DESENVOLVIMENTO DE UMA APLICAÇÃO WEB DE LISTA DE TAREFAS COM INTEGRAÇÃO DE CALENDÁRIO USANDO HTML, CSS E JAVASCRIPT

Acadêmico: Rafael Martinelli Araujo Lima

Turma: FLD209110CET

Tutor: Marcio Poffo

**RESUMO**

Este trabalho descreve o desenvolvimento de uma aplicação web de lista de tarefas utilizando HTML, CSS e JavaScript, com o objetivo de criar uma ferramenta interativa, responsiva e acessível para gerenciamento de atividades. A aplicação permite criar, editar, marcar como concluído e remover tarefas, configurar lembretes com notificações sonoras, exibir tarefas em um calendário mensal com destaque para dias com tarefas, e alternar entre temas claro/escuro. A metodologia combinou abordagem qualitativa, pesquisa bibliográfica e desenvolvimento prático, utilizando Visual Studio Code e navegadores para testes de usabilidade. Testes com cinco usuários confirmaram a intuitividade da interface, com destaque para modais, acessibilidade via ARIA e design responsivo. A persistência de dados usa localStorage, e o código é modular, facilitando manutenção. Conclui-se que HTML, CSS e JavaScript são eficazes para aplicações web modernas, com potencial para expansões como integração com calendários externos, visão semanal e categorização de tarefas.

**Palavras-chave**: *Desenvolvimento Web*, *Lista de Tarefas*, *JavaScript*.

**INTRODUÇÃO**

O desenvolvimento de aplicações web é crucial na era digital, atendendo à demanda por ferramentas de produtividade. Este trabalho apresenta uma aplicação de lista de tarefas com integração de calendário, utilizando HTML para estruturação, CSS para estilização e JavaScript para interatividade. A pergunta central é: como desenvolver uma aplicação web funcional, responsiva e acessível utilizando tecnologias front-end? O objetivo geral é criar uma ferramenta intuitiva para estudantes e profissionais organizarem tarefas. Os objetivos específicos são: (1) estruturar a interface com HTML semântico e atributos ARIA; (2) implementar estilos responsivos com CSS, incluindo temas claro/escuro e cores vibrantes; (3) desenvolver funcionalidades dinâmicas com JavaScript, como edição de tarefas, lembretes com notificações sonoras e visualização de calendário mensal. A aplicação aborda desafios como acessibilidade, responsividade e usabilidade, alinhando-se a tendências de design centrado no usuário.

**FUNDAMENTAÇÃO TEÓRICA**

O desenvolvimento web moderno baseia-se em HTML, CSS e JavaScript. Niederst Robbins (2018) destaca que o HTML5 oferece marcação semântica, essencial para acessibilidade. A aplicação usa elementos como <header>, <main>, <section> e <div>, com atributos ARIA (ex.: role="listitem", aria-checked) para suportar leitores de tela.

O CSS é responsável pela estilização responsiva. Meyer (2020) enfatiza que flexbox e grid adaptam interfaces a diferentes dispositivos. A aplicação utiliza flexbox na seção de entrada e grid no calendário, com media queries para telas menores, garantindo usabilidade em dispositivos móveis.

JavaScript proporciona interatividade. Haverbeke (2018) explica que a manipulação do DOM atualiza interfaces em tempo real. Funções como TaskManager.updateTaskList() e CalendarManager.updateMonthView() gerenciam a exibição dinâmica de tarefas, com validação via Utils.validateTask().

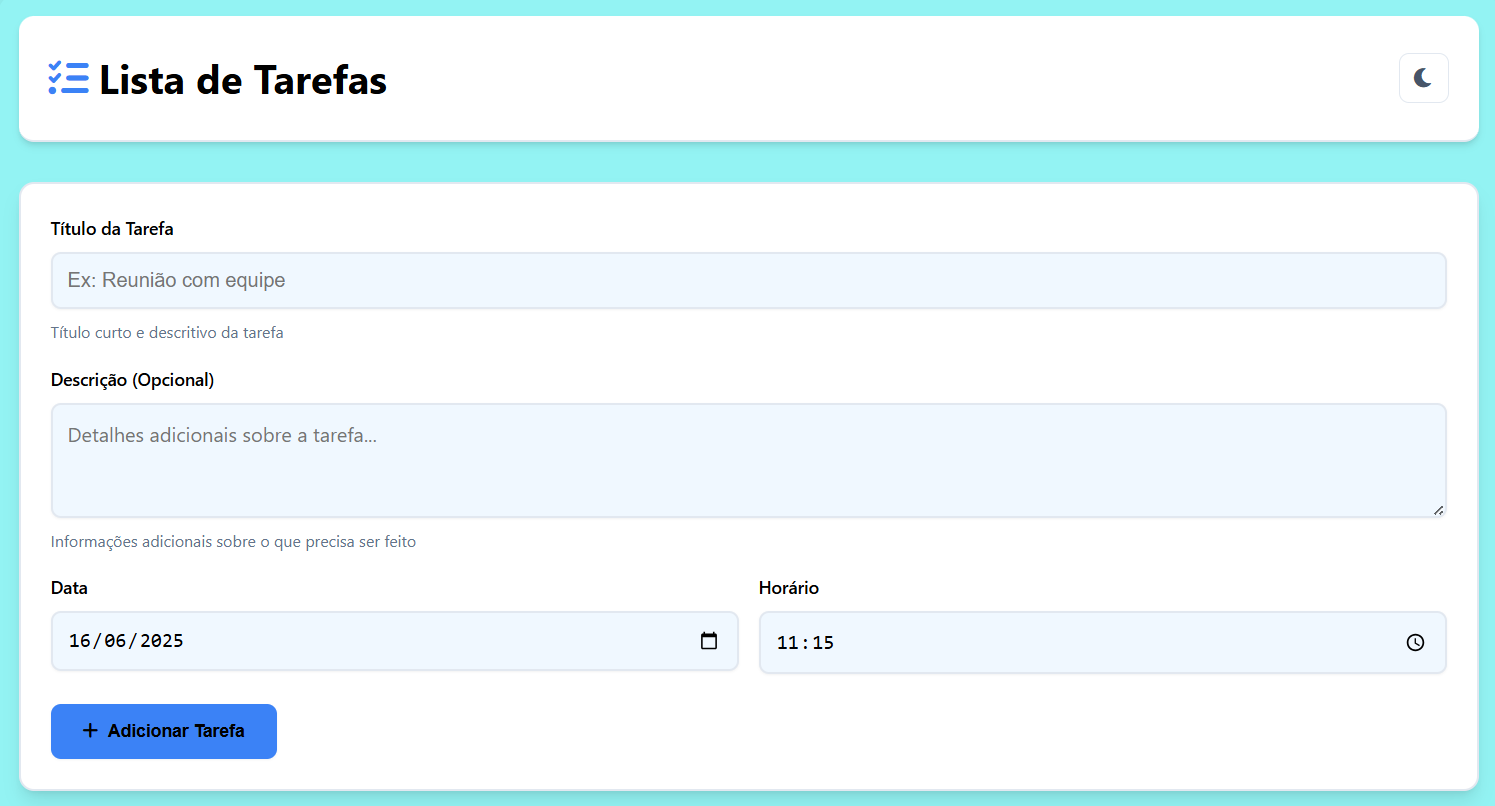
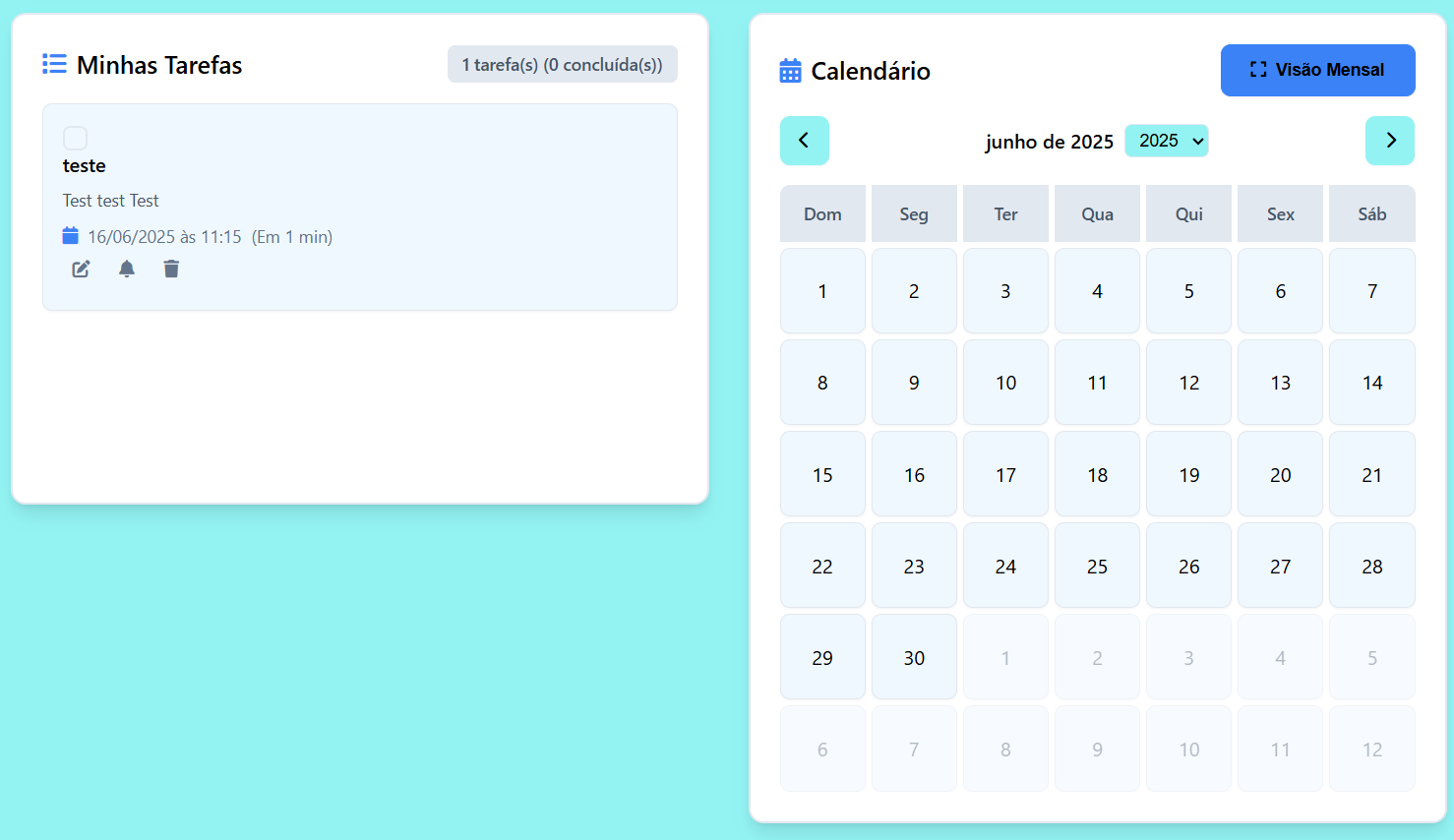
A usabilidade é crítica para interfaces intuitivas. Nielsen (2012) recomenda minimizar a curva de aprendizado. A aplicação oferece botões claros, modais para lembretes e feedback visual (ex.: animação fadeIn), facilitando a interação.  
   
Figura1:

Figura2:  
Fonte: Elaborado pelo autor, 2025.  
  
A modularidade no código JavaScript aumenta a manutenibilidade. McFarland (2015) recomenda funções específicas, como ThemeManager.toggleTheme() para alternar temas e Storage.saveTasks() para persistência em localStorage, adotadas na aplicação.

A responsividade é essencial. Marcotte (2012) define o design responsivo como adaptação a diferentes telas. A aplicação usa unidades relativas (vw, %) e temas claro/escuro, ajustando-se automaticamente via [data-theme="dark"].

A segurança protege contra erros. Flanagan (2020) sugere validar entradas. A função Utils.validateTask() verifica campos vazios e datas inválidas, exibindo alertas claros para o usuário.

Modais focam a atenção do usuário. Johnson (2014) destaca sua eficácia em tarefas específicas. O modal de lembrete da aplicação, com animação fadeIn, permite configurar notificações com opções predefinidas.

A acessibilidade é priorizada. O W3C (2020) recomenda atributos ARIA e foco visível. A aplicação implementa aria-label, aria-checked, e estilos :focus, garantindo navegação acessível com teclados e leitores de tela.

**METODOLOGIA**

A pesquisa utilizou o método aplicado, com abordagem qualitativa, focando no desenvolvimento de um protótipo funcional. O tipo de pesquisa foi bibliográfica e prática. A pesquisa bibliográfica consultou livros e artigos sobre desenvolvimento web, usabilidade e design, como Niederst Robbins (2018) e Haverbeke (2018). A pesquisa prática envolveu a codificação da aplicação, usando Visual Studio Code e navegadores (Chrome, Firefox) para testes.

Os instrumentos incluíram ferramentas de codificação (HTML, CSS, JavaScript, Font Awesome), documentações (W3C, MDN Web Docs) e testes de usabilidade com cinco usuários. As fontes foram livros, artigos e documentações técnicas. O processo envolveu: (1) planejamento da interface, com esboços e paleta de cores (ex.: azul turquesa #93f3f3); (2) codificação, separando estilos em styles.css e lógica em script.js e check\_notifications.js; (3) implementação de funcionalidades, incluindo edição de tarefas, notificações e visão mensal; (4) testes de usabilidade, avaliando tempo de resposta do modal, clareza de botões, legibilidade de cores e compatibilidade móvel.

Os testes de usabilidade verificaram a intuitividade da interface, com feedback qualitativocollected via observações e questionários, levando a ajustes no contraste e na navegação por teclado.

**CONSIDERAÇÕES**

A aplicação alcançou os objetivos, oferecendo uma ferramenta funcional, responsiva e acessível. Usuários podem criar, editar e remover tarefas, configurar lembretes com notificações sonoras, e visualizar atividades em um calendário mensal com destaque visual para dias com tarefas (ícone de sino). Testes confirmaram a usabilidade, com destaque para modais, temas claro/escuro (usando azul turquesa #93f3f3 e azul escuro #293a56) e acessibilidade via ARIA.

Aprendizados incluem a importância de modularidade, animações CSS e validação de dados. Desafios como otimizar notificações e garantir responsividade foram superados. Limitações incluem a ausência de visão semanal, categorização de tarefas por prioridade e integração com calendários externos. Próximos passos sugerem adicionar visão semanal, categorização de tarefas (ex.: alta, média, baixa prioridade), exportação para CSV e sincronização com APIs de calendário, como ical.js. O projeto demonstra o potencial de HTML, CSS e JavaScript para soluções de produtividade.

**REFERÊNCIAS BIBLIOGRÁFICAS**

FLANAGAN, D. *JavaScript: The Definitive Guide*. 7. ed. Sebastopol: O’Reilly Media, 2020.  
HAVERBEKE, M. *Eloquent JavaScript: A Modern Introduction to Programming*. 3. ed. San Francisco: No Starch Press, 2018.  
JOHNSON, J. *Designing with the Mind in Mind: Simple Guide to Understanding User Interface Design Guidelines*. 2. ed. Waltham: Morgan Kaufmann, 2014.  
LIDWELL, W.; HOLDEN, K.; BUTLER, J. *Universal Principles of Design*. 2. ed. Beverly: Rockport Publishers, 2010.  
MARCOTTE, E. *Responsive Web Design*. 2. ed. [S.l.]: A Book Apart, 2012.  
MCFARLAND, D. S. *JavaScript & jQuery: The Missing Manual*. 3. ed. Sebastopol: O’Reilly Media, 2015.  
MEYER, E. A. *CSS: The Definitive Guide*. 5. ed. Sebastopol: O’Reilly Media, 2020.  
NIEDERST ROBBINS, A. *Learning Web Design: A Beginner’s Guide to HTML, CSS, JavaScript, and Web Graphics*. 5. ed. Sebastopol: O’Reilly Media, 2018.  
NIELSEN, J. *Usability 101: Introduction to Usability*. [S.l.]: Nielsen Norman Group, 2012. Disponível em: <https://www.nngroup.com/articles/usability-101-introduction-to-usability/>. Acesso em: 10 jun. 2025.  
W3C. *Web Accessibility Initiative (WAI): Accessible Rich Internet Applications (ARIA)*. [S.l.]: W3C, 2020. Disponível em: <https://www.w3.org/WAI/standards-guidelines/aria/>. Acesso em: 10 jun. 2025.