabalhos que foram utilizados por professores e estudantes para auxiliar no aprendizado da programação, principalmente nas disciplinas iniciais, podendo aumentar a curva de aprendizado.

Já na seção 3.2, é analisado alguns exemplos de aplicativos web legados que foram migrados para soluções modernas, visando ter aplicações responsivas e que funcionem em dispositivos móveis.

Por último, a seção 3.3, lista pesquisas e estudos sobre a responsividade em páginas web, servindo de norte para as decisões das melhores escolhas para a interface.

3.1 FERRAMENTAS PARA O AUXÍLIO DO APRENDIZADO NA PROGRAMAÇÃO

Primeiramente, vale ressaltar as *Integrated Development Environment* (IDE)s web voltadas para o aprendizado da programação de C e C++, como o OnlineGDB. Essa ferramenta é um compilador e depurador web - utilizando o *GNU Debugger* (GDB) - que permite criar e executar programas em linguagens de programação, como o C e C++. O OnlineGDB, possui uma interface semelhante as IDEs utilizadas pelos estudantes, facilitando o seu uso para os alunos

que estão tendo seu primeiro contato com a programação, conforme a Figura 3 (ONLINEGDB, 2025).

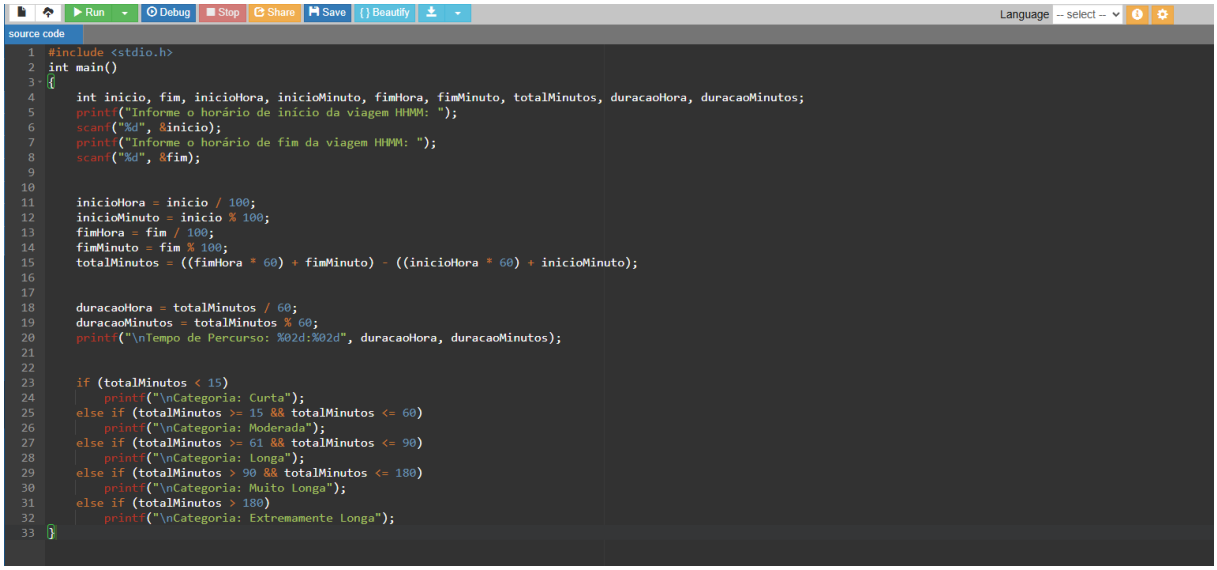


Figura 3 – Exemplo de código no OnlineGDB no Desktop na resolução 1920x1080

Fonte: (ONLINEGDB, 2025)

No entanto, ao utilizar o OnlineGDB em um celular, percebe-se que ele não se adapta responsivamente em um navegador do celular, escondendo parte do código, como pode ser observado na Figura 4, no item A. Dessa forma, o usuário necessita constantemente ficar movimentado a tela para visualizar todo o código de uma linha.

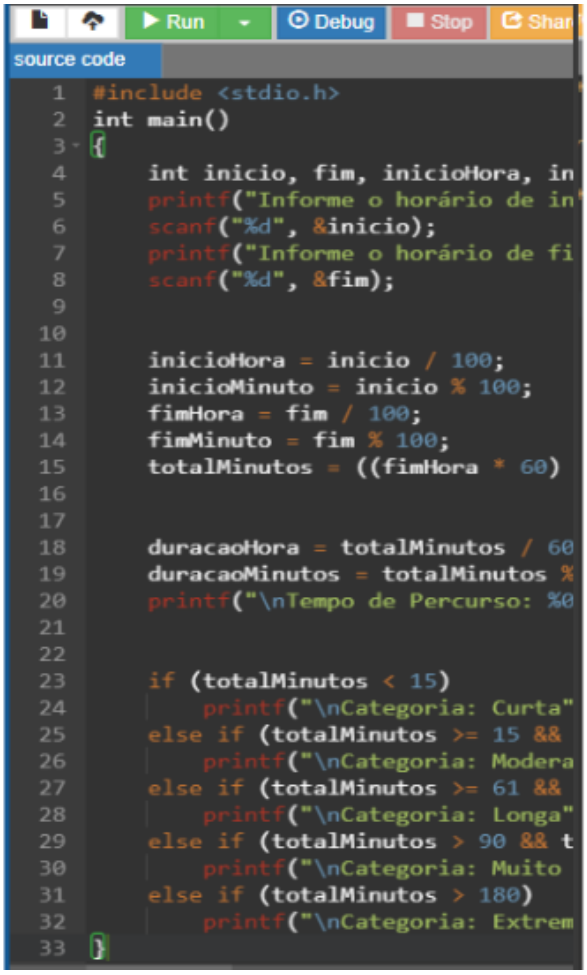
Outrossim, vale destacar que essa ferramenta possui uma seção dedicada para ensinar a programar passo a passo, tanto em C como C++. Essa, pode ser usada por iniciantes ou pessoas mais experientes, contendo conteúdos sobre a sintaxe e lógica, além de possuir exercícios práticos para testar os conhecimentos (ONLINEGDB, 2025).

Em contrapartida ao compilador web, há também ferramentas nativas para os dispositivos móveis, como o Cxxdroid, desenvolvido para exclusivamente para celulares com o sistema operacional Android e com uma interface voltada para telas menores. Dessa forma, observando a Figura 4, no item B, utilizando um Motorola Edge Neo 40, com a resolução de 1080x2400 pxs, percebe-se que a tela se adapta para mostrar todo conteúdo sem a necessidade de arrastar ou movimentar a tela para o lado (IIEC, 2025).

Ademais, um dos principais diferenciais do aplicativo é seu funcionamento offline, ou seja, não tem a necessidade de estar conectado na internet. Dessa forma, os estudantes podem usufruir dessa ferramenta em qualquer situação, necessitando apenas do seu celular e do aplicativo baixado, conseguindo rodar seus testes e realizar os exercícios necessários (IIEC, 2025).

No entanto, além dos compiladores, existem ferramentas dedicadas aos estudantes, como o The Husley, uma ferramenta web com problemas prontos para os alunos testarem seus conhecimentos em diferentes tecnologias, como C, Python e C++ (JUNIOR; MORAIS, 2020).

A.



B.

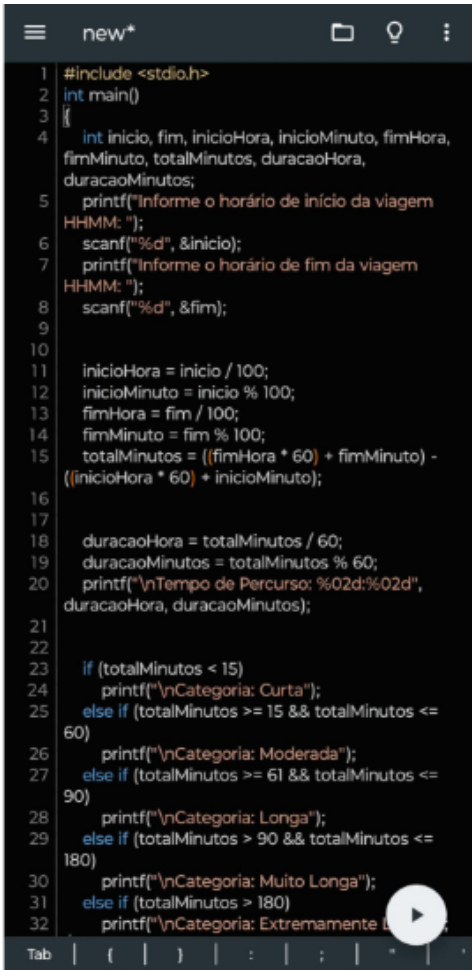


Figura 4 – Exemplo de código no OnlineGDB e Cxxdroid no celular

Fonte: (ONLINEGDB, 2025), (IIEC, 2025)

Destacando-se por corrigir automaticamente os algoritmos criados - analisando as saídas retornadas pelo código criado e comparando-as com uma lista fechada de casos de testes - e retornando para o usuário um feedback (JUNIOR; MORAIS, 2020). Além disso, o The Husley se destaca por sua abordagem gamificada, que visa aumentar o engajamento dos estudantes, apresentando um sistema de pontuação e *ranking*, onde os alunos acumulam pontos ao resolverem os problemas propostos, criando um ambiente de aprendizado mais dinâmico e competitivo

Além disso, o The Husley se destaca por sua abordagem gamificada, que visa aumentar o engajamento dos estudantes. A ferramenta apresenta um sistema de pontuação e ranking, onde os alunos acumulam pontos ao resolverem os problemas propostos, criando um ambiente de aprendizado mais dinâmico e competitivo. A plataforma oferece uma variedade de exercícios que abrangem desde conceitos básicos até tópicos mais avançados, permitindo uma progressão contínua no aprendizado. A correção automática é realizada por meio da comparação da saída do código submetido com saídas esperadas para uma série de casos de teste, proporcionando um feedback imediato que é crucial para o processo de aprendizagem, especialmente em um contexto de estudo independente ou em turmas numerosas onde o acompanhamento individu-

alizado seria um desafio. Essa combinação de desafios práticos, recompensas gamificadas e correção instantânea posiciona o The Husley como uma ferramenta auxiliar valiosa para consolidar o conhecimento de programação. (JUNIOR; MORAIS, 2020)

O Beecrowd é uma ferramenta que também contém problemas e avalia as soluções, mas se destaca pela prática da gamificação (CRUZ *et al.*, 2022). Realizando isso através de ranks globais e locais, podendo incentivar os estudantes de uma turma a realizar uma competição saudável (CRUZ *et al.*, 2022).

Além desses softwares já disponíveis publicamente, há artigos que abordam a criação de uma ferramenta própria para auxiliar no ensino. Como o Algo+, um portal para auxiliar os professores, que pode ser utilizado de forma autônoma pelos alunos, focando principalmente nos princípios da programação (AMARAL *et al.*, 2017). Vale ressaltar, que esse projeto realizou uma pesquisa de satisfação, com turmas dos cursos de Engenharia - totalizando 47 alunos - na qual, 86% classificou como interessante a nova organização dos estudos e 94% recomendaria esse Portal para outros colegas (AMARAL *et al.*, 2017).

Em suma, há inúmeras ferramentas que os programadores podem utilizar, podendo ser uma tarefa árdua escolher qual a mais adequada para o nível de experiência e a linguagem usada, principalmente para os estudantes brasileiros que não tem fluência no inglês (JUNIOR; MORAIS, 2020). Por isso, é interessante realizar um mapeamento delas e entender quando foram desenvolvidas, conforme a Figura 5. Além disso, pode se observar a Tabela 2, com uma breve comparação do ano, propósito, se é responsivo ou não e qual a forma de acesso.

Figura 5 – Linha do tempo da criação das ferramentas

Fonte: Autor

Tabela 2 – Tabela comparativa das ferramentas de programação

Ferramenta	Ano	Propósito	Responsivo	Forma de acesso
Dev C++	1998	IDE para C/C++	Não	Instalado
CodeBlocks	2005	IDE para C/C++	Não	Instalado
WebAlgo	2009	Plataforma de aprendizado de programação	Não	Instalado
OnlineGDB	2010	IDE online para múltiplas linguagens	Sim	Online
Algo+	2017	Ferramenta de ensino de algoritmos	Sim	Online
Cxxdroid	2018	IDE para dispositivos Android	Sim	Aplicativo móvel
Beecrowd	2021	Plataforma de ploblemas de programação	Sim	Online

Fonte: Autor

Figura 6 – Protótipo do EcuadorLegalOnLine

Fonte: (CAJAS *et al.*, 2021)

3.2 MIGRAÇÃO DE APLICATIVOS WEB LEGADOS

Esta seção aborda trabalhos que realizaram a migração de aplicativos web legados, servindo para o desenvolvimento do presente trabalho. Modificando sites antigos que podem não ser responsivos para os dispositivos móveis. Para isso, uma das primeiras tarefas é identificar se um site é legado, podendo, para isso, abrir o sistema através de um celular e verificar se possui problemas de renderização visíveis ou se é necessário dar *zoom* ou um *scroll* para realizar determinada função (CAJAS *et al.*, 2021).

Após identificar uma ferramenta que precisa ser modificada, existem etapas que devem ser desenvolvidas, como, priorizar as funcionalidades mais utilizadas, discutir as alternativas, definir as melhorias e por último refatorar a interface (CAJAS *et al.*, 2021). Um exemplo, seguindo esse modelo, é o Ecuador Legal Online, com informações espalhadas pela tela - que tornava a experiência em dispositivos móveis inacessível (CAJAS *et al.*, 2021). Portanto, foi elencado as principais funções do sistema e adaptado para tornar responsivo o aplicativo web, como por exemplo, o menu que foi transformado em um menu lateral que se esconde e não atrapalha a visualização do sistema, como pode ser visto na Figura 6 (CAJAS *et al.*, 2021).

No entanto, conforme o estudo que analisou adaptações entre 2006 e 2017, identificou-se que a migração de sistemas legados para a plataforma móvel é uma tarefa complexa, pois demanda a especialização de um programador para investigar e implementar a melhor forma de tornar a interface acessível e prática. Como solução, o autor sugere a adaptação da estrutura do DOM ou de arquivos *Extensible Markup Language* (XML), complementada pela criação de

protótipos de tela. Essa abordagem não apenas facilita o processo de adaptação, mas também permite uma melhor segmentação das responsabilidades entre os profissionais envolvidos no projeto (CAJAS *et al.*, 2020).

Outro exemplo, seguindo a mesma abordagem do Ecuador Legal Online, foi a migração da *The Academic Management System* da Universidade da Indoamérica, modificando a interface para dispositivos móveis (CAJAS *et al.*, 2019). Vale ressaltar, que para esse projeto foi realizada uma pesquisa com dez professores, o público alvo desse sistema, visando entender se as mudanças realmente tornam a experiência mais fácil - instruindo-os a realizar tarefas, como localizar informações dos seus apontamentos (CAJAS *et al.*, 2019). Após realizar essa breve pesquisa, percebe-se que reduziu significativamente o tempo para realizar determinada tarefa (CAJAS *et al.*, 2019).

Outra abordagem é a utilização de *Frameworks*, como o desenvolvido por Almonaies, que consegue automatizar a adaptação de aplicações web em PHP, para ambientes *Service-Oriented Architecture* (SOA) . Sua abordagem ainda precisa de melhorias para suprir todas funcionalidades da linguagem, porém, melhorias podem ser feitas, respeitando as etapas dele: refatoração e separação dos componentes, definição dos tipos, conversão dos componentes e a refatoração da base de dados. Essa refatoração é subdividida em 5 etapas, sendo elas, refatoração e separação dos serviços candidatos, inferir os tipos, conversão dos componentes e por último a refatoração do banco (ALMONAIES *et al.*, 2013).

3.3 RESPONSIVIDADE EM PÁGINAS WEB

Por último, para realizar essa tarefa é essencial entender as práticas e diretrizes para aplicar a responsividade nas aplicações. Dessa forma, deve-se respeitar alguns princípios, como algumas elencadas por Jacob Nielsen (NIELSEN, 1994):

- A interface deve sempre informar o seu status atual de forma clara
- Siga sempre os padrões do mercado, evitando uma carga cognitiva desnecessária
- Foque no essencial, priorizando sempre mostrar apenas as informações essenciais
- Utilizar linguagem simples para facilitar o entendimento claro

Para isso, as aplicações web utilizam como tecnologias base o HTML e o CSS, buscando atender esse público crescente dos celulares, sem abandonar as telas grandes de computadores e notebooks (SUN; CAO, 2014). Além dessas, pode ser utilizado também o React Native, Flutter, Ionic, *jQuery Mobile* e o PhoneGap, permitindo o acesso às funcionalidades nativas e garantindo um desempenho adequado para as aplicações (SUN; CAO, 2014).

Dessa forma, com o aumento do uso de dispositivos móveis, há desafios para adaptar os layouts e manter as funcionalidades em todas telas, principalmente por causa do espaço pequeno em dispositivos menores (KIM, 2013). Portanto, deve-se utilizar os *layouts* fluídos, evitando utilizar medidas fixa como os px; usufruir das funcionalidades mais novas do CSS, como o *media query* - que personaliza os estilos de acordo com o tamanho da tela e a configuração correta das *meta tags viewport* (MONTEIRO *et al.*, 2020; GARDNER, 2011).

Ademais, além das soluções manuais, há a possibilidade de utilizar *frameworks*, que possui estilos, componentes e um layout adequado, acelerando o desenvolvimento e padronizando o código (SPURLOCK, 2013). Essa ferramenta, se adapta de acordo com o tamanho da tela do dispositivo e possui funcionalidades prontas, como botões, títulos, menus, *badges* e até mesmo paginações (SPURLOCK, 2013).

Para garantir essa responsividade é necessário que as aplicações sejam prototipadas previamente, garantindo que o cliente tenha uma visão do que será entregue e que o desenvolvedor tenha mapeado seu trabalho (NASCIMENTO *et al.*, 2020). Um exemplo de ferramenta para criar protótipos é o Figma, utilizado para prototipar aplicativos, sites e como comentado por (NASCIMENTO *et al.*, 2020), foi utilizado em um aplicativo de ensino de saúde.

De acordo com a linha de se ter muitos sistemas parecidos (GERCHEV, 2022), a pesquisa de (SALMI, 2023) demonstra que há *frameworks* para garantir a responsividade das páginas mais utilizados, como o Bootstrap, Foundation, Skeleton, Bulma e o Tailwind CSS, respectivamente, de acordo o Google Trends e até mesmo a popularidade do repositório.

A responsividade se mostra necessária, podendo aumentar o número de usuários e permitindo que pessoas com acesso apenas a dispositivos móveis possam realizar certas tarefas, como foi o caso do COVID *Pass*, aplicativo que gera a triagem dos sintomas do COVID (ZHANG *et al.*, 2020). Sistema que foi utilizado por cerca de 2 milhões de pacientes, tendo apenas desses, 8.1% preenchidos manualmente, podendo liberar os funcionários para realizar funções mais críticas do hospital (ZHANG *et al.*, 2020).

No entanto, não basta apenas modificar o *layout* do sistema para se adaptar as resoluções menores, é necessário manter a acessibilidade também. Dessa forma, um *site* deve garantir que funcione e se adapta no mínimo a largura de 320px e a altura de 256px - garantindo que até mesmo com o zoom, seja possível visualizar os componentes. Tendo em vista isso, recomenda-se que os botões que tenham alguma ação, tenham no mínimo 24px, certificando-se que sejam clicáveis, porém, o sugerido é seguir o padrão da W3C, tendo no mínimo 44px (GITHUB, 2023)

Além disso, é de suma importância verificar que a disposição ou funcionalidades não fiquem limitadas apenas a ação de *hover* - ação de manter o cursor ou toque prolongado sob um componente - do usuário, visto que, celulares ou tablets não suportam essa função (GITHUB, 2023).

Há bibliotecas do JavaScript que foram desenvolvidas para permitir a divisão da apli-

cação em camadas, segregando as funcionalidades, complexidades e garantindo que sigam os padrões como o *Model-View-Controller* (MVC), sendo elas o React, Vue.js, Bootstrap, AngularJS, jQuery e o Node.js. Essas tecnologias possibilitam que o *software* seja separado em dois serviços, sendo um o Cliente e outro o Servidor, que se comunicam e entregam uma página amigável para dispositivos móveis (SHAHZAD, 2017).

As aplicações *web* tem se mostrado muito úteis no dia a dia e no âmbito profissional, podendo ser utilizada para a classificação de doenças ou até mesmo para auxiliar indivíduos com Parkinson a calcular a quantidade de levodopa que deve ser aplicada (DRISS *et al.*, 2024; VERBER *et al.*, 2020). O EQUIDopa utiliza o Angular e o Bootstrap para garantir um sistema que rode nos dispositivos, independente da resolução, podendo ser utilizado por pessoas até mesmo com o Parkinson (VERBER *et al.*, 2020).

A utilização dos sistemas através de dispositivos móveis vem aumentando, conforme a pesquisa de (BERNACKI *et al.*, 2016), que demonstra através de uma pesquisa utilizando o PSSUQ quantos problemas são identificados por diferentes grupos e diferentes resoluções. Esse trabalho demonstra onde está o maior problema de um sistema, podendo mapear recomendações para o futuro do site (BERNACKI *et al.*, 2016).

Além dos aplicativos voltados para a saúde, há também para a educação, como o ODYSC, que utiliza HTML, CSS, JavaScript e o Bootstrap para garantir um *layout* dinâmico. Fato necessário, visto que, 69,64% dos participantes da pesquisa estavam usando celulares para acessar o sistema e o restante em computadores. Esse trabalho, também, realizou uma pesquisa para validar a usabilidade do programa, através de um teste baseado no *Likert-type*, que tem como nota máxima 5 e avalia a parte técnica, interação, valor educacional e o *feedback* do usuário (DEKEMELE; CHEVALIER; LOCCUFIER, 2018).

Um site responsivo trás vantagens para uma aplicação, como manter o mesmo conteúdo - apresentando as informações que aparecem em um computador no celular - e gerenciar apenas um sistema, ajustando o *layout*. No entanto, encontra-se problemas, por exemplo, a página tende a ficar mais pesada, carregando conteúdos que não são nem visualizados na tela ou até mesmo o costume dos usuários em visualizar a tela grande, podendo ter dificuldades de se localizar em dispositivos móveis. Como exemplo, o site da Universidade de McGill, que o *layout* se adapta, porém, a leitura fica maçante, com muitos *links* e textos (KIM, 2013).

Com o surgimento dos *websites*, não existia muitas variações das dimensões dos dispositivos, assim, os designers não precisavam se preocupar com diferentes *layouts*, porém, isso mudou, sendo que uma pesquisa de 2013, demonstrou que 65% das pessoas entre 18 e 29 anos estão acessando a internet por celulares. Dessa forma, é necessário possuir componentes dispostos de maneiras diferentes conforme a resolução, como pode ser visualizado na Figura 7, na qual, os elementos mudam de tamanho, local e disposição (BADER; HAMMOURI, 2016).

Para decidir qual *layout* deve ser construído na tela, há um padrão que se sugere, para

Figura 7 – Exemplo de disposição dos componentes no celular, tablet e computador

Fonte: (BADER; HAMMOURI, 2016)

garantir um bom funcionamento, conforme pode ser observado na Figura 8. Essas divisões são passadas para o código utilizando *media-queries*, aplicando diferentes propriedades de estilo. No entanto, vale ressaltar que é necessário tentar criar disposições fluidas, evitando utilizar valores fixos, como por exemplo, utilizar px, visto que, ao modificar a resolução esse valor se mantém constante, podendo extrapolar a tela e deixar a usabilidade afetada (BADER; HAMMOURI, 2016).

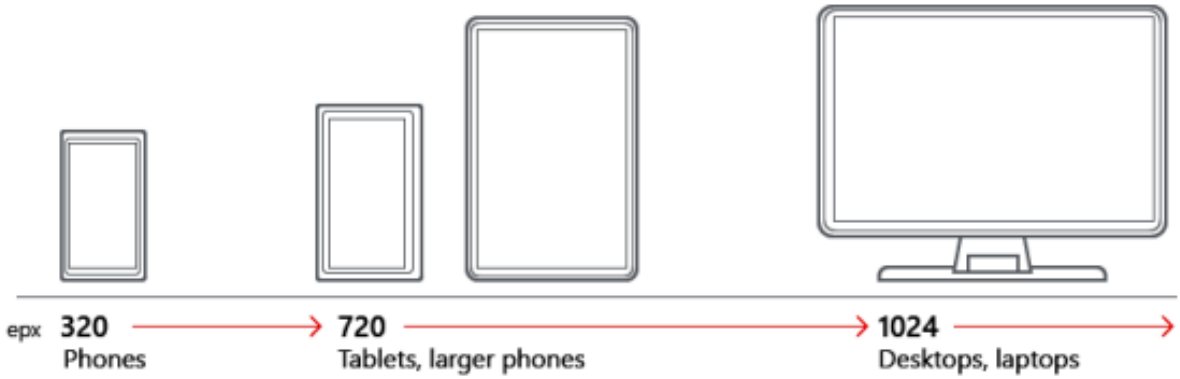


Figura 8 – Divisão dos dispositivos por resolução

Fonte: (BADER; HAMMOURI, 2016)

Dessa forma, uma prática é esconder elementos dispostos na tela da versão computador para quando são convertidos para celulares ou tablets. Porém, com a introdução da renderização de páginas no lado do servidor, há a possibilidade de detectar a resolução antes de carregar a página, para isso se utiliza os cabeçalhos presentes nas requisições, como por exemplo, o *User-Agent* - que contém informações do dispositivo e navegador. Portanto, essas práticas entregam vantagens do desenvolvimento responsivo, como manter apenas uma página, experiência do usuário confortável e menos uso da banda de internet, visto que, as imagens em resoluções menores diminuem, evitando usar uma banda desnecessária (BADER; HAMMOURI, 2016).

4 PROPOSTA DE SOLUÇÃO

Esta seção tem como objetivo apresentar a proposta de solução que disponibiliza o WebAlgo em dispositivos móveis, através de navegadores, utilizando como base os trabalhos correlatos e os fundamentos.

Inicialmente, foi realizado um levantamento para visualizar e enumerar as funcionalidades do WebAlgo, garantindo que estão presentes nas resoluções dos dispositivos móveis. Dessa forma, conforme a Figura 9, há as seguintes funcionalidades:

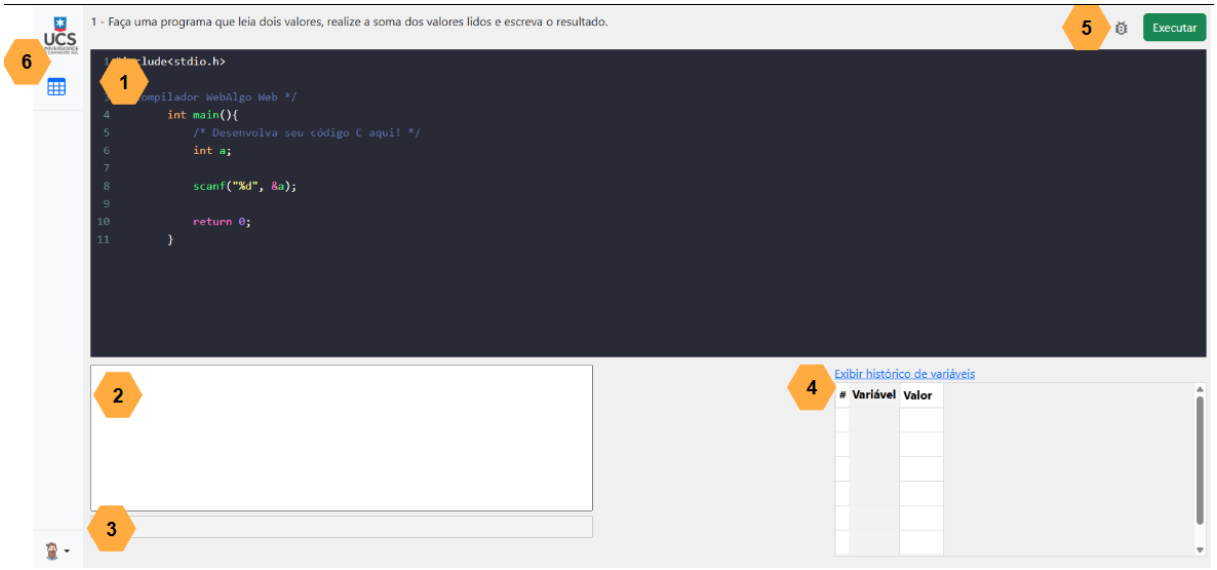


Figura 9 – Página inicial do WebAlgo no computador

Fonte: autor

- **Funcionalidade 1:** editor de código, área destinada a escrita completa do código
- **Funcionalidade 2:** registro de *logs* da aplicação, retornando os erros, *status* e retornos
- **Funcionalidade 3:** campo para entrada de dados quando solicitado
- **Funcionalidade 4:** tabela contendo histórico das variáveis utilizadas
- **Funcionalidade 5:** botões para compilar e executar o código
- **Funcionalidade 6:** menu lateral, com ação de gerar o C3E

Tendo em vista o *layout* presente no WebAlgo, para mapear e entender as funcionalidade é possível gerar um *wireframe*. Essa tarefa fez com que seja possível visualizar para qual local as funcionalidades se locomoveram e se todas estão presentes sempre. Portanto, pode-se observar

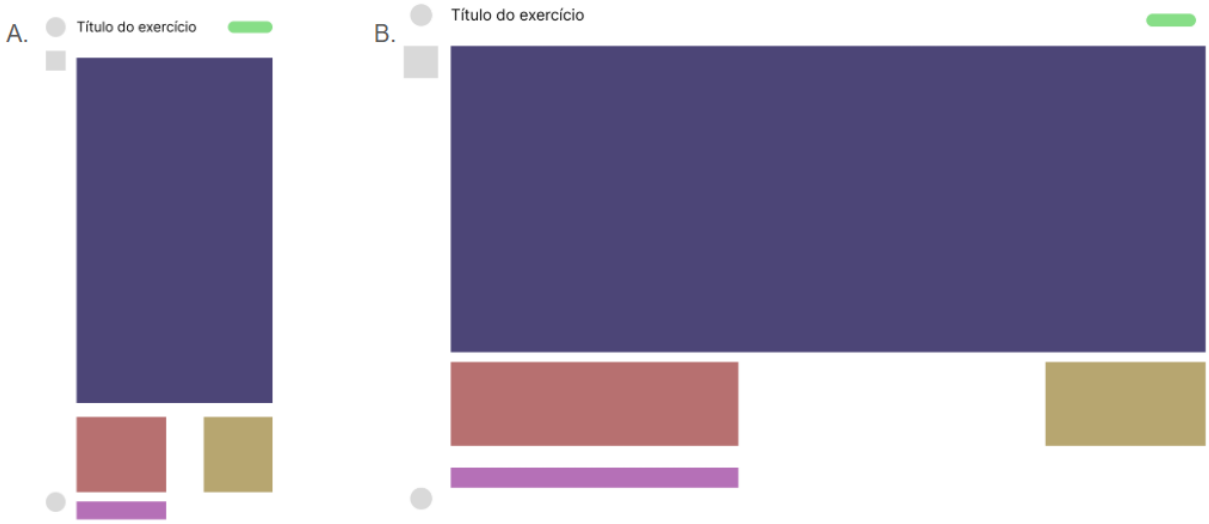


Figura 10 – *Wireframe* do WebAlgo

Fonte: autor

na Figura 10 a versão de dispositivos móveis, no item A, e a versão para computador no item B.

Com as funcionalidades levantadas, foi possível criar um diagrama de caso de uso, conforme pode ser visualizado na Figura 11, que evidência as ações que o usuário deve poder realizar.

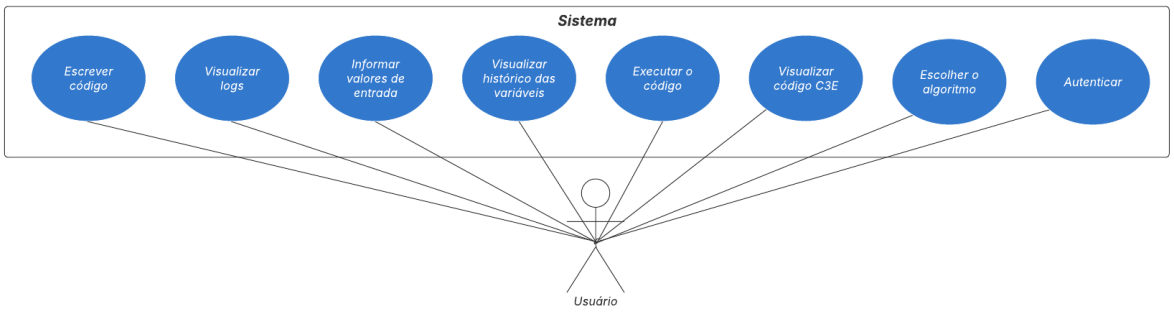


Figura 11 – Diagrama de Caso de uso do WebAlgo

Fonte: autor

O sistema WebAlgo *web* foi desenvolvido inicialmente, por (SUSIN, 2023), utilizando apenas CSS, HTML e JavaScript, sem se preocupar com diferentes dimensões, além da *Desktop*, visto que, não era o objetivo do trabalho realizar essa responsividade. Portanto, para validar a escolha de tecnologia para a estilização, será apresentada uma POC, utilizando como base o WebAlgo, validando como o projeto e bibliotecas existentes se comportam com a instalação do Tailwind ou do Bootstrap, sem realizar nenhum ajuste ou customização. Primeiramente, testou-se o Tailwind 4.1, adicionando a importação completa do *framework* na página, conforme pode ser observado na Figura 12. Com o Tailwind a página não apresentou erros nos componentes

ou espaçamentos, mantendo a consistência, como pode ser observado na tabela de histórico das variáveis.

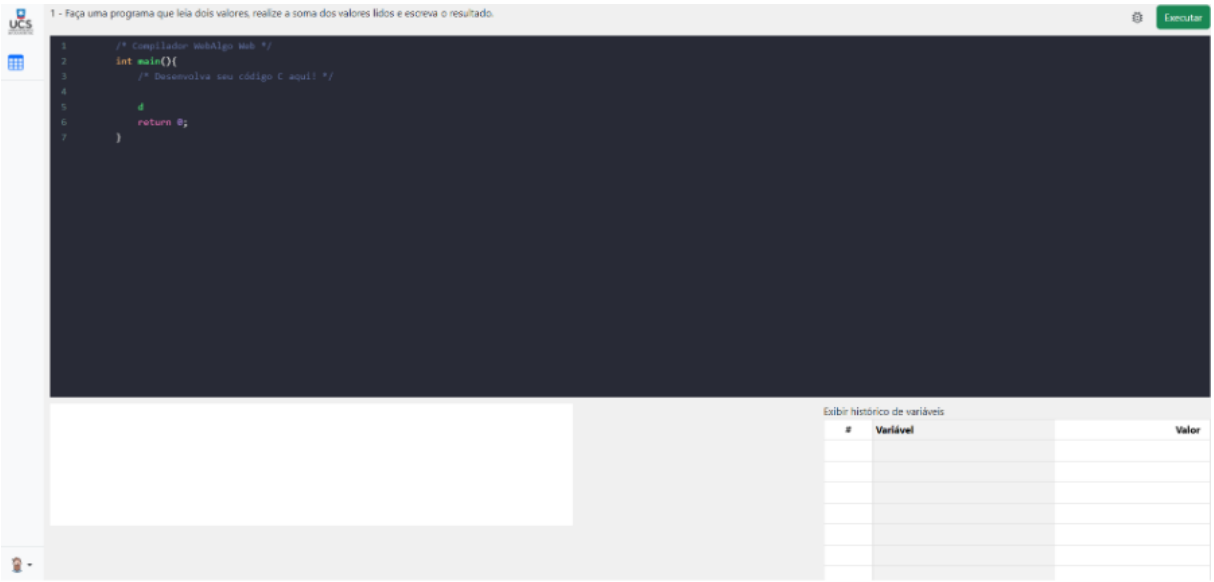


Figura 12 – POC do WebAlgo utilizando o Tailwind

Fonte: autor

Outrossim, utilizando o Bootstrap, importou-se dois arquivos, o arquivo de estilização e o os *scripts* JavaScript, para um *setup* inicial da biblioteca, assim, obteve-se o resultado na Figura 13, o qual modificou o tamanho dos elementos, como o logo da UCS e gerou problemas de *layout*, como a barra de rolagem horizontal e o estilo da tabela que não permite uma distinção das colunas.

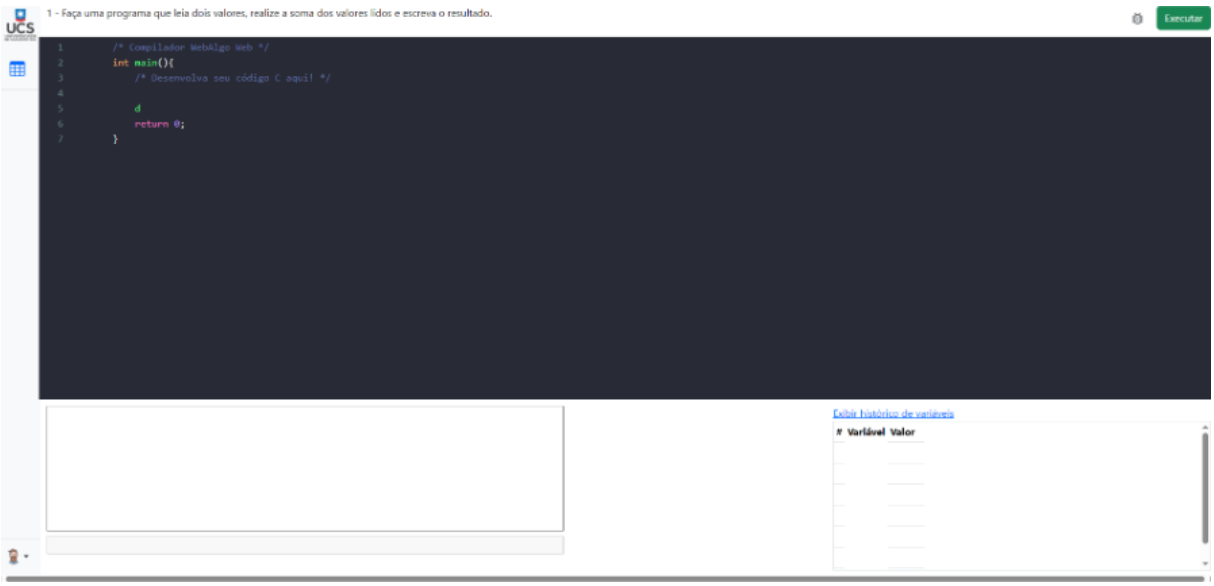


Figura 13 – POC do WebAlgo utilizando o Bootstrap

Fonte: autor

Portanto, com base nas POCs levantadas e os pontos observados nos trabalhos correlatos

Tabela 3 – Notas de desempenho medidas no *PageSpeed Insights* e *LightHouse* no WebAlgo *web* de (SUSIN, 2023)

		Desempenho	Acessibilidade	Práticas recomendadas	SEO
PageSpeed Insights	Computador	100	81	93	82
	Celular	89	82	89	82
LightHouse	Computador	91	81	89	82
	Celular	99	85	93	82

Fonte: autor

e nos fundamentos, a tecnologia escolhida foi o Bootstrap, visto que, ele possibilita customizar o tema do WebAlgo, aproveitando os componentes prontos da biblioteca, o que agilizará o desenvolvimento. Além de ser a tecnologias mais utilizada e possuir uma comunidade para auxiliar, com tabelas, *inputs*, *modals* prontos, apenas modificando para as cores do sistema. Para esse projeto, será utilizado a versão 5.3.6 do Bootstrap, a mais atual, garantindo um suporte maior e as últimas mudanças e correções.

Ademais, independentemente da tecnologia escolhida, foi necessário segregar o sistema em diferentes resoluções, cada uma tendo seus estilos e disposição. Portanto, com base no WebAlgo e nos trabalhos analisados a divisão será feita em três proporções, a para computadores, tendo uma largura mínima de 1024px, a para tablets, que tem como largura mínima 720px e máxima de 1023px e por último a disposição para celulares, tendo como largura máxima de 719px. Essa divisão garante, conforme as pesquisas apresentadas, que os estilos sigam os padrões do desenvolvimento e seja possível criar *layouts* e telas responsivas e amigáveis.

Para complementar a ferramenta, foi implementado também o Google Analytics, ferramenta utilizada para analisar métricas do sistema, como quantas pessoas acessaram, quais dispositivos foram utilizados e como interagem com o aplicativo. Para implementar este serviço será necessário criar uma conta no Google Analytics e inserir alguns *scripts* na página, apontando para a conta criada, garantindo que o sistema consiga mapear as métricas (PLAZA, 2011).

Outro serviço que foi utilizado é o *PageSpeed Insights* e o *LightHouse*, que possibilitam gerar métricas sobre o desempenho, acessibilidade, práticas recomendadas e SEO, com a nota variando de 0 até 100 - sendo 0 a pior nota e 100 excelente. Além disso, vale ressaltar que para ser considerado uma nota boa, o valor deve variar entre 90 a 100. Como pode ser observado na Tabela 3, há as métricas do WebAlgo *web* desenvolvido por (SUSIN, 2023), que possui uma nota boa, muito próxima do 90, mas pode ser melhorada, principalmente para dispositivos móveis.

Para validar todas as etapas desenvolvidas, foi aplicado formulários nas turmas de Programação I, utilizando como base o PSSUQ. Dessa forma, foi possível comparar as notas obtidas com o produto final gerado, podendo tirar a conclusão se o sistema teve melhorias e em quais partes melhorou ou piorou. Sendo os seguintes questionários aplicados: sobre a IDE uti-

lizada pelos participantes para programar, utilizar uma IDE no dispositivo móvel e finalmente exclusivamente para o WebAlgo.

4.1 PROTOTIPAGEM

Outra tarefa necessária, foi realizar a prototipação completa das telas e estados, garantindo que todas funções estarão presentes e que no momento de codificar seja possível seguir uma trajetória linear, sem retrabalhos. Ademais, é necessário que os botões tenham no minímo 24px, garantindo o clique acessível nos dispositivos (GITHUB, 2023). Para isso, utiliza-se o Figma, criando os *layouts* para dispositivos móveis e computadores. Primeiramente, criou-se as telas menores, como pode ser observado na Figura 14, nela pode-se visualizar o modo escuro do aplicativo, com a disposição principal do sistema, sendo dividido em um menu no topo, com as principais ações. Entre elas destaca-se o botão para executar o código, o para realizar o *debug*, uma lista de abas, utilizando o componente *nav-tabs* do Bootstrap, que dividem o conteúdo do algoritmo em: Questão, Código, Resposta, Variáveis e C3E.

Ademais, vale citar que foi utilizado os icónes do próprio Bootstrap, mantendo uma consistência e aproveitando os recursos importados no projeto.

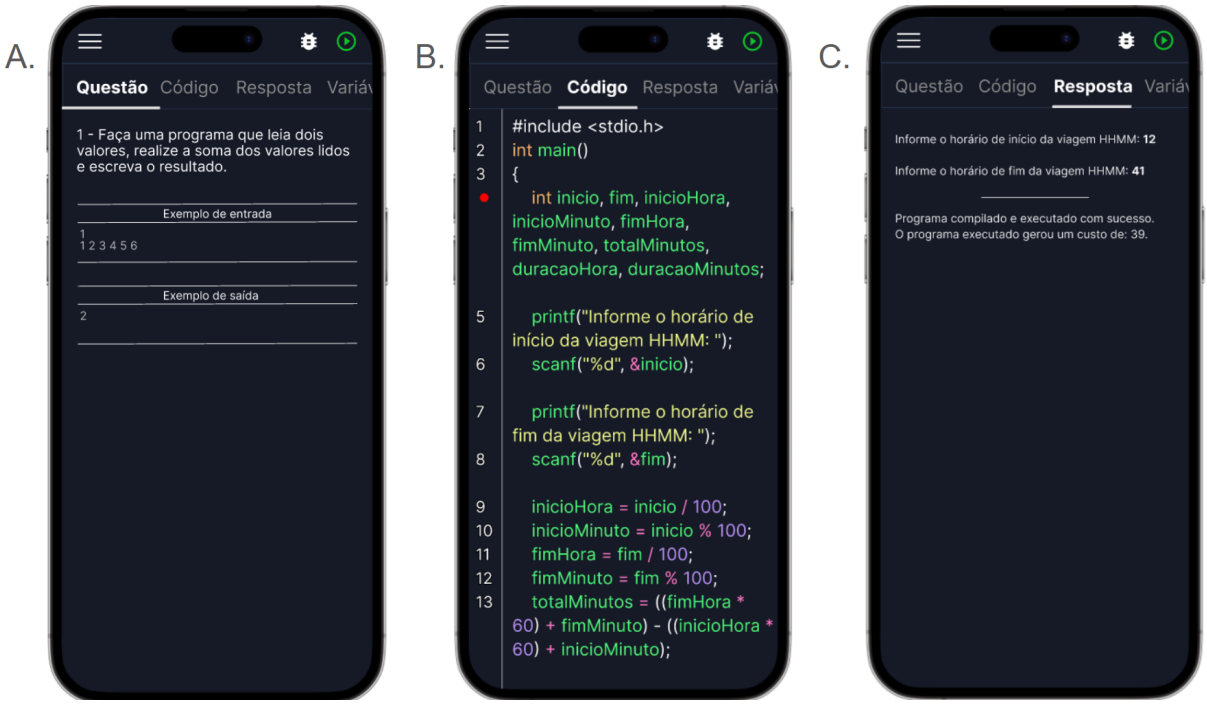


Figura 14 – Protótipos da Questão, Codigo e Resposta no celular

Fonte: autor

Observando a Figura 14, no item A, tem-se a tela que lista as informações do algoritmo escolhido, com uma breve descrição e os exemplos de entrada e saída. Já no item B, contém a principal tela do sistema, a IDE, local que o usuário vai digitar o seu código, contendo um

padrão de cores, de acordo com o tipo. Com o código criado, pode-se visualizar as saídas e os registros da execução no item C, listando também o estado da execução e o custo gerado.

Na Figura 15, pode-se observar as outras abas, primeiramente, no item A, tem-se as variáveis, listando todas instânciadas no código e um histórico completo das suas mudanças. Além disso, com o código executado, pode-se gerar diretamente o C3E - listado no item B - que pode ser utilizado para estudos ou para rodar, com uma breve listagem das linhas. Complementando a tela do código, ilustrada no item B, da Figura 14, tem-se o item C, que demonstra o comportamento de descer a tela, mantendo fixo os menus superiores e apenas modificando a tela na parte do código.

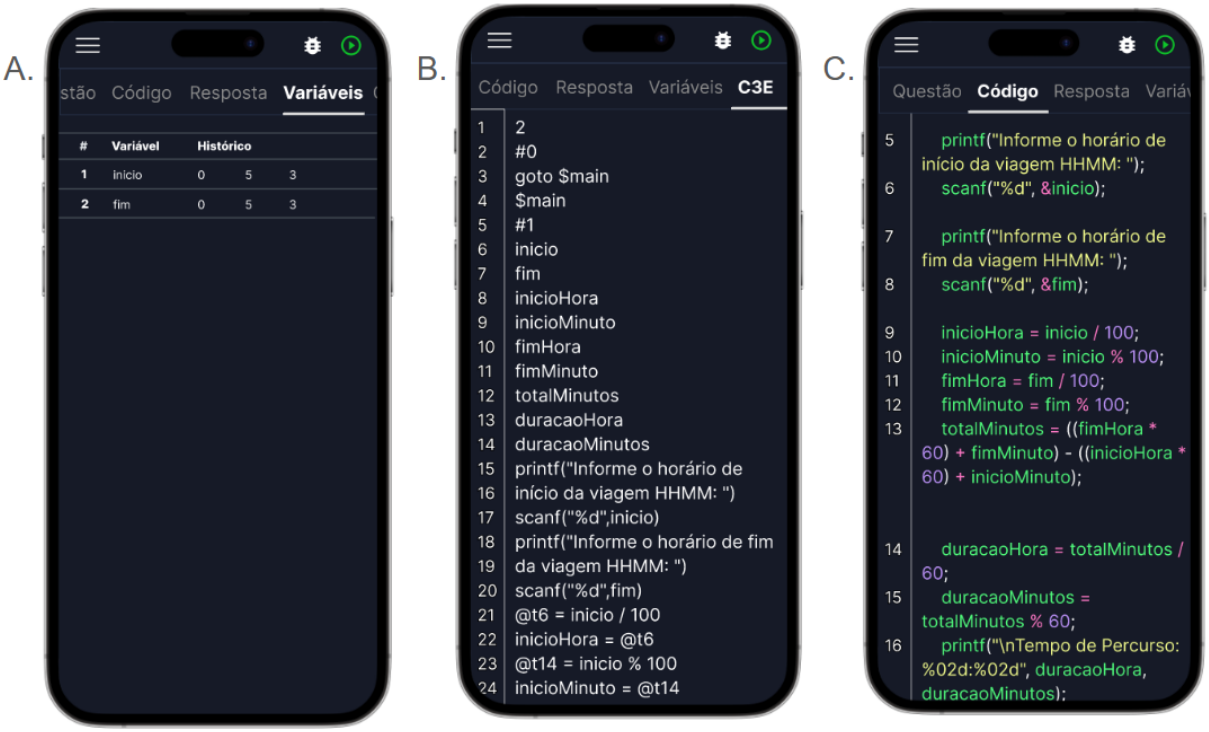


Figura 15 – Protótipos das Variáveis, C3E e código no celular

Fonte: autor

Prosseguindo, criou-se o menu lateral, que pode ser aberto clicando no botão no topo esquerdo da tela, conforme a Figura 15, criando um elemento que se sobrepõe, que pode ser observado na Figura 16, no item B. Esse menu garante que as opções sejam visíveis no celular, sem atrapalhar o conteúdo da tela. O menu apresenta sempre as opções de: ir para a página inicial, selecionar um algoritmo, informações do WebAlgo, ajuste do tamanho da fonte e alteração do tema, variando se está num estado autenticado ou anônimo, conforme pode ser visualizado no item C, que contém um botão de entrar e o estado autenticado no item B, com os dados do usuário. Além disso, no item A, observa-se o estado de carregamento da página, após o usuário clicar para executar o problema, compilando-o e somente depois liberando a página totalmente.

Outrossim, ao rodar um código, há a possibilidade de ser necessário que o usuário preencha algumas entradas para as variáveis, como pode ser visualizado na Figura 17, no item A,

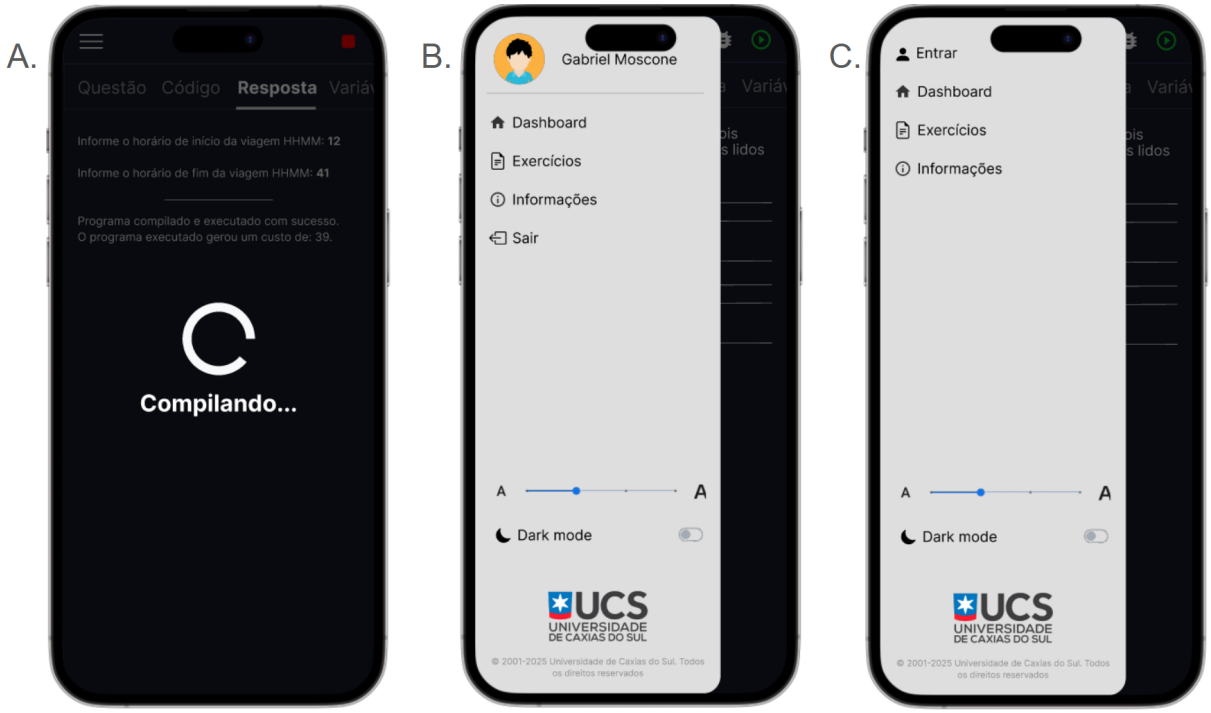


Figura 16 – Protótipo do *menu* lateral e carregamento no celular

Fonte: autor

criando um *modal*, componente do Bootstrap, que se sobrepõe a página, ainda nesse tópico é possível visualizar o estado de um altoritmo rodando, no topo da tela, alterando para um botão de parar, que cancela a execução. Ademais, no item B, tem-se o estado de rodando quando o usuário realizar o *debug*, permitindo rodar todo código, pular para o próximo *breakpoint* e parar, respectivamente. Por outro lado, ao clicar na opção Exercícios do menu lateral, será aberto um *modal* para selecionar um algoritmo, com as opções de definir um tipo e o problema, alterando todas configurações de dados.

Com todas essas funcionalidade é essencial ter um controle do usuário que utiliza o sistema, principalmente, para os professores acompanharem as evoluções dos alunos. Dessa forma, observando a Figura 18, no item A, visualiza-se a tela de autenticação, na qual, deve ser incluído o usuário e senha para obter as informações completas. Por outro lado, caso seja a primeira vez do usuário, há a tela de cadastro, no item B, dividida em passos, sendo no primeiro pedido apenas o nome completo, permitindo avançar para as próximas.

Portanto, na Figura 19, observa-se os próximos passos do cadastro respectivamente, solicitando o estado e cidade, no item A, sempre permitindo avançar ou voltar para corrigir algum dado. Prosseguindo, o usuário pode preencher o gênero, no item B, finalizando as informações pessoais e podendo preencher seu email, que sera utilizado para se autenticar, no item C. Seguindo, com o email validado, no item D, deve-se informar uma senha para proteger a conta, permitindo avançar para o item E, no qual, pode-se informar quaisquer observações para a conta. Com todos esses dados, a conta é criada, mostrando o item F, que informa que deu tudo

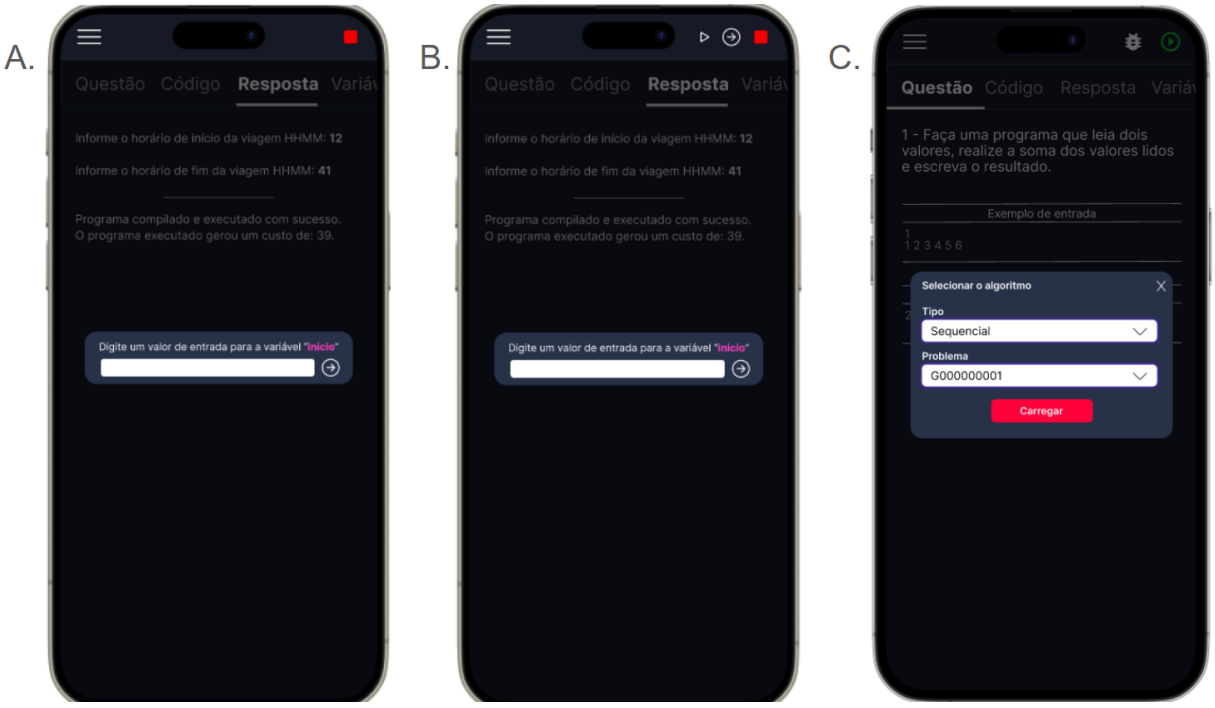


Figura 17 – Protótipo do *modal* de entrada e algoritmo no celular

Fonte: autor

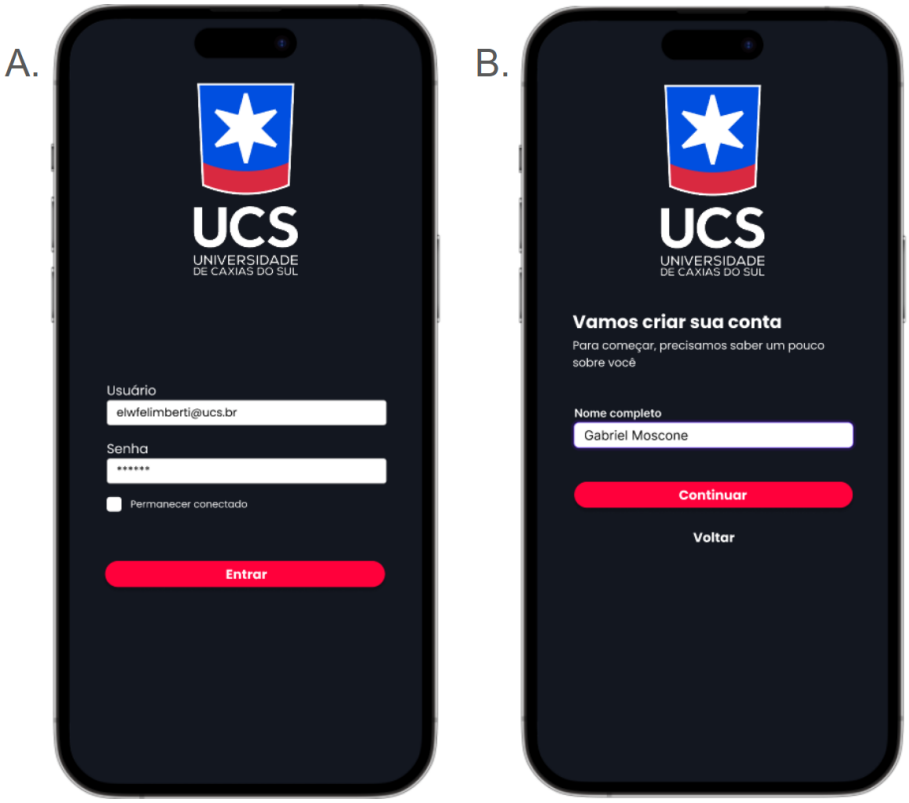


Figura 18 – Protótipo do *login* e cadastro no celular

Fonte: autor

certo e direciona para o sistema.

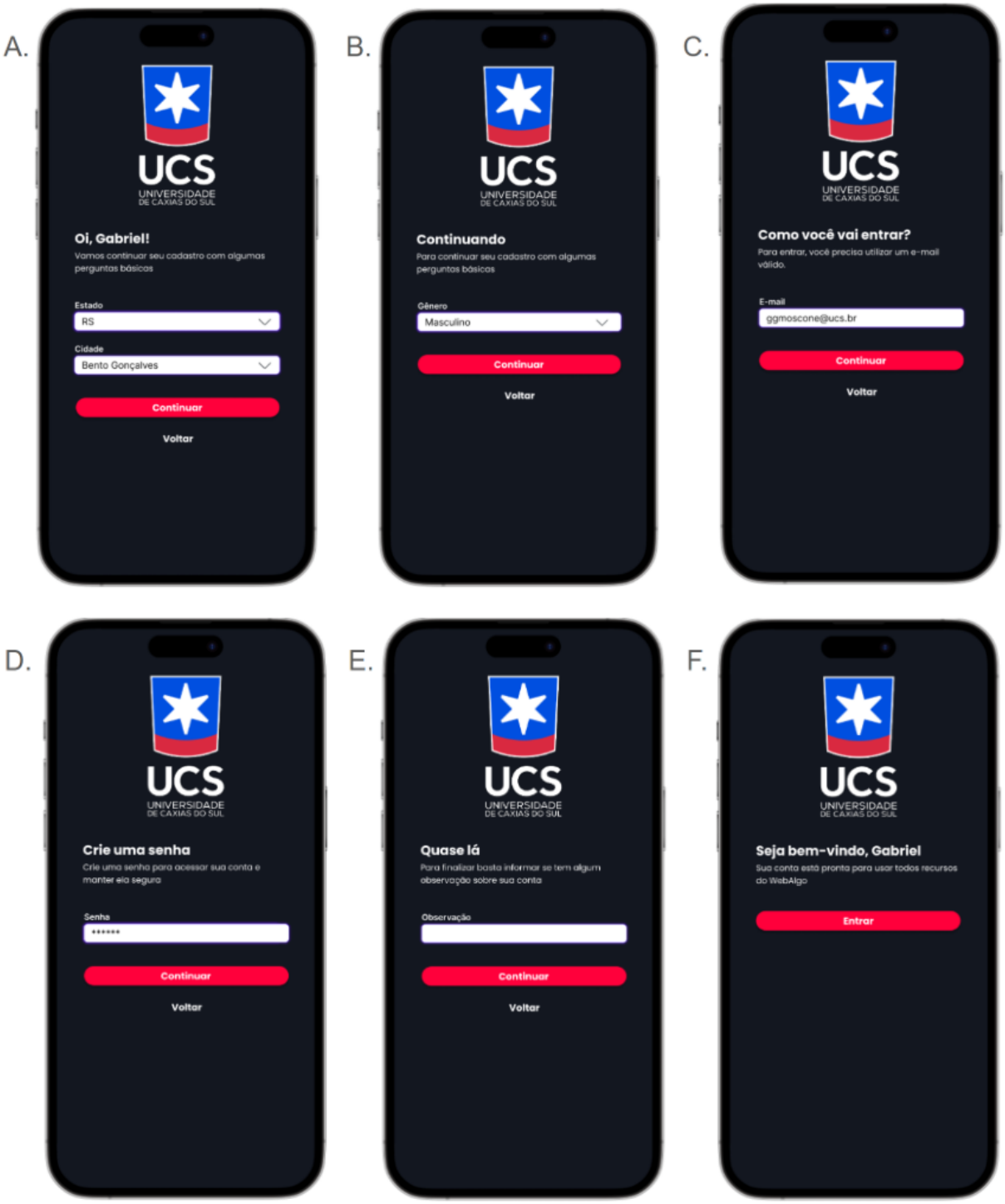


Figura 19 – Protótipos das telas de cadastro no celular

Fonte: autor

Complementando o tema do *menu*, presente na Figura 16, há a variação de tema claro para as telas, que pode ser visualizado na Figura 20, item A. Nela, pode-se analisar que todos os componentes se mantiveram, apenas invertendo as cores de claro para escuro, conforme o fundo muda. Além disso, pode-se observar no item B, a variação do tamanho da fonte para o tamanho máximo, garantindo a acessibilidade.

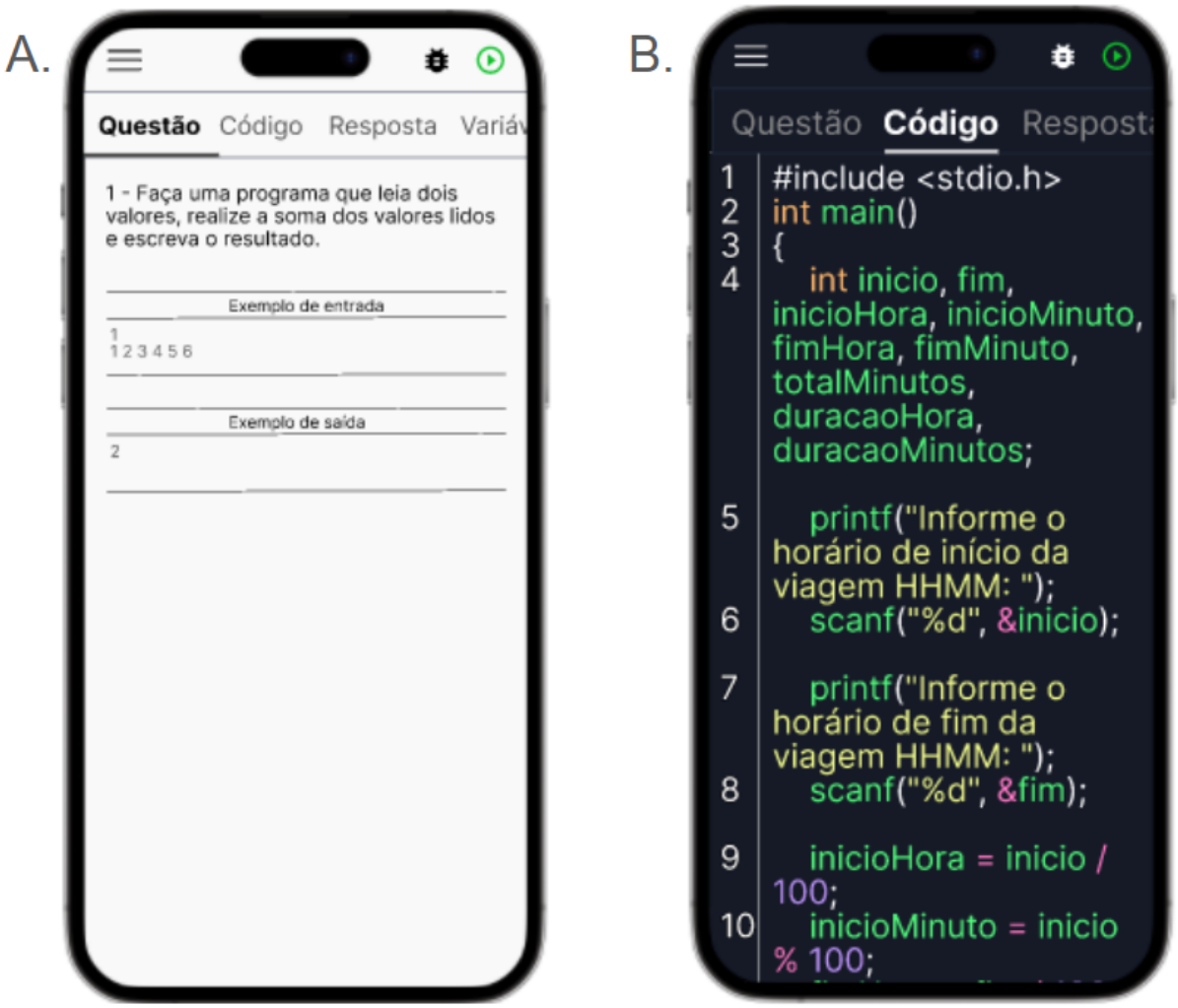


Figura 20 – Protótipo do tema claro e fonte maior no celular

Fonte: autor

Por outro lado, além das telas para os celulares, é primordial criar a mesma identidade visual para os computadores e tablets, garantindo uma consistência entre todas as resoluções. Para isso, pode-se visualizar a Figura 21, mostrando a disposição de um computador, que é possível carregar mais componentes que o celular e na Figura 22, visualiza-se a disposição em um tablet, que mantem uma consistência com o celular.

Com os protótipos prontos do sistema, é essencial criar um processo de *onboarding*, permitindo que o usuário possa se familiarizar com o produto, principalmente, por poder ser o primeiro contato com a programação. Para isso, foram criados inúmeros passos orientando e apresentando as funcionalidades da ferramenta, tendo como base a biblioteca *intro.js* que possibilita a criação de focos, textos sobrepostos e um passo a passo para o sistema (MEHRABANI, 2023). Primeiramente, pode-se observar na Figura 24, no item A que descreve uma breve introdução ao usuário, explicando o que é o sistema, Já no item B, explica-se a aba da questão, dando foco nas informações relevantes. Por fim, o item C, foca-se na aba de código, exemplificando o código escrito.

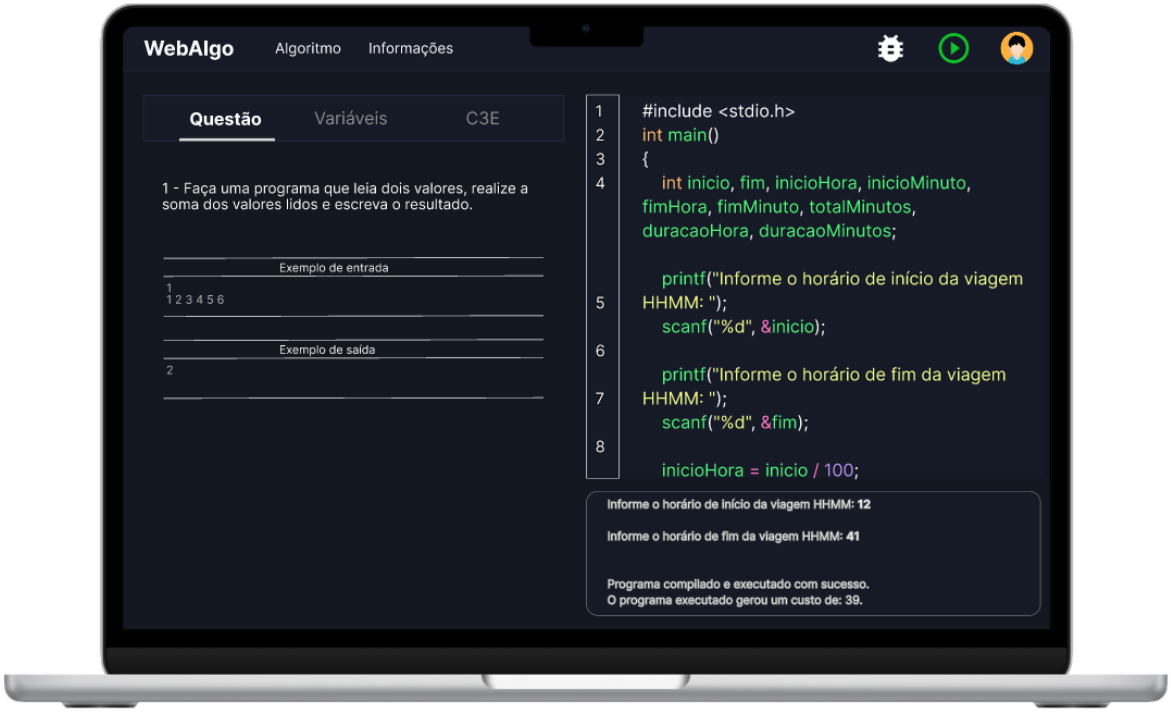


Figura 21 – Protótipo da tela principal para computador

Fonte: autor

Figura 22 – Protótipo da tela principal para tablet

Fonte: autor

Figura 23 – Protótipo da tela principal para celular deitado

Fonte: autor

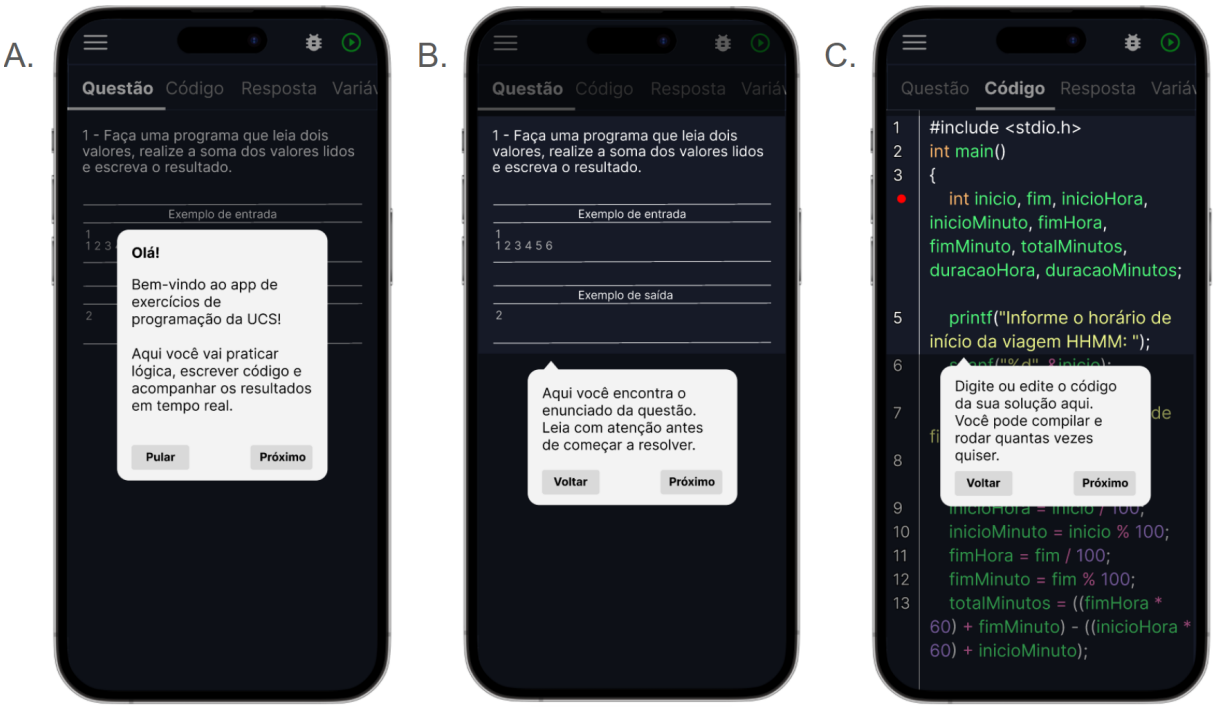


Figura 24 – Protótipo das abas com o *onboarding* - 1

Fonte: autor

Seguindo no processo de *onboarding*, o usuário seguirá navegando pelas abas, conforme a Figura 15, no qual, o próximo passo é a aba de respostas, conforme item A, que mostra todas as saídas do código. Seguindo o fluxo, no item B, será encaminhado para a aba de variáveis, listando o histórico completo. Por fim, no item C, há o C3E do código escrito.

Por fim, há os protótipos presentes na Figura 26, focando no menu, com a introdução da funcionalidade logado, no item A. Além disso, explica sobre a troca de questões, conforme o item B. Por último, no item C, há as opções de acessibilidade focadas, mostrando as funcionalidades presentes, como a opção de aumentar ou diminuir o tamanho da fonte ou escolher entre o modo claro ou escuro.

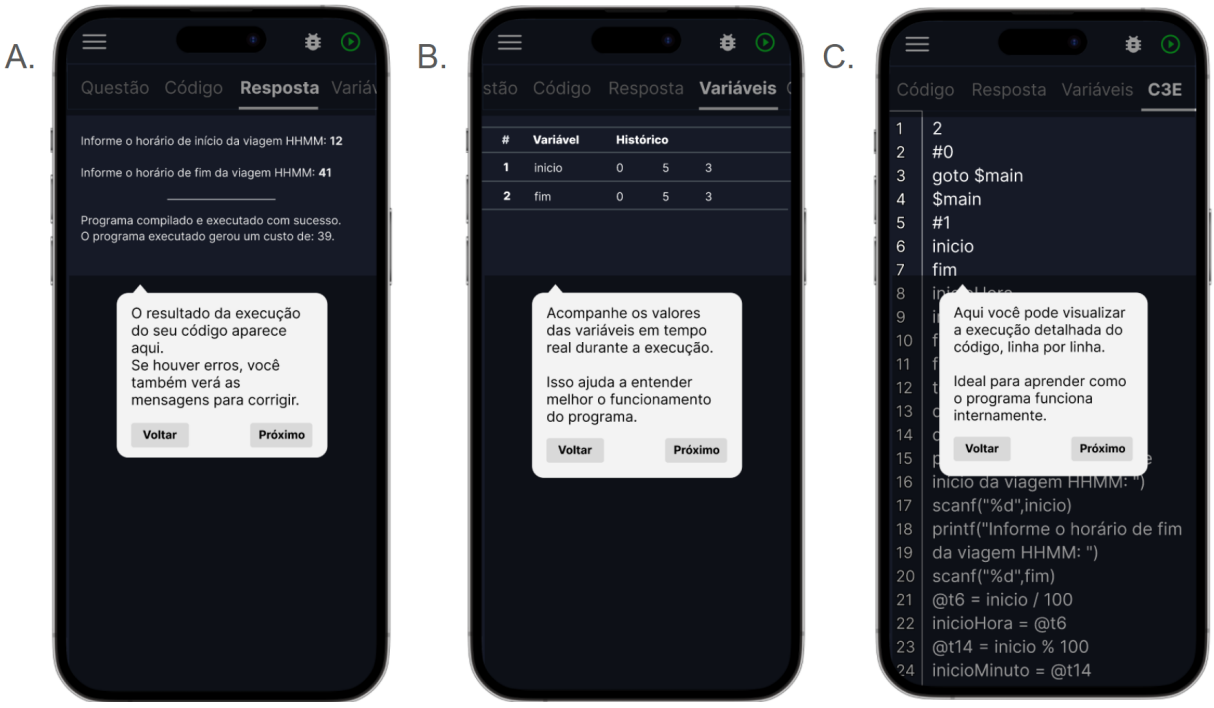


Figura 25 – Protótipo das abas com o *onboarding* - 2

Fonte: autor

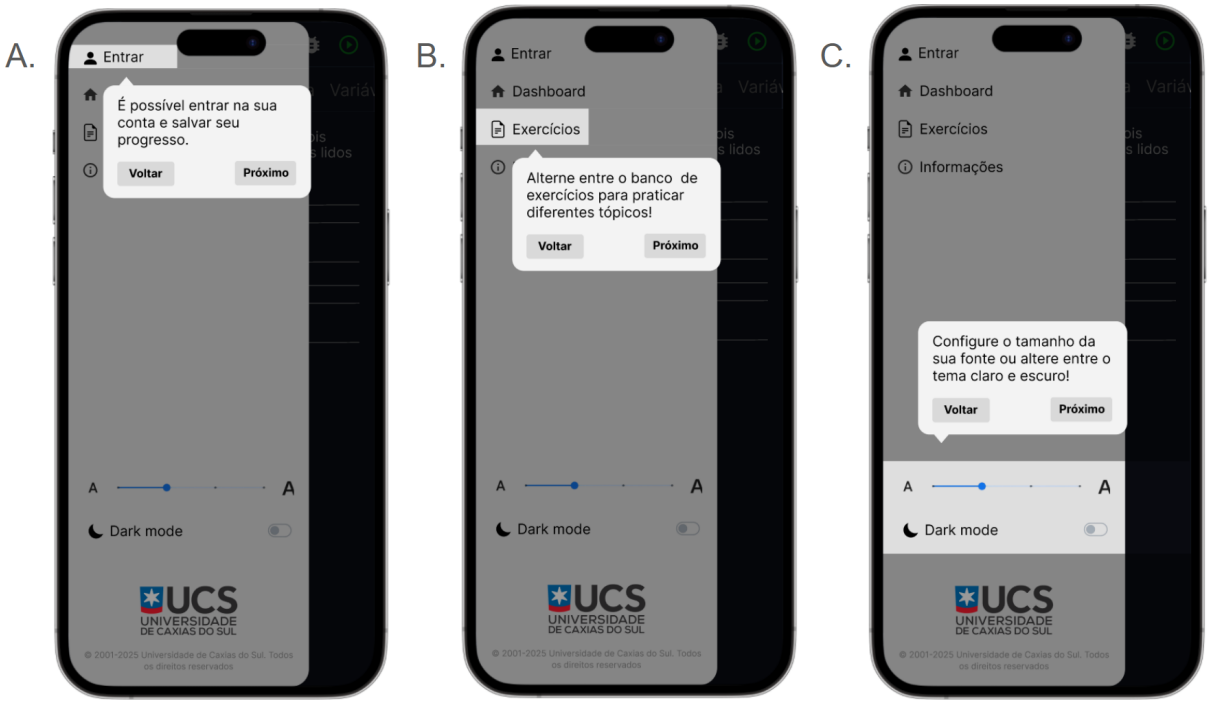


Figura 26 – Protótipo das abas e do menu lateral com o *onboarding* - 2

Fonte: autor

5 RESULTADOS

Essa etapa, busca relatar os resultados obtidos no trabalho. Primeiramente, aplicou-se o PSSUQ nas turmas de Programação I, de Caxias do Sul e Bento Gonçalves, questionando quais IDEs eram utilizadas pelos estudantes e avaliando-as, como o VS Code e o Online GDB, resultando em 50 respostas, de um total de 60 possíveis respondentes. Além das categorias medidas pelo questionário PSSUQ, foram feitas perguntas sobre IDE,buscando identificar as principais utilizadas pelos alunos e as notas que atribuem aos sistemas, como pode ser observado na Tabela 4, sendo que, quanto maior a nota melhor, tendo como máximo 7 pontos.

Tabela 4 – Aplicação do PSSUQ na turma de Programação I

	Geral	Utilidade do sistema	Qualidade da informação	Qualidade da *interface*
Online GDB	5,35	5,83	4,94	4,97
VS Code	5,43	5,56	5,04	5,94

Fonte: autor

Além disso, o questionário avaliou se os alunos utilizam o celular para programar, como pode ser visto na Figura 27, no item A, no qual 47 não usam e apenas 3 já utilizaram. Também, elencou-se quais dispositivos os estudantes possuem a sua disposição, resultando em uma divisão maior, na qual, 1 pessoa tem o Tablet, 15 alunos o celular, 26 possuem um computador pessoal e 22 utilizam o da faculdade.

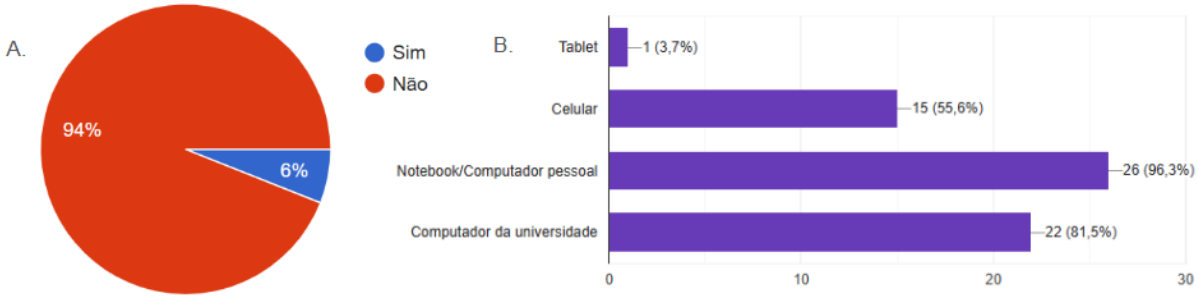


Figura 27 – Questionário sobre utilização do celular e dispositivos utilizados

Fonte: autor

Outrossim, criou-se POCs do WebAlgo para identificar a melhor alternativa, protótipos das telas utilizando o Figma que serão desenvolvidas no próximo trabalho. Além de estudar métricas do *LightHouse* e *PageSpeed Insights* para ser possível comparar com as melhorias que serão entregues.

Dessa forma, com a escolha das tecnologias e o mapeamento das mudanças, e necessidades do sistema, iniciou-se o desenvolvimento. A primeira tarefa foi adicionar o Bootstrap

5.3.6 ao WebAlgo, garantindo que será possível usufruir de todas as classes e tornar a programação mais ágil. O framework foi adicionado ao projeto através da importação via URL, incluindo arquivos CSS e *Javascript* na seção <head> do HTML.

Com o Bootstrap funcional no projeto, iniciou-se a reestilização do WebAlgo, adaptando os componentes já existentes para continuarem funcionais, porém, com a aplicação de um tema novo, que garante o constrate e responsividade necessário. Antes de começar a desenvolver os componentes, foi necessário modificar as cores do Bootstrap, garantindo que os elementos serão desenvolvidos de acordo com o protótipo desenvolvido.

O primeiro componente criado foi a *navbar*, iniciando pela resolução dos celulares - indo de acordo com a abordagem do *mobile first* - elemento presente no topo da página e utilizado para navegação entre as páginas. Para isso utilizou-se o próprio componente do Bootstrap, o *navbar*, que já é criado com os estilos prontos e responsivo, apenas adicionando os itens necessários, como os botões para executar o código e a botão para abrir o menu lateral no celular. Assim, ao clicar nesse botão será aberto o menu, que foi criado utilizando o *offcanvas* do Bootstrap, que cria um elemento que se sobrepõe na tela com uma animação lateral. Dessa forma, com o menu criado, foi adicionado todos os redirecionamentos necessários e ações de acessibilidade, como pode ser observado na Figura 28, sendo possível configurar o tamanho da fonte, aumentando-a ou diminuindo e até mesmo escolher entre os temas claro ou escuro.

Dessa forma, ao clicar na opção "Exercícios", conforme a Figura 28, foram criados os *modais*, elementos que se sobrepõem a página, conforme pode ser observado na Figura 29, para destacar um componente, utilizando o elemento *modal* do próprio Bootstrap. Portanto, o usuário poderá escolher o exercício que quer resolver sem que tenha que trocar de página ou contexto totalmente.

Voltando para a página inicial, para ser possível organizar todos os conteúdos necessários, foi divididos os elementos no celular em abas, garantindo que o usuário pode selecionar qual deseja ter foco. Para desenvolver essa funcionalidade foi utilizado o elemento *tab* do Bootstrap, permitindo que o sistema mostre todas as opções, como: Questão, Variáveis, C3E, Classificação e Código. Dessa forma, a configuração desse elemento garante que de acordo com o tipo selecionado, seja montado uma tela diferente, permitindo mais agilidade e praticidade para o usuário.

Com essa separação, desenvolveu-se uma aba por vez, iniciando pela aba questão, nela foi codificado alguns elementos de textos, para descrever o propósito do exercício e um pequeno formulário para carregar ou salvar suas soluções. Posteriormente, criou-se a aba de variáveis, a qual criou-se uma tabela - utilizando a classe *table* do Bootstrap - que será montada após o código ser executado, listando os últimos valores das variáveis e permitindo abrir um *modal*, que listará o histórico completo de cada variável. A terceira aba criada foi a C3E, que foi contruída apenas com elementos textuais, listando o código C3E gerado a partir do código escrito. Prosseguindo, iniciou-se a aba Classificação, similar a aba de variáveis, com uma tabela que

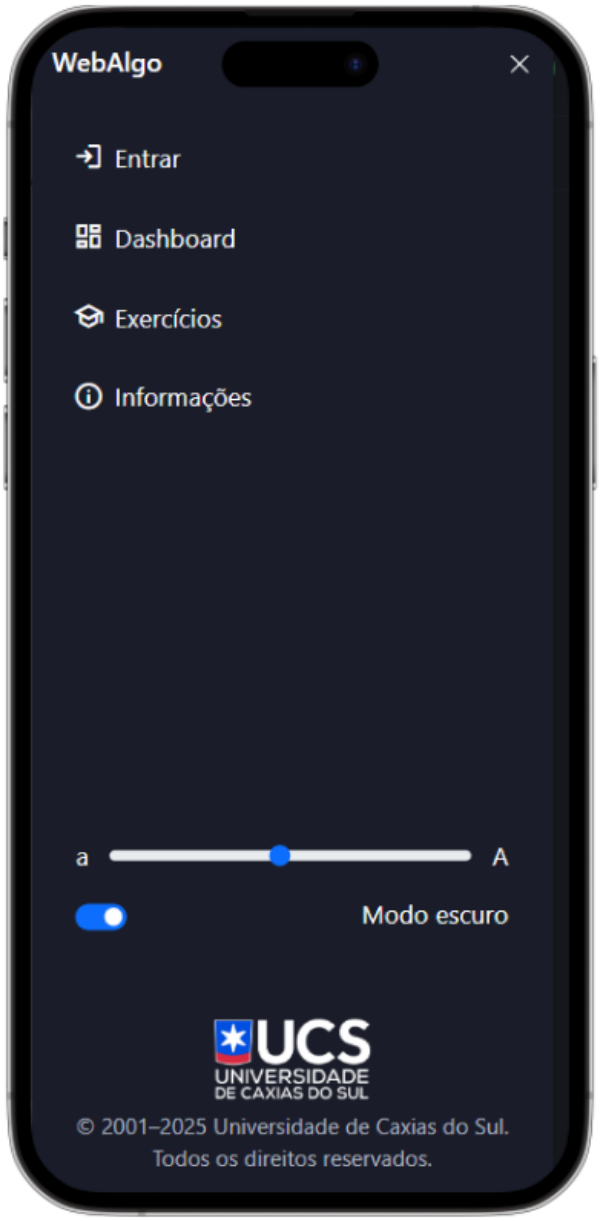


Figura 28 – Menu lateral no WebAlgo novo

Fonte: autor

listará o ranking completo do exercício selecionado, com informações como: posição e nome. Por último, exclusivamente para a resolução do celular, foi criado a aba do código, que será constituída pelo editor de texto, pela saída e pelo campo de entrada.

Com a estrutura do sistema já consolidada, iniciaram-se as adaptações para diferentes resoluções de tela, especialmente para computadores. Inicialmente, foi ajustado o *menu*, de modo que o menu lateral fosse ocultado em determinadas resoluções. Para isso, utilizaram-se propriedades do CSS, como *display: none*, em conjunto com a diretiva *media query*, permitindo a personalização da interface conforme o tamanho do dispositivo. Além disso, ao invés da divisão completa da tela em abas, dividiu-se ela em duas colunas, na qual a parte da esquerda ficou com uma versão menor das abas, enquanto a direita ficou com o editor de texto, as saídas do

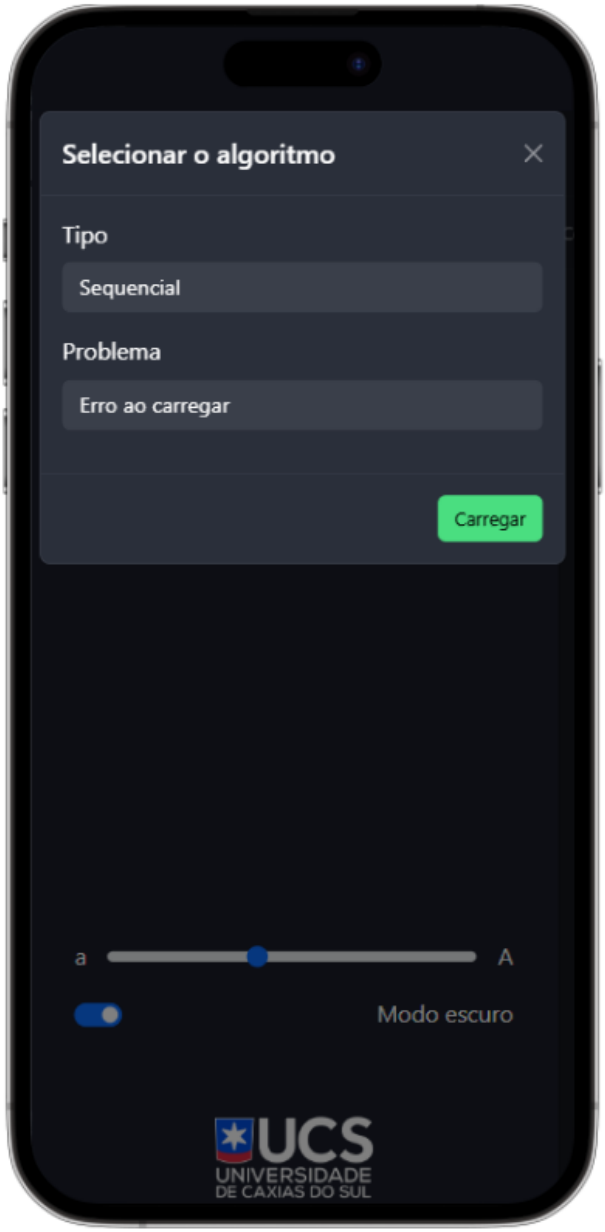


Figura 29 – *Modal* para selecionar o exercício no WebAlgo novo

Fonte: autor

código e o campo de entrada.

Posteriormente, no *menu* para a versão do computador, foi criado um elemento de *popup* com as opções faltantes, que existem no menu lateral do celular, como: o botão para sair e as configurações de acessibilidade - permitindo alterar o tamanho da fonte e trocar de tema. Dessa forma, é aproveitado todo espaço disponível, sem prejudicar a experiência do usuário e mantendo uma usabilidade adequada.

Partindo para os componente iterativos, como o editor de texto, as saídas e os históricos, criou-se uma lógica para movimentar esses elementos entre as páginas, visando manter funcional a lógica já existente. Para isso, utilizou-se o Javascript e suas funções como *getEle-*

mentById e *querySelector* para selecionar os elementos no HTML, e de acordo com a resolução do dispositivo mover ele com a função *appendChild* para a posição adequada.

Com a página principal adaptada e responsiva, inicou-se as mudanças relacionadas a acessibilidade, para garantir o aumento das métricas de usabilidade. Para isso, além do contraste adequado dos componentes, a disposição e o *layout* responsivo, desenvolveu-se duas funções principais. A primeira, foi um controle do tamanho da fonte, possibilitando que o usuário através de um elemento *input* do HTML do tipo *range*, possa aumentar ou diminuir o tamanho da fonte de toda a página, facilitando a leitura dos textos. Por fim, codificou-se um *input* do tipo *checkbox*, para alterar entre o tema escuro e o tema claro, permitindo o usuário personalizar sua IDE como preferir, como pode ser observado na Figura 30 o tema claro no celular. Para essa alteração entre os dois temas, utilizou-se uma classe global do CSS para definir qual seria utilizado, e a partir dela se modifica as variáveis que contém as cores da página, facilitando a troca e a manutenção do padrão da página. Além do mais, foi necessário ter duas imagens do logo da UCS no projeto, visto que o fundo precisava ter um contraste adequado com as escritas do mesmo.

Com o tema definido e os componentes customizados, partiu-se para o desenvolvimento das telas de autenticação necessárias, iniciando pela tela de *login*. Vale ressaltar, que aproveitou-se para todas as telas as mesmas classes, temas e componentes do Bootstrap e os personalizados. Dessa forma, a tela foi montada com os seguintes elementos: o logo da UCS, dois campos de texto, um para informar o usuário e outro para a senha - com as opções de ocultar ou mostrar a senha - e os *links* para o esqueceu sua senha e para o cadastro. Ademais, criou-se os elementos para as tratativas de erros dos serviços utilizados, estilizando-os com o CSS e criando-os no DOM com o Javascript. Posteriomente, desenvolveu-se a tela do cadastro, que foi dividida em passos para diminuir a carga cognitiva do usuário, para isso, utilizou-se o CSS e o Javascript para esconder e mostrar os passos, além de validar todos os campos a cada passo. Dessa forma, dividiu-se nos seguintes passos: a primeira solicita o nome completo e o cargo; a segunda, o estado e a cidade; a terceira, o sexo; a quarta, o e-mail e o nome de usuário; e a quinta e última etapa, a criação e a confirmação da senha.

Portanto, com o usuário preenchendo adequadamente todos os campos, é direcionado para uma tela de sucesso e depois pode usar as mesmas informações de autenticação no *login* para se identificar no sistema. Por último, criou-se a tela para alterar a senha, na qual, o usuário informa os seguintes campos: usuário, senha atual, nova senha e repetir a senha. Vale ressaltar, que utilizou-se os mesmo componentes das outras telas, como o logo, os campos de textos e os botões estilizados.

Sendo assim, com as principais funcionalidades do sistema ajustadas e criadas - agora com um *layout* responsivo e foco na usabilidade - foi possível rodar novamente as ferramentas de análise do *PageSpeed Insights* e o *LightHouse*, para entender como as métricas de acessibilidade, desempenho, boas práticas e SEO estão com as mudanças do WebAlgo novo. Assim,

Figura 30 – Tela principal com tema branco no celular do WebAlgo novo

Fonte: autor

observando a Tabela 5 se percebe que algumas métricas pioraram, como o desempenho, sendo diretamente relacionado as novas importações do Bootstrap e *intro.js*.

Dessa forma, através desses relatórios, buscou-se melhorar todas as métricas, iniciando pelo desempenho. Para isso, otimizou-se o carregamento dos *scripts* da página, criando uma política de carregamento tardio, para não bloquear o primeiro acesso do usuário. Um exemplo, é a configuração *defer*, que permite que os arquivos sejam carregados em paralelo ao processamento principal, além disso, adaptou-se o *layout* para evitar que ele seja trocado constatantemente, permitindo ter um corpo pré estruturado. Ademais, foi reduzida a qualidade e tamanho das imagens, sem afetar o usuário, para diminuir o tamanho total dos arquivos baixados.

Outrossim, focou-se na acessibilidade, na qual identificou-se pontos com um constraste

Tabela 5 – Notas de desempenho medidas no *PageSpeed Insights* e *LightHouse* no WebAlgo *web* novo, antes das melhorias

		Desempenho	Acessibilidade	Práticas recomendadas	SEO
PageSpeed Insights	Computador	70	86	91	86
	Celular	72	85	92	86
LightHouse	Computador	64	89	91	90
	Celular	75	90	90	90

Fonte: autor

Tabela 6 – Notas de desempenho medidas no *PageSpeed Insights* e *LightHouse* no WebAlgo *web* novo, depois das melhorias

		Desempenho	Acessibilidade	Práticas recomendadas	SEO
PageSpeed Insights	Computador	97	100	96	100
	Celular	98	100	96	100
LightHouse	Computador	99	100	96	100
	Celular	99	100	95	100

Fonte: autor

menor que o ideal e elementos sem as configurações de acessibilidade adequadas. Primeiramente, alterou-se as cores da página, permitindo que os textos e o fundo, tenham o contraste adequado para todos os leitores. Posteriormente, verificou-se todas *tags* de acessibilidade, no HTML, como o *aria-label*, se todos os campos de textos tem uma *label* relacionada, o tamanho mínimo dos elementos.

Posteriormente, focou-se nas práticas recomendadas, removendo *logs* desnecessários e de erros do *console* do navegador, utilizando de protocolos inseguros, como o *Hypertext Transfer Protocol* (HTTP). Dessa forma, garante-se uma nota acima de 90 nas práticas.

Finalmente, cuidou-se do SEO, sendo necessário configurar títulos e descrições para as páginas no elemento *meta* do HTML - elemento que fornece metadados sobre o sistema. Para isso, criou-se configurações para a linguagem, descrição, título e até para a responsividade da página.

Portanto, com todas essas melhorias, rodou novamente os relatórios, para observar a evolução do sistemas, como pode ser observado na Tabela 6. Analisando os números, percebe-se que as métricas estam quase beirando o 100, ou seja, com as melhores práticas do mercado.

Dessa forma, com todas as métricas computadas, pode-se observar a Figura 31, verificando a diferença das métricas de acessibilidade, desempenho, práticas recomendadas e SEO das diferentes versões do WebAlgo visualmente.

Com as métricas otimizadas, para aprimorar a usabilidade, principalmente, para novo usuários, criou-se um tuor para as principais funções do WebAlgo. Para isso, utilizou-se a biblioteca *intro.js* que facilita o desenvolvimento, permitindo criar mensagens customizadas e dar

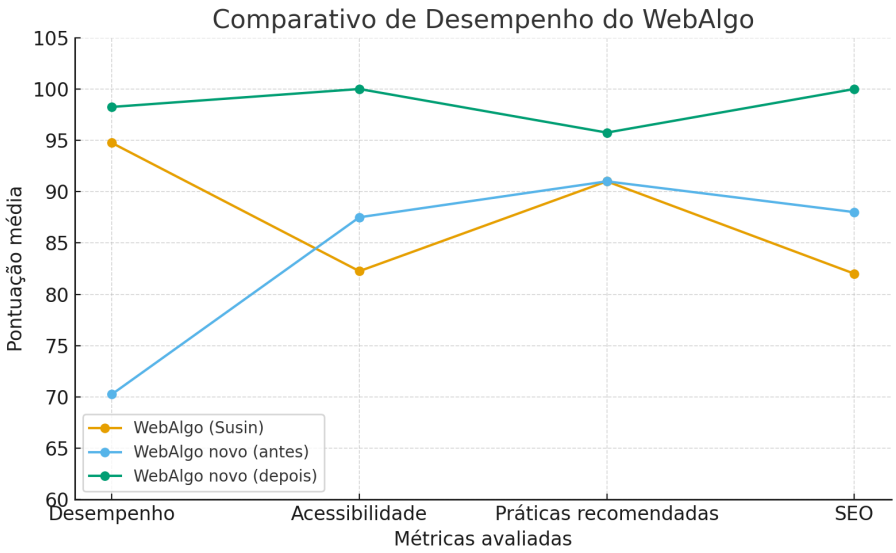


Figura 31 – Gráfico das métricas de acessibilidade e desempenho do WebAlgo

Fonte: autor

foco em áreas específicas do sistema, como é visto na Figura 32. Portanto, basta informar a mensagem que deve ser mostrada a cada passo - podendo utilizar o HTML para customizar e identificando o elemento de foco através das funções do JavaScript, como o *getElementById*, conforme pode ser observado no exemplo Algoritmo 1

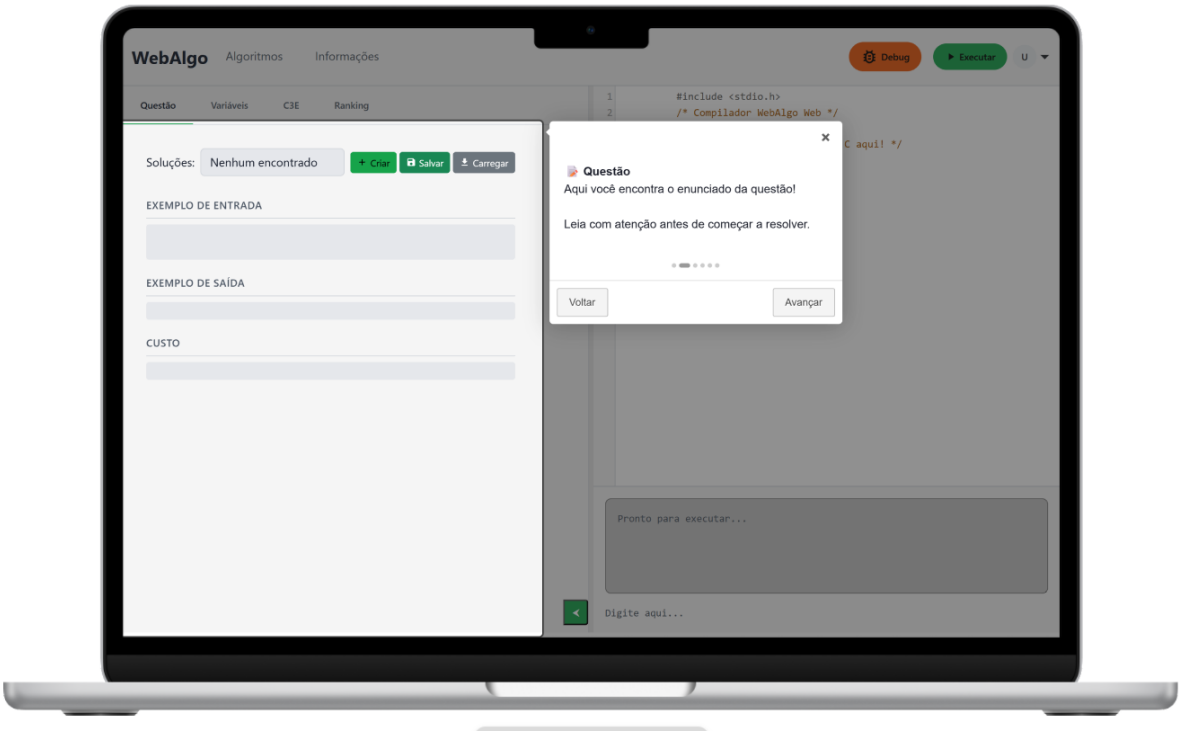


Figura 32 – Exemplo de passo do *tuor* no computador

Fonte: autor

Algoritmo 1 – Exemplo de uso do *intro.js*

```
introJs()
    .setOptions({
      steps: [
        {
          element: '.header',
          intro:
            '<strong>Ola!</strong><br>Bem-vindo ao app de exercicios de
               programa o da UCS!<br/><br/>Aqui voc vai praticar
               l gica, escrever c digo e acompanhar os resultados em
               tempo real.',
          position: 'bottom',
        }
     ]
    });
```

Com o projeto do WebAlgo se tornando online e permitindo o acesso, para diferentes dispositivos, é importante quantificar e segregar os usuários em tipos. Dessa forma, para isso, criou-se uma conta no *Google Analytics*, ferramenta de análise de dados de acesso e utilização dos sistemas e com a conta criada, foi inserido o *script* do mesmo para iniciar a análise. Com isso, será possível entender o comportamento dos utilizadores, os dispositivos mais utilizados e as páginas acessadas, identificando pontos de melhorias e o público alvo.

Com a evolução constante da plataforma, elencou-se uma nova melhoria, otimizando a usabilidade, criando a opção de permitir o usuario esconder a aba da esquerda, em resoluções maiores, dando maior foco para o editor de texto. Para isso, criou-se um componente de arrastar no centro da tela, possibilitando que o usuário diminua a aba da direita, até no máximo 30% da tela, conforme pode ser visualizado na Figura 33. Essa largura mínima foi definida para não permitir que o usuário diminua demasiadamente a largura da coluna, tornando difícil a leitura da página.

Para otimizar ainda mais, foi criado um botão que esconde totalmente a aba da esquerda, como pode ser visto na Figura 33, no canto inferior da esquerda, permitindo que o código ocupe toda a tela, como pode ser visualizado na Figura 34. Vale ressaltar, que esse botão apenas é mostrado para resoluções maiores, que possuem duas abas verticais.

Dessa forma, utilizando o mesmo código testado em outros sistemas, como o Online GDB, utilizou ele no WebAlgo novo, como pode ser visualizado na Figura 35 na versão do computador e na Figura 36 no celular. Dessa forma, sendo possível visualizar a utilização real do sistema, e como o *layout* e o código se comportam.

Ademais, com o projeto consolidado, criou-se a documentação do sistema desenvolvido, dentro do repositório, disponível em GitHub - GabrielMoscone/web-algo, explicando: a organização do projeto, as tecnologias utilizadas e como rodar o sistema localmente.

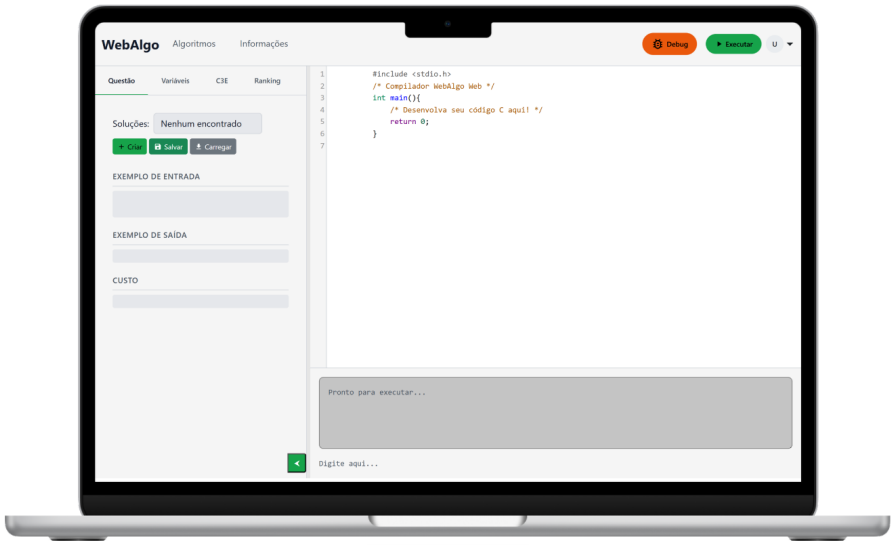


Figura 33 – Divisão das abas customizadas no computador

Fonte: autor

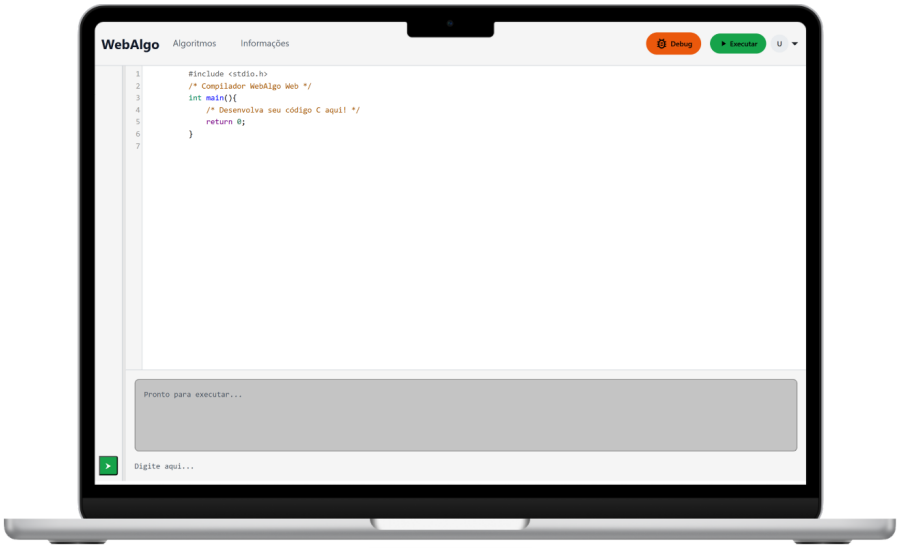


Figura 34 – Aba da esquerda escondida no computador

Fonte: autor

5.1 PESQUISAS

No entanto, mesmo com a proposta desenvolvida com integridade, é de suma importância que o sistema seja validado com usuários reais, podendo obter *feedbacks* e identificar melhorias para serem realizadas e garantir que o WebAlgo seja um programa apreciado pelos estudantes. Para isso, aplicou-se o PSSUQ, novamente, em três turmas diferentes de Programação, somando um total de 69 possíveis participantes, no qual, obtive-se 34 respostas. Ademais, adicinou-se três perguntas, questionando qual IDE é a mais utilizada para a disciplina de programação, se utiliza o celular para programar e por fim, quais equipamentos tem a disponibilidade fora da universidade.

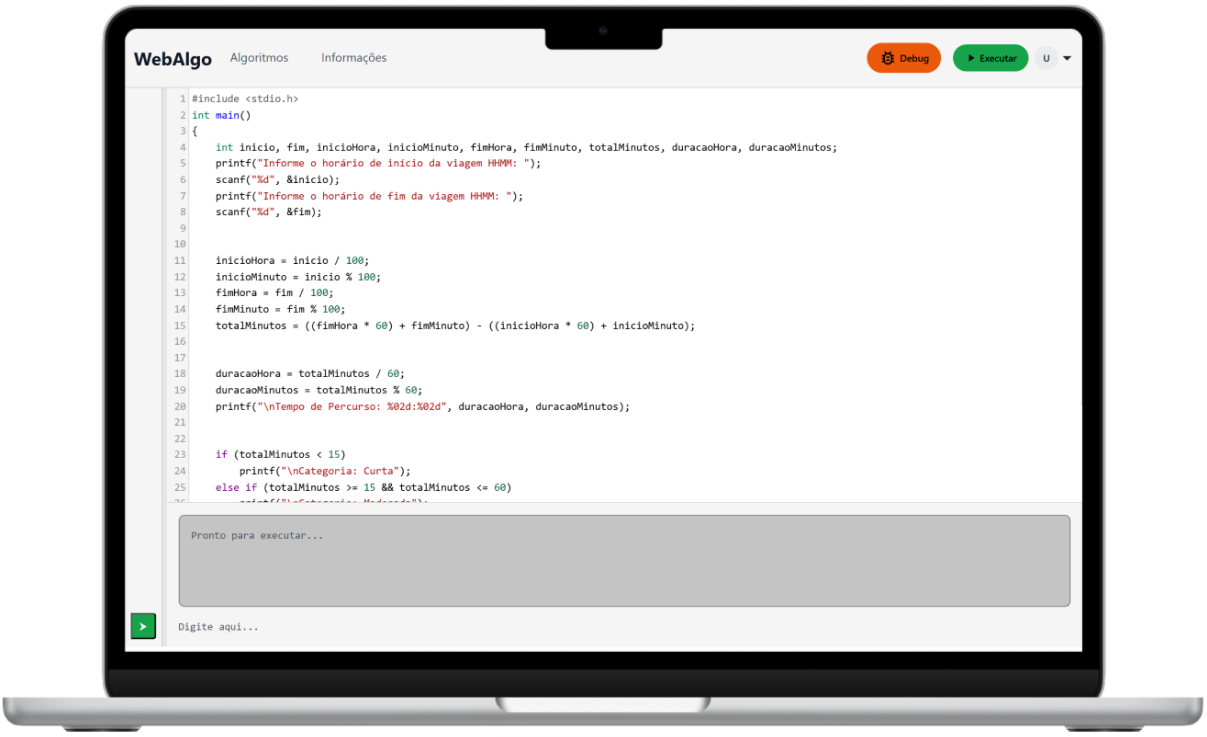


Figura 35 – Testando código no WebAlgo *web* novo no computador

Fonte: autor

Portanto, com a contabilização dos resultados, pode-se calcular o resultado das notas do PSSUQ, conforme pode ser observado na Tabela 7 e podendo compará-los com os resultados obtidos posteriormente de ferramentas já presentes no mercado, como o *Online GDB* e o VS Code. Dessa forma, pode-se elencar que o sistema teve um resultado muito positivo, obtendo médias próximas a nota 6 e tendo a nota geral como 5,89 - sendo 7 a nota máxima - e resultando em notas superiores as notas desses produtos em todas as métricas avalidas. Ao analisar cada métrica individualmente, percebe-se a maior foi a Utilidade do sistema, com 5,97, indicando que os usuários conseguiram e localizaram as funcionalidades desenvolvidas, mantendo a consistência. Já a qualidade da *interface* e da informação, também apresentaram notas altas, sendo elas 5,78 e 5,90, respectivamente, indicando que a interatividade e o visual foram prototipados de forma adequada.

Tabela 7 – Aplicação do PSSUQ no WebAlgo novo

	Geral	Utilidade do sistema	Qualidade da informação	Qualidade da *interface*
WebAlgo	5,89	5,97	5,78	5,90

Fonte: autor

Outrossim, vale destacar os resultados das três perguntas adicionais feitas, conforme pode ser visualizado na Figura 37, com porcentagens muito semelhantes ao primeiro teste aplicado, observado na Figura 27, com uma taxa de 97,1% dos participantes não utilizando o celular para programar, enquanto no primeiro questionário esse número foi 94%. Por outro lado, so-

Figura 36 – Testando código no WebAlgo *web* novo no celular

Fonte: autor

bre os dispositivos, esse segundo questionário focou-se em identificar os dispositivos utilizados fora da universidade, resultando em 34 alunos com notebook ou computadores, 10 com celular e apenas 2 com tablets. E por fim, identificou-se as principais IDEs utilizadas, sendo elencado o VS Code, o Online GDB e o CodeBlocks.

No entanto, ao mesmo tempo que foram realizados esses questionários nas turmas, foi disponibilizado o sistema do WebAlgo novo para ser utilizado, permitindo testar o sistema, obtendo métricas da utilização e até mesmo elencar melhorias e problemas. Como, por exemplo, em uma aplicação que tinha 22 alunos presentes, foi possível visualizar a quantidade de usuários simultâneos através do Google Analytics, conforme a Figura 38, item A. Além da quantidade de usuários ativos, foi possível identificar as páginas, acessadas, na qual obteve-se 15 visualizações

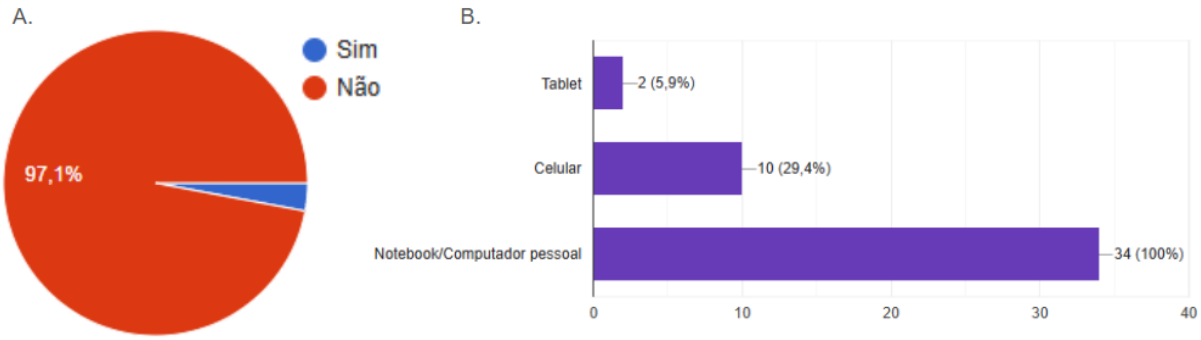


Figura 37 – Questionário sobre utilização do celular e dispositivos utilizados fora da universidade

Fonte: autor

na página principal e 5 na página de autenticação.

Figura 38 – Métricas do Google Analytics durante a utilização com 22 alunos

Fonte: autor

Além desses valores, uma ferramenta como o Google Analytics, consegue identificar também os principais dispositivos utilizados e até navegadores, permitindo que seja categorizado e até mesmo priorizado os mais utilizados, conforme se observa na Figura 38, no item B, no qual foi utilizado por 92,2% por computadores e o restante em dispositivos móveis. Ademais, observando a Figura 39, percebe-se a utilização do sistema em um período mais amplo - 03 de outubro de 2025 até 30 de outubro de 2025 - com os navegadores mais utilizados sendo o Chrome e o Edge, e sendo dividido em diversas resoluções, como 1600x900, 1920x1080 e 1536x864.

Durante os questionários, também foram identificados alguns problemas pelos usuários, como por exemplo, para os celulares, quando o código escrito tinha muitas linhas na IDE, mais de 25 linhas no dispositivo identificado, a página quebrava a estrutura, escondendo o menu do topo e ficando visível apenas o editor e o elemento de saída, tornando impossível rodar o código e acessar o menu lateral. Para isso, foi corrigido o problema através da aplicação de regras do

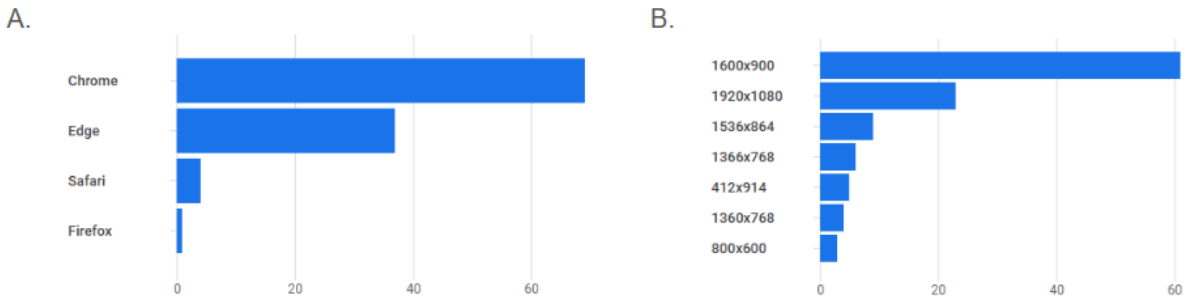


Figura 39 – Métricas do Google Analytics relacionado aos dispositivos utilizados

CSS para limitar o tamanho máximo do editor de texto, garantindo que todos os elementos tem o tamanho mínimo viável para seu funcionamento, além de garantir a possibilidade de rolar a tela, quando necessário.

6 CONCLUSÃO

Este trabalho obteve como resultado principal a disponibilidade em celulares de um site existe para ensino de programação, que fosse acessível, responsivo e de fácil utilização, especialmente voltado para os estudantes. Primeiramente, vale ressaltar que o trabalho foi desenvolvido adequadamente, sendo possível atender todos os pontos elencados na proposta de solução, como aumentar as métricas de acessibilidade - atigindo 100 pontos, medidos através do *PageSpeed Insights* e o *LightHouse* - criando temas, funcionalidades novas e permitindo que dispositivos distintos usem o sistema sem problemas. Além disso, utilizou-se as tecnologias e ferramentas indicadas, como a introdução do Bootstrap, o *Google Analytics* e até mesmo o *intro.js*.

Outrossim, a configuração do *onboarding*, contribui para otimizar e introduzir o usuário no seu primeiro acesso, oferecendo um passo a passo intuitivo das principais funcionalidades do sistema, tornando a curva de aprendizado mais tranquila. Essas melhorias, refletiram diretamente na usabilidade e experiência do usuário, fato observado através das pesquisas, como o PSSUQ - com uma média geral de 5,89, utilidade de 5,97, qualidade da informação com 5,78 e qualidade da *interface* com 5,90, de um total de 7. Percebe-se que o sistema alcançou os objetivos de usabilidade, acessibilidade e responsividade propostos, com uma satisfação positiva dos usuários.

Portanto, percebe-se um sistema entregue com qualidade e todas as funcionalidades propostas, possuindo as telas propostas, com a possibilidade de alterar o tema, o tamanho da fonte e adaptar a tela conforme o usuário preferir, integrando as tecnologias propostas - HTML, CSS, Javascript e a adição do Bootstrap.

6.1 TRABALHOS FUTUROS

Tendo em vista o desenvolvimento do trabalho e os resultados das pesquisas supracitadas, elenca-se pontos de evolução que podem ser realizados em trabalhos futuros, sendo eles:

- Visualização de estrutura de dados, através de gráficos e figuras visuais, permitindo que os professores e os usuários, visualizem algoritmos como o de ordenação, busca e até a estrutura de árvores.
- Indentação do código automática, facilitando a leitura do código e mantendo um padrão no sistema
- Integração com Inteligência Artificial (IA), com explicações e sugestões do código escrito

- Configuração de atalhos do teclado, como um atalho para executar o código, comentar uma linha e até mesmo marcar a linha do *debug*

REFERÊNCIAS

AKRAM, M.; SULAIMAN, R. B. A systematic literature review to determine the web accessibility issues in saudi arabian university and government websites for disable people. **International Journal of Advanced Computer Science and Applications**, Science and Information (SAI) Organization Limited, v. 8, n. 6, 2017.

ALMONAIES, A. A. *et al.* A framework for migrating web applications to web services. In: SPRINGER. **Web Engineering: 13th International Conference, ICWE 2013, Aalborg, Denmark, July 8-12, 2013. Proceedings 13**. [S.l.], 2013. p. 384–399.

AMARAL, É. *et al.* Algo+ uma ferramenta para o apoio ao ensino de algoritmos e programaçao para alunos iniciantes. In: **Brazilian Symposium on Computers in Education (Simpósio Brasileiro de Informática na Educação-SBIE)**. [S.l.: s.n.], 2017. v. 28, n. 1, p. 1677.

AQUILES, A.; FERREIRA, R. **Controlando versões com Git e GitHub**. [S.l.]: Casa do Código, 2014.

ARA, J.; SIK-LANYI, C.; KELEMEN, A. Accessibility engineering in web evaluation process: a systematic literature review. **Universal Access in the Information Society**, Springer, v. 23, n. 2, p. 653–686, 2024.

BADER, W. I.; HAMMOURI, A. I. Responsive web design techniques. **International Journal of Computer Applications**, Foundation of Computer Science, v. 150, n. 2, p. 18–27, 2016.

BARBOSA, C. *et al.* Redesenho e prototipagem de um repositório de recursos educacionais para metaverso. In: SBC. **Simpósio Brasileiro de Informática na Educação (SBIE)**. [S.l.], 2024. p. 1012–1027.

BARROZO, G. C.; VINHAS, H. M.; REIS, J. C. de S. Refatoraçao: Aperfeiçoando um código existente. **RE3C-Revista Eletrônica Científica de Ciência da Computação**, v. 7, n. 1, 2012.

BAUMER, D. *et al.* User interface prototyping-concepts, tools, and experience. In: IEEE. **Proceedings of IEEE 18th International Conference on Software Engineering**. [S.l.], 1996. p. 532–541.

BERNACKI, J. *et al.* Responsive web design: testing usability of mobile web applications. In: SPRINGER. **Intelligent Information and Database Systems: 8th Asian Conference, ACIIDS 2016, Da Nang, Vietnam, March 14–16, 2016, Proceedings, Part I 8**. [S.l.], 2016. p. 257–269.

BLISCHAK, J. D.; DAVENPORT, E. R.; WILSON, G. A quick introduction to version control with git and github. **PLoS computational biology**, Public Library of Science San Francisco, CA USA, v. 12, n. 1, p. e1004668, 2016.

BOTELLA, P. *et al.* Iso/iec 9126 in practice: what do we need to know. In: DATA PROCESSING ORGANIZATION SRL, VIA VALENTINO MAZZOLA ROME, ITALY. **Software Measurement European Forum**. [S.l.], 2004. v. 2004.

CAJAS, V. *et al.* Challenges of migrating legacies web to mobile: a systematic literature review. **IEEE Latin America Transactions**, IEEE, v. 18, n. 05, p. 861–873, 2020.

____. Migrating legacy web applications. **Cluster Computing**, Springer, v. 24, p. 1033–1049, 2021.

____. An approach for migrating legacy applications to mobile interfaces. In: SPRINGER. **New Knowledge in Information Systems and Technologies: Volume 1**. [S.l.], 2019. p. 916–927.

CHALLAPALLI, S. S. N. *et al.* Web development and performance comparison of web development technologies in node. js and python. In: IEEE. **2021 International Conference on Technological Advancements and Innovations (ICTAI)**. [S.l.], 2021. p. 303–307.

CONFORTO, D.; SANTAROSA, L. M. C. Acessibilidade à web: Internet para todos. **Informática na educação: teoria & prática. Porto Alegre. Vol. 5, n. 2 (nov. 2002), p. 87-102**, 2002.

CORRÊA, P. V.; RIBEIRO, D. F. Acessibilidade e usabilidade na web: Recursos utilizados para inclusão de usuários. **Revista Interface Tecnológica**, v. 12, n. 1, p. 7–17, 2015.

COSENTINO, V.; IZQUIERDO, J. L. C.; CABOT, J. A systematic mapping study of software development with github. **Ieee access**, IEEE, v. 5, p. 7173–7192, 2017.

CRUZ, A. K. B. S. da *et al.* Utilizaçao da plataforma beecrowd de maratona de programaçao como estratégia para o ensino de algoritmos. In: SBC. **Simpósio Brasileiro de Jogos e Entretenimento Digital (SBGames)**. [S.l.], 2022. p. 754–764.

DEFICIÊNCIA, E. d. P. com. Lei nº 13.146, de 6 de julho de 2015. **Institui a Lei Brasileira de Inclusão da Pessoa com Deficiência**, 2015.

DEKEMELE, K.; CHEVALIER, A.; LOCCUFIER, M. Odysc: A responsive educational web app for dynamics and control. **IFAC-PapersOnLine**, Elsevier, v. 51, n. 4, p. 310–315, 2018.

DORNELES, R. V.; JR, D. P.; ADAMI, A. G. Algoweb: a web-based environment for learning introductory programming. In: IEEE. **2010 10th IEEE International Conference on Advanced Learning Technologies**. [S.l.], 2010. p. 83–85.

DRISS, L. M. *et al.* Version [1.1. 0]-[dldiagnosis: A mobile and web application for diseases classification using deep learning]. **SoftwareX**, Elsevier, v. 26, p. 101745, 2024.

ECONOMIC, U. N. D. of; AFFAIRS, S. **Disability and Development Report 2024: Accelerating the realization of the Sustainable Development Goals by, for and with persons with disabilities**. New York: United Nations, 2024. Full report (unedited version) available at UN DESA website. Disponível em: <https://social.desa.un.org/sites/default/files/publications/2024-06/Final-UN-DDR-2024-Executive%20Summary.pdf>.

ESTATíSTICA, I. I. B. de Geografia e. **Pesquisa Nacional por Amostra de Domicílios Contínua - 2023**. 2023. Acesso em: 2025-04-20. Disponível em: <https://www.ibge.gov.br/estatisticas/sociais/trabalho/9171-pnad-continua.html>.

FLANAGAN, D. **JavaScript: o guia definitivo**. [S.l.]: Bookman Editora, 2012.

FRAIN, B. **Responsive Web Design with HTML5 and CSS: Build future-proof responsive websites using the latest HTML5 and CSS techniques**. [S.l.]: Packt Publishing Ltd, 2022.

GARDNER, B. S. Responsive web design: Enriching the user experience. **Sigma Journal: Inside the Digital Ecosystem**, v. 11, n. 1, p. 13–19, 2011.

GERADE, N. Responsividade em aplicações web e mobile. 002, 2024.

GERCHEV, I. **Tailwind CSS**. [S.l.]: SitePoint Pty Ltd, 2022.

GHIOTTO, G. *et al.* On the nature of merge conflicts: a study of 2,731 open source java projects hosted by github. **IEEE Transactions on Software Engineering**, IEEE, v. 46, n. 8, p. 892–915, 2018.

GITHUB. **Responsive Foundations**. 2023. Accessed: 2023-11-15. Disponível em: <https://primer.style/product/getting-started/foundations/responsive/>.

GOMES, A.; HENRIQUES, J.; MENDES, A. J. Uma proposta para ajudar alunos com dificuldades na aprendizagem inicial de programação de computadores. **Educação, Formação e Tecnologias**, Associação Portuguesa de Telemática Educativa, v. 1, n. 01, p. 93–103, 2008.

IIEC. **Cxxdroid: C/C++ Compiler for Android**. 2025. Acessado em 26 mar. 2025. Disponível em: <https://play.google.com/store/apps/details?id=ru.iiec.cxxdroid&hl=pt_BR>.

JALOLOV, T. Frontend and backend developer difference and advantages. **Multidisciplinary Journal of Science and Technology**, v. 4, n. 2, p. 178–179, 2024.

JENKINS, T. On the difficulty of learning to program. In: CITESEER. **Proceedings of the 3rd Annual Conference of the LTSN Centre for Information and Computer Sciences**. [S.l.], 2002. v. 4, n. 2002, p. 53–58.

JOHNSON, R. E.; FOOTE, B. Designing reusable classes. **Journal of object-oriented programming**, v. 1, n. 2, p. 22–35, 1988.

JUNIOR, S. M. da S.; MORAIS, M. A. C. de. Mapeamento de ferramentas utilizadas no ensino de programação na ead. **Brazilian Journal of Development**, v. 6, n. 11, p. 87466–87476, 2020.

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

LEE, L.-H.; LAM, K.-Y.; HUI, P. Exploring user engagement by diagnosing visual guides in onboarding screens with linear regression and xgboost. **Displays**, Elsevier, v. 87, p. 102975, 2025.

LEITE, T. d. O. *et al.* Design de ux e prototipagem: Moldando as escolhas do usuário. In: SBC. **Escola Regional de Informática de Mato Grosso (ERI-MT)**. [S.l.], 2024. p. 189–195.

LEWIS, J. R. Psychometric evaluation of the pssuq using data from five years of usability studies. **International Journal of Human-Computer Interaction**, Taylor & Francis, v. 14, n. 3-4, p. 463–488, 2002.

LOUDON, K. Desenvolvimento de grandes aplicações web. **Revista Telfract**, v. 1, n. 1, 2018.

MANZINI, E. J. Inclusão e acessibilidade. **Revista da Sobama**, v. 10, n. 1, p. 31–36, 2005.

MARCOTTE, E. **Responsive web design: A book apart n 4**. [S.l.]: Editions Eyrolles, 2017.

____. **Responsive Design: Patterns and Principles**. 2nd. ed. [S.l.]: A Book Apart, 2018.

MASRUROH, S. U. *et al.* Evaluation of usability and accessibility of mobile application for people with disability: Systematic literature review. In: IEEE. **2022 International Conference on Science and Technology (ICOSTECH)**. [S.l.], 2022. p. 1–7.

MEHRABANI, A. **Intro.js: User Onboarding and Product Walkthrough Library**. 2023. JavaScript library for step-by-step guides and feature introduction. Disponível em: <https://introjs.com>.

MOHD, T. K. *et al.* Comparative analysis on various css and javascript frameworks. **J. Softw.**, v. 17, n. 6, p. 282–291, 2022.

MONTEIRO, E. d. J. P. *et al.* Recomendações para adoção de padrões de usabilidade e responsividade no desenvolvimento de aplicações web. **Brazilian Journal of Development**, v. 6, n. 5, p. 24790–24819, 2020.

NASCIMENTO, K. A. S. do *et al.* Ferramenta de prototipagem para criação de um aplicativo para o ensino na saúde. In: SBC. **Workshop de Informática na Escola (WIE)**. [S.l.], 2020. p. 509–513.

NIELSEN, J. **10 Usability Heuristics for User Interface Design**. 1994. Acessado em: 1 abr. 2025. Disponível em: <https://www.nngroup.com/articles/ten-usability-heuristics/>.

OLIVEIRA, A. D. A.; ELER, M. M. Analyzing accessibility, usability, and user experience in mobile apps through user reviews: An extended systematic literature review. **Journal on Interactive Systems**, v. 16, n. 1, p. 641–665, 2025.

ONLINEGDB. **OnlineGDB: Online Compiler and Debugger**. 2025. Acessado em 26 mar. 2025. Disponível em: <https://www.onlinegdb.com/>.

PANDUWIKA, P.; SOLEHATIN, S. Performance measurement implementation on the smart fisheries village website using pagespeed insight. **Journal of Soft Computing Exploration**, v. 5, n. 2, p. 161–172, 2024.

PARLAKKILIÇ, A. Evaluating the effects of responsive design on the usability of academic websites in the pandemic. **Education and information technologies**, Springer, v. 27, n. 1, p. 1307–1322, 2022.

PLAZA, B. Google analytics for measuring website performance. **Tourism Management**, Elsevier, v. 32, n. 3, p. 477–481, 2011.

POWELL, T. **HTML & CSS: the complete reference**. [S.l.]: McGraw-Hill, Inc., 2010.

PRESSMAN, R. S.; MAXIM, B. R. **Engenharia de software-9**. [S.l.]: McGraw Hill Brasil, 2021.

RAO, P. R.; CHAURASIA, A.; SINGH, S. Modern web design: Utilizing html5, css3, and responsive techniques. **The International Journal of Research and Innovation in Dynamics of Engineering**, v. 1, n. 8, p. a1–a18, 2023.

RÍOS, J. C. C.; EMBURY, S. M.; ERASLAN, S. A unifying framework for the systematic analysis of git workflows. **Information and Software Technology**, Elsevier, v. 145, p. 106811, 2022.

RODRIGUES, M. B.; BECHER, A. Acessibilidade e usabilidade na web. **Mato Grosso Digital/SUCESU-MT**, p. 24, 2008.

ROSA, A. F. *et al.* European portuguese validation of the post-study system usability questionnaire (pssuq). In: IEEE. **2015 10th Iberian Conference on Information Systems and Technologies (CISTI)**. [S.l.], 2015. p. 1–5.

SALMI, H. A. Comparative css frameworks. **Multi-Knowledge Electronic Comprehensive Journal for Education and Science Publications (MECSJ)**, v. 66, 2023.

SANTOS, I. *et al.* Software solutions for newcomers' onboarding in software projects: A systematic literature review. **Information and Software Technology**, Elsevier, v. 177, p. 107568, 2025.

SARIOĞLU, A.; METIN, H.; BORK, D. Accessibility in conceptual modeling—a systematic literature review, a keyboard-only uml modeling tool, and a research roadmap. **Data & Knowledge Engineering**, Elsevier, p. 102423, 2025.

SHAHZAD, F. Modern and responsive mobile-enabled web applications. **Procedia Computer Science**, Elsevier, v. 110, p. 410–415, 2017.

SHIVAKUMAR, S. K. Modern web performance optimization. **Methods, Tools, and Patterns to Speed Up Digital Platforms**, Springer, 2020.

SOMMERVILLE, I. **Software Engineering, 9/E**. [S.l.]: Pearson Education India, 2011.

SOUZA, O. d.; TABOSA, H. R. Virando a página: um novo conceito de acessibilidade na web para deficientes visuais. Informação & Sociedade: Estudos, 2014.

SPURLOCK, J. **Bootstrap: responsive web development**. [S.l.]: "O'Reilly Media, Inc.", 2013.

STAIANO, F. **Designing and Prototyping Interfaces with Figma: Learn essential UX/UI design principles by creating interactive prototypes for mobile, tablet, and desktop**. [S.l.]: Packt Publishing Ltd, 2022.

STOIBER, C. **Visualization Onboarding. Supporting Users in Understanding Unfamiliar Visual Representations**. Tese (Doutorado) — Technische Universität Wien, 2024.

SUN, S.; CAO, S. X. The web development technology research of cross platform mobile application. **Applied Mechanics and Materials**, Trans Tech Publ, v. 644, p. 3090–3093, 2014.

SUSIN, G. Webalgo: Desenvolvimento de um compilador da linguagem c e uma máquina virtual baseada em registradores utilizando c3e na web. **UCS**, 2023.

SUSIN, G. L. Trabalho de Conclusão de Curso, **WEBALGO: Desenvolvimento de um Compilador da Linguagem C e uma Máquina Virtual Baseada em Registradores Utilizando C3E na Web**. Caxias do Sul: [s.n.], 2024. Orientador: Prof. Dr. Ricardo Vargas Dorneles.

SZEKELY, P. User interface prototyping: Tools and techniques. In: SPRINGER. **Software Engineering and Human-Computer Interaction: ICSE'94 Workshop on SE-HCI: Joint Research Issues Sorrento, Italy, May 16–17, 1994 Proceedings**. [S.l.], 1995. p. 76–92.

TERRA, J. C. *et al.* Usabilidade: conceitos centrais. **Online. Disponível em**, 2008.

UTO, N.; MELO, S. Vulnerabilidades em aplicações web e mecanismos de proteção. **Minicursos SBSeg**, p. 237–283, 2009.

VALE, G. *et al.* On the relation between github communication activity and merge conflicts. **Empirical Software Engineering**, Springer, v. 25, p. 402–433, 2020.

VERBER, D. *et al.* Equidopa: A responsive web application for the levodopa equivalent dose calculator. **Computer methods and programs in biomedicine**, Elsevier, v. 196, p. 105633, 2020.

WROBLEWSKI, L. **Mobile first: Preface de jeffrey zeldmann**. [S.l.]: Editions Eyrolles, 2012.

ZAKAS, N. C. **High performance JavaScript: build faster web application interfaces**. [S.l.]: "O'Reilly Media, Inc.", 2010.

____. **Understanding ECMAScript 6: the definitive guide for JavaScript developers**. [S.l.]: No Starch Press, 2016.

ZHANG, H. *et al.* A web-based, mobile-responsive application to screen health care workers for covid-19 symptoms: rapid design, deployment, and usage. **JMIR formative research**, JMIR Publications Inc., Toronto, Canada, v. 4, n. 10, p. e19533, 2020.

APÊNDICE A – QUESTIONÁRIO PSSUQ

19 Perguntas utilizadas para o questionário do PSSUQ, utilizados para validar a usabilidade dos sistemas:

1. No geral, estou satisfeito com a facilidade de usar este sistema.
2. Foi simples de usar este sistema.
3. Eu poderia efetivamente completar as tarefas e cenários usando este sistema.
4. Consegui concluir as tarefas e cenários rapidamente usando este sistema.
5. Consegui concluir as tarefas e cenários eficientemente usando este sistema.
6. Eu me senti confortável usando este sistema.
7. Foi fácil aprender a usar este sistema.
8. Acredito que posso me tornar produtivo rapidamente usando este sistema.
9. O sistema deu mensagens de erro que me disseram claramente como corrigir os problemas.
10. Sempre que cometi um erro ao usar o sistema, eu consegui me recuperar com facilidade e rapidez.
11. As informações (como ajuda on-line, mensagens na tela e outras documentações) fornecidas com este sistema eram claras.
12. Foi fácil encontrar as informações que eu precisava.
13. As informações fornecidas para o sistema eram fáceis de entender.
14. As informações foram eficazes e me ajudaram a completar as tarefas e cenários.
15. A organização das informações nas telas do sistema foi clara.
16. A interface deste sistema foi agradável.
17. Eu gostei de usar a interface deste sistema.
18. Este sistema tem todas as funções e capacidades que eu espero que ele tenha.
19. No geral, estou satisfeito com este sistema.