Sergio Henrique Mendes de Assis

Avaliação de ferramentas para criação de *dashboards*: um estudo de caso aplicado ao sistema de regularização fundiária do SEAPA-MG

Belo Horizonte

2026

Sergio Henrique Mendes de Assis

Avaliação de ferramentas para criação de *dashboards*: um estudo de caso aplicado ao sistema de regularização fundiária do SEAPA-MG

Trabalho de Conclusão de Curso apresentado ao Curso de Engenharia de Computação do Centro Federal de Educação Tecnológica de Minas Gerais, como requisito parcial para a obtenção do título de Bacharel em Engenharia de Computação.

Centro Federal de Educação Tecnológica de Minas Gerais – CEFET-MG

Departamento de Computação

Curso de Engenharia da Computação

Orientador: Prof. Dr. Edson Marchetti da Silva

Belo Horizonte

2026

Assis, Sergio Henrique Mendes de
A848a Avaliação de ferramentas para criação de Dashboards: um estudo de caso aplicado ao sistema de regularização fundiária do SEAPA-MG / Sergio Henrique Mendes de Assis. – 2026.
58 f. : il.

Trabalho de conclusão de curso apresentado ao curso de Engenharia de Computação do Centro Federal de Educação Tecnológica de Minas Gerais.
Orientador: Edson Marchetti da Silva.
Trabalho de conclusão de curso (graduação) – Centro Federal de Educação Tecnológica de Minas Gerais, Departamento de Computação.

1. Dashboards (Sistemas de informação gerencial) – Teses. 2. Visualização de informações – Teses. 3. Interfaces de usuário (Sistemas computacionais) – Teses. 4. Google Analytics – Teses. 5. Apache ECharts (Software) – Teses. 6. Bibliotecas digitais – Teses. I. Silva, Edson Marchetti. II. Centro Federal de Educação Tecnológica de Minas Gerais. Departamento de Computação. III. Título.

CDD 005.7406

Elaboração da ficha catalográfica pela bibliotecária Jane Marangon Duarte, CRB 6ª 1592 / Cefet/MG

MINISTÉRIO DA EDUCAÇÃO
CENTRO FEDERAL DE EDUCAÇÃO TECNOLÓGICA DE MINAS GERAIS
DEPARTAMENTO DE COMPUTAÇÃO - NG

FOLHA DE APROVAÇÃO DE TCC Nº 14 / 2026 - DECOM (11.56.03)

Nº do Protocolo: 23062.032314/2026-90

Belo Horizonte-MG, 23 de junho de 2026.

SÉRGIO HENRIQUE MENDES DE ASSIS

AVALIAÇÂO DE FERRAMENTAS PARA CRIAÇÃO DE *DASHBOARDS*: um estudo de caso aplicado ao sistema de regularização fundiária do SEAPA-MG

Trabalho de Conclusão de Curso apresentado ao Curso de **Engenharia de Computação** do Centro Federal de Educação Tecnológica de Minas Gerais, do Campus Nova Gameleira, como parte do requisito para obtenção do grau de Bacharel em Engenharia de Computação.

Aprovado em 16 de junho de 2026.

Banca Examinadora:

Prof. Dr. Edson Marchetti da Silva - Orientador
Centro Federal de Educação Tecnológica de Minas Gerais

Prof. Dra. Kecia Aline Marques Ferraeira
Centro Federal de Educação Tecnológica de Minas Gerais

Prof. Dr. Ismael Santana Silva
Centro Federal de Educação Tecnológica de Minas Gerais

Belo Horizonte
2026

(Assinado digitalmente em 25/06/2026 12:32)
EDSON MARCHETTI DA SILVA
PROFESSOR ENS BASICO TECN TECNOLOGICO
DECOM (11.56.03)
Matrícula: 1509224

(Assinado digitalmente em 25/06/2026 15:39)
ISMAEL SANTANA SILVA
PROFESSOR ENS BASICO TECN TECNOLOGICO
DECOM (11.56.03)
Matrícula: 1028758

(Assinado digitalmente em 30/06/2026 08:07)
KECIA ALINE MARQUES FERREIRA
PROFESSOR ENS BASICO TECN TECNOLOGICO
DECOM (11.56.03)
Matrícula: 1670931

Visualize o documento original em **https://sig.cefetmg.br/public/documentos/index.jsp** informando seu número: **14**, ano: **2026**, tipo: **FOLHA DE APROVAÇÃO DE TCC**, data de emissão: **23/06/2026** e o código de verificação: **d721f2ef60**

Agradecimentos

Agradeço, primeiramente, à minha mãe, Silvana, por todo o amor, apoio e dedicação ao longo da minha vida. Foi graças aos seus esforços, incentivos e sacrifícios que tive a oportunidade de estudar, crescer e chegar até este momento tão importante da minha trajetória. Esta conquista também é sua.

Agradeço também aos meus tios e tias, que sempre estiveram presentes, oferecendo apoio, ensinamentos e exemplos que contribuíram para a formação dos meus valores pessoais e profissionais.

Aos meus amigos, deixo minha sincera gratidão pela compreensão, companheirismo e apoio nos momentos mais difíceis. A amizade de vocês foi fundamental para superar os desafios encontrados ao longo do curso e da vida.

Agradeço ainda aos professores do CEFET-MG, que contribuíram significativamente para minha formação acadêmica e profissional. Em diversos momentos desta jornada enfrentei dificuldades e questionei minha própria capacidade, mas encontrei professores dispostos a orientar, compreender e incentivar meu desenvolvimento.

Em especial, agradeço ao meu orientador, Professor Edson Marchetti da Silva, pela orientação, paciência, disponibilidade e confiança durante a realização deste trabalho. Seus ensinamentos, sugestões e acompanhamento foram fundamentais para a conclusão desta pesquisa e para meu crescimento acadêmico.

Agradeço também aos professores do Departamento de Computação (DECOM), que ao longo do curso compartilharam conhecimentos, experiências e valores que certamente levarei para minha vida profissional.

Por fim, agradeço a todos que, de alguma forma, contribuíram para esta conquista e fizeram parte desta caminhada. Cada apoio recebido teve papel importante para que este momento se tornasse realidade.

"Sempre existe uma questão maior, mais abrangente: o grande objetivo. O problema é que, se você fica obcecado pela enormidade desse objetivo, acaba perdendo o foco."

(Blake Crouch)

Resumo

O presente Trabalho de Conclusão de Curso tem como objetivo desenvolver e avaliar comparativamente diferentes bibliotecas e *frameworks front-end* aplicados à criação de *dashboards* interativos para visualização de dados da regularização fundiária da Secretaria de Estado de Agricultura, Pecuária e Abastecimento de Minas Gerais (SEAPA-MG). A pesquisa caracteriza-se como aplicada, com abordagem quali-quantitativa, integrando desenvolvimento prático e análise empírica de desempenho e usabilidade. Para a condução do estudo, foi implementado um *dashboard* em React utilizando inicialmente a biblioteca Victory Charts, seguido da construção de versões equivalentes com Recharts e ECharts. A avaliação foi realizada com base em métricas de desempenho obtidas por ferramentas como Lighthouse, React Profiler e Web Vitals API, incluindo indicadores como tempo de renderização, Largest Contentful Paint (LCP) e Interaction to Next Paint (INP), além da análise de consumo de recursos. Complementarmente, foi realizada uma análise qualitativa fundamentada nas heurísticas de Nielsen (1995) e nos princípios de design de interação, considerando aspectos observáveis da interface, da interação e da implementação das bibliotecas analisadas. Os resultados evidenciam diferenças significativas entre as bibliotecas analisadas, destacando aspectos de desempenho técnico, fluidez de interação e clareza visual. Observou-se que não há uma solução única ideal, sendo a escolha da ferramenta dependente do contexto de aplicação e dos requisitos do sistema. Como contribuição, o trabalho apresenta diretrizes práticas para a seleção e utilização de bibliotecas de visualização em aplicações web, ampliando o entendimento sobre a relação entre desempenho, usabilidade e experiência do usuário em *dashboards*. Os resultados obtidos visaram apoiar o aprimoramento do sistema SIGET da SEAPA-MG e servir como referência para o desenvolvimento de soluções similares no setor público.

Palavras-chave: *Dashboards*; visualização de dados; desempenho *front-end*; React; usabilidade; métricas Web Vitals; Echarts; Recharts; Victory Charts.

Abstract

This Undergraduate Thesis aims to develop and comparatively evaluate different *front-end* libraries and *frameworks* applied to the creation of interactive *dashboards* for visualizing land regularization data from the Secretaria de Estado de Agricultura, Pecuária e Abastecimento de Minas Gerais (SEAPA-MG). The research is characterized as applied, with a quali-quantitative approach, integrating practical development and empirical analysis of performance and usability. To conduct the study, a *dashboard* was implemented in React initially using the Victory Charts library, followed by the development of equivalent versions using Recharts and ECharts. The evaluation was based on performance metrics obtained through tools such as Lighthouse, React Profiler, and Web Vitals API, including indicators such as rendering time, Largest Contentful Paint (LCP), and Interaction to Next Paint (INP), in addition to resource consumption analysis. Additionally, a qualitative analysis was conducted based on Nielsen's heuristics (1995) and interaction design principles, considering observable aspects of the interface, interaction, and implementation of the analyzed libraries. The results show significant differences among the evaluated libraries, highlighting aspects related to technical performance, interaction fluidity, and visual clarity. It was observed that there is no single ideal solution, since the choice of the tool depends on the application context and system requirements. As a contribution, this work presents practical guidelines for selecting and using visualization libraries in web applications, expanding the understanding of the relationship between performance, usability, and user experience in *dashboards*. The results obtained aimed to support the improvement of the SIGET system of SEAPA-MG and to serve as a reference for the development of similar solutions in the public sector.

Keywords: Dashboards; data visualization; front-end performance; React; usability; Web Vitals metrics; Echarts; Recharts; Victory Charts.

Lista de ilustrações

Lista de tabelas

Lista de abreviaturas e siglas

API	Application Programming Interface
BI	Business Intelligence
CDN	Content Delivery Network
CLS	Cumulative Layout Shift
CSS	Cascading Style Sheets
DOM	Document Object Model
FCP	First Contentful Paint
FID	First Input Delay
FPS	Frames Per Second
IHC	Interação Humano-Computador
INEP	Instituto Nacional de Estudos e Pesquisas Educacionais Anísio Teixeira
INP	Interaction to Next Paint
LCP	Largest Contentful Paint
SEAPA-MG	Secretaria de Estado de Agricultura, Pecuária e Abastecimento de Minas Gerais
SVG	Scalable Vector Graphics
TBT	Total Blocking Time
TCC	Trabalho de Conclusão de Curso
TTI	Time to Interactive
UX	User Experience

Sumário

1 Introdução

A crescente disponibilização de dados públicos e institucionais tem ampliado a necessidade de desenvolver soluções capazes de transformar grandes volumes de dados em informações visuais claras, acessíveis e úteis para a tomada de decisão. No setor público, essa necessidade é ainda mais evidente, uma vez que diferentes órgãos governamentais lidam com quantidades massivas de dados que, quando bem estruturados e visualizados, podem apoiar a gestão, aumentar a transparência e facilitar análises estratégicas (BRASIL, 2024; SECRETARIA DO PLANEJAMENTO E GESTÃO DO ESTADO DO CEARÀ, 2025; AGÊNCIA SP, 2024).

A Secretaria de Estado de Agricultura, Pecuária e Abastecimento de Minas Gerais (SEAPA-MG), órgão responsável por políticas públicas ligadas à agricultura, pecuária, abastecimento e desenvolvimento rural no estado, administra processos de regularização fundiária cujos dados são relevantes para o acompanhamento e planejamento de ações governamentais. Nesse contexto, o desenvolvimento *front-end* exerce papel fundamental, pois é responsável por transformar dados e regras de negócio em interfaces visuais acessíveis, responsivas e adequadas ao uso. A qualidade de um *dashboard* depende não apenas dos dados disponíveis, mas também das tecnologias utilizadas em sua construção, da forma como os componentes são renderizados, da fluidez das interações e da capacidade da aplicação de se manter eficiente diante de diferentes volumes de informação.

Apesar da importância de gerar conhecimento a partir dessas informações, muitas instituições ainda enfrentam dificuldades na apresentação de dados de forma integrada, dinâmica e visualmente compreensível. Os *dashboards* interativos surgem como uma alternativa eficiente, pois permitem a exploração de indicadores, identificação de tendências e análise de padrões de maneira mais intuitiva e organizada (FEW, 2014).

Entre as tecnologias utilizadas no desenvolvimento de interfaces web modernas, o React destaca-se por sua abordagem baseada em componentes reutilizáveis, favorecendo a organização do código, a manutenção da aplicação e a construção de interfaces dinâmicas. No contexto de *dashboards*, essa característica é especialmente relevante, pois gráficos, filtros, cartões informativos e demais elementos visuais podem ser estruturados como componentes independentes, facilitando a evolução do sistema e a comparação entre diferentes bibliotecas de visualização.

A motivação deste trabalho surgiu a partir da participação do Centro Federal de Educação Tecnológica de Minas Gerais (CEFET-MG), instituição na qual o autor cursa Engenharia de Computação, em um projeto voltado ao desenvolvimento de um sistema para gerenciamento de processos de regularização fundiária da Secretaria de Estado de

Agricultura, Pecuária e Abastecimento de Minas Gerais (SEAPA-MG). Durante esse projeto, identificou-se a necessidade de criação de *dashboards* para apoiar a visualização e interpretação dos dados relacionados aos processos de regularização fundiária. A partir dessa demanda prática, o orientador deste trabalho apresentou a oportunidade de desenvolver uma pesquisa aplicada voltada à comparação de bibliotecas de visualização de dados, com o objetivo de contribuir para a escolha da solução mais adequada ao contexto do sistema da Secretaria de Estado de Agricultura, Pecuária e Abastecimento de Minas Gerais (SEAPA-MG).

Dessa forma, este trabalho possui caráter aplicado, pois parte de uma demanda real de desenvolvimento e busca gerar resultados que possam apoiar decisões técnicas no contexto do sistema desenvolvido para a Secretaria de Estado de Agricultura, Pecuária e Abastecimento de Minas Gerais (SEAPA-MG). O foco da pesquisa está na comparação entre bibliotecas *front-end* para construção de *dashboards* em React, considerando critérios como desempenho técnico, responsividade, tamanho do *bundle*[1], fluidez de interação, clareza visual e complexidade de implementação.

Foram selecionadas três bibliotecas para a comparação: Victory Charts, Recharts e ECharts. A escolha da Victory Charts ocorreu porque essa biblioteca já havia sido utilizada anteriormente pelo autor em seu estágio, o que proporcionou familiaridade inicial com sua estrutura e funcionamento. Já Recharts e ECharts foram selecionadas por aparecerem com frequência em pesquisas e levantamentos sobre bibliotecas de visualização de dados utilizadas em aplicações React, sendo frequentemente apresentadas como alternativas consolidadas para construção de gráficos e *dashboards*. Assim, a seleção das três bibliotecas combinou experiência prática prévia, recorrência em pesquisas sobre ferramentas de visualização e relevância no ecossistema React.

Neste trabalho, foi desenvolvido um *dashboard* interativo para visualização dos dados de regularização fundiária da Secretaria de Estado de Agricultura, Pecuária e Abastecimento de Minas Gerais (SEAPA-MG), utilizando React. Inicialmente, a implementação foi realizada com a biblioteca Recharts e, posteriormente, foram desenvolvidas versões equivalentes utilizando Victory Charts e ECharts. As três versões mantiveram a mesma estrutura visual, o mesmo conjunto de dados e as mesmas rotas de comunicação com o *back-end*, de forma a garantir maior consistência na comparação entre as bibliotecas analisadas.

Durante o desenvolvimento, foram implementados gráficos de pizza e gráficos de barras com recursos de interação, legendas customizadas e *tooltips*, buscando manter equivalência visual e funcional entre as diferentes soluções. Além disso, foram adicionados marcadores de desempenho diretamente no código-fonte para mensuração do tempo de

[1] Tamanho total do "pacote" de um software, que agrupa código, recursos como imagens e CSS, além de bibliotecas utilizadas pela aplicação.

renderização dos componentes. Para realização dos experimentos, foi utilizada uma base de dados local populada artificialmente, devido à limitação de registros disponíveis no ambiente de produção.

As avaliações de desempenho foram realizadas a partir de versões de produção das aplicações, geradas por meio do processo de *build*, utilizando ferramentas como Lighthouse, React Profiler, Web Vitals API e análises de *bundle*. Foram coletadas métricas relacionadas ao tempo de renderização, Largest Contentful Paint (LCP), Interaction to Next Paint (INP), tamanho do *bundle* JavaScript e fluidez de interação.

Além das métricas quantitativas, também foi realizada uma análise técnica e qualitativa das interfaces implementadas, considerando aspectos observáveis como clareza visual, organização dos componentes, responsividade, feedback de interação e complexidade de desenvolvimento. Essa análise não teve como objetivo realizar uma avaliação formal de usabilidade com usuários finais, mas sim complementar os resultados de desempenho com observações técnicas sobre a implementação e o comportamento das bibliotecas no contexto do *dashboard* desenvolvido.

Os resultados obtidos permitiram identificar diferenças relevantes entre Victory Charts, Recharts e ECharts em relação ao desempenho técnico, tamanho do *bundle*, recursos disponíveis, facilidade de implementação e adequação ao contexto do sistema de regulação fundiária da SEAPA-MG. A partir da comparação realizada, são apresentadas recomendações técnicas que podem contribuir para a escolha de bibliotecas de visualização em aplicações web, especialmente em sistemas públicos orientados a dados, nos quais desempenho, clareza visual e manutenção do código são fatores importantes para a evolução da solução.

1.1 Objetivo

Este trabalho tem como objetivo principal desenvolver, implementar e avaliar comparativamente *dashboards* interativos construídas com três *frameworks* e bibliotecas *frontend*, a fim de identificar qual tecnologia oferece o melhor equilíbrio entre desempenho, usabilidade, escalabilidade e clareza visual para a visualização dos dados de Regularização Fundiária da SEAPA-MG.

1.1.1 Objetivos Específicos

Este trabalho possui os seguintes objetivos específicos:

1. Implementar *dashboards* utilizando três bibliotecas/*frameworks* de visualização distintos no ecossistema React, tais como Victory Charts, Recharts e ECharts.

2. Mensurar e comparar o desempenho das soluções com base em métricas objetivas, como tempo de renderização, tempo de carregamento, Time to Interactive (TTI), Largest Contentful Paint (LCP), consumo de memória e tamanho do *bundle*.

3. Documentar e analisar os resultados obtidos, destacando vantagens, limitações e potenciais aplicações de cada solução no contexto de *dashboards* para exibição de dados da Secretaria de Estado de Agricultura, Pecuária e Abastecimento de Minas Gerais (SEAPA-MG).

4. Produzir uma análise c omparativa contendo recomendações técnicas sobre as ferramentas de visualização mais adequadas ao cenário da Secretaria de Estado de Agricultura, Pecuária e Abastecimento de Minas Gerais (SEAPA-MG) e diretrizes para futuros desenvolvimentos de sistemas baseados em dados.

1.2 Justificativa

Essa pesquisa se justifica por contribuir para o avanço do conhecimento sobre avaliação de bibliotecas *frontend* para visualização de dados, um tema altamente relevante nas áreas de Engenharia de Software, Sistemas de Informação e Interação Humano-Computador (IHC). Ao propor e aplicar uma metodologia sistemática de comparação envolvendo desempenho, experiência do usuário e escalabilidade, o estudo fornece um modelo replicável para investigações futuras. Além disso, reforça o diálogo entre teoria e prática ao testar tecnologias contemporâneas amplamente utilizadas no mercado.

No contexto institucional, os resultados podem auxiliar a SEAPA-MG na adoção de soluções mais eficientes para transformar dados brutos em informações visuais acessíveis, favorecendo processos de gestão, transparência pública e tomada de decisão. Para os cidadãos, *dashboards* mais rápidos e claros melhoram o acesso à informação e fortalecem práticas de governança digital. Do ponto de vista profissional, o trabalho aproxima o ambiente acadêmico de demandas reais do governo de Minas Gerais, fomentando inovação aplicada, otimização de recursos e uso estratégico de tecnologias abertas.

1.3 Estrutura do Trabalho

Este trabalho está estruturado da seguinte forma: o Capítulo 2 apresenta o referencial teórico sobre visualização de dados, usabilidade e bibliotecas de gráficos; o Capítulo 3 cita trabalhos relacionados; o Capítulo 4 apresenta a metodologia; o Capítulo 5 o desenvolvimento do *dashboard* e os experimentos realizados; o Capítulo 6 apresenta uma discussão sobre os resultados obtidos; e o Capítulo 7 apresenta as conclusões, contribuições, recomendações para trabalhos futuros e as limitações do estudo.

2 Referencial Teórico

Este capítulo apresenta o referencial teórico que fundamenta o desenvolvimento e a avaliação dos *dashboards* interativos propostos neste trabalho. Inicialmente, são discutidos os conceitos centrais de visualização de dados, destacando sua importância para a interpretação de informações complexas e para o suporte à tomada de decisão, especialmente em contextos governamentais e de gestão pública. Em seguida, são abordados os princípios de Interação Humano-Computador (IHC) e usabilidade, enfatizando modelos e diretrizes amplamente consolidados na literatura, como as heurísticas de Nielsen (1995) e os fundamentos de *design* de interação, que orientam a construção de *interfaces* eficientes, intuitivas e centradas no usuário, conforme discutido por Sharp, Rogers e Preece (2019) e Dix *et al.* (2004).

Na sequência, são discutidos aspectos relacionados ao desempenho em aplicações *frontend*, com foco em métricas técnicas e na influência do tempo de resposta percebida sobre a experiência do usuário. São apresentadas métricas e ferramentas comumente utilizadas na avaliação de aplicações web modernas, especialmente aquelas voltadas à visualização de dados.

Por fim, são avaliadas as bibliotecas de visualização de dados utilizadas no ecossistema React, detalhando suas arquiteturas, princípios de design, componentes principais e capacidades de interação. São exploradas, em especial, as bibliotecas Victory Charts, Recharts e ECharts, que constituem a base tecnológica do estudo comparativo desenvolvido nos capítulos subsequentes.

2.1 Visualização de Dados

A visualização de dados é um campo interdisciplinar que envolve técnicas, ferramentas e princípios destinados a transformar dados brutos em representações visuais capazes de facilitar a interpretação, a descoberta de padrões e a tomada de decisão. Segundo FEW (2014), um *dashboard* eficiente deve apresentar informações de forma clara, objetiva e visualmente organizada, permitindo ao usuário compreender rapidamente o estado de um sistema ou fenômeno analisado. Nesse sentido, gráficos, mapas, indicadores visuais e componentes interativos desempenham um papel fundamental para transmitir informações complexas de modo intuitivo.

A literatura destaca que a visualização não se resume à estética, mas envolve escolhas técnicas relacionadas ao tipo de gráfico, distribuição espacial dos elementos, hierarquia visual e redução de ruído cognitivo. Pentland (2013) reforça que representações

visuais adequadas melhoram tanto a compreensão quanto a tomada de decisões estratégicas em sistemas baseados em dados.

No contexto de *dashboards* governamentais, como os que envolvem dados de regularização fundiária, a visualização permite identificar tendências, inconsistências e regiões críticas, oferecendo suporte essencial para políticas públicas. Trabalhos recentes, como Arruda e Mesquita (2024), Rosa e Pestana (2025), demonstram que *dashboards* interativos têm se tornado ferramentas de grande relevância em ambientes com grande volume de dados.

2.2 Interação Humano Computador (IHC) e Usabilidade

A usabilidade é um elemento central para o sucesso de qualquer sistema interativo. A área de Interação Humano-Computador (IHC) analisa como pessoas interagem com softwares e dispositivos, buscando criar interfaces mais eficientes, intuitivas e acessíveis. De acordo com Sharp, Rogers e Preece (2019), a qualidade da experiência do usuário depende de fatores como facilidade de aprendizado, eficiência de uso, satisfação subjetiva e ausência de erros críticos durante a navegação.

Um dos modelos mais utilizados para avaliação da usabilidade é o conjunto de Heurísticas de Nielsen (NIELSEN, 1995), que corresponde a dez princípios gerais para o desenvolvimento e avaliação de interfaces:

1. visibilidade do estado do sistema;
2. correspondência entre o sistema e o mundo real;
3. controle e liberdade do usuário;
4. consistência e padronização;
5. prevenção de erros;
6. reconhecimento em vez de memorização;
7. flexibilidade e eficiência de uso;
8. estética e design minimalista;
9. auxílio no reconhecimento, diagnóstico e recuperação de erros;
10. ajuda e documentação.

No contexto deste trabalho, algumas heurísticas receberam maior destaque durante a análise das implementações desenvolvidas. A heurística de visibilidade do estado do sistema

foi considerada relevante devido à necessidade de fornecer feedback visual claro durante interações com os gráficos e carregamento das informações. A heurística de consistência e padronização também teve papel importante, especialmente na manutenção de equivalência visual entre as bibliotecas analisadas. Além disso, os princípios de reconhecimento em vez de memorização e estética e design minimalista foram considerados essenciais para garantir clareza visual, organização das informações e redução da carga cognitiva durante a interpretação dos dados apresentados nos *dashboards*.

Dix *et al.* (2004) destacam que interfaces de visualização devem reduzir a carga cognitiva do usuário, evitando sobrecarga informacional e priorizando elementos significativos para o contexto. Em *dashboards*, isso envolve a seleção adequada de gráficos, o cuidado com cores e contrastes e a disposição espacial dos elementos visualizados, garantindo que a interação seja fluida e eficaz.

2.3 Desempenho em Aplicações *Frontend*

O desempenho é um componente essencial para a experiência do usuário, especialmente em aplicações baseadas em visualização de dados, que tendem a exigir maior processamento gráfico. De acordo com Alves (2025, p. 2), "a performance do front-end impacta diretamente a experiência do usuário, influenciando a eficiência de carregamento, a estabilidade visual e a responsividade das aplicações web".

No ecossistema do desenvolvimento web moderno, métricas de desempenho são utilizadas para avaliar diferentes aspectos da experiência percebida pelo usuário durante o carregamento e uso de uma aplicação. Entre as métricas apontadas por Chopra (2021) métricas, destaca-se o Largest Contentful Paint (LCP), que mede o tempo necessário para que o maior elemento visível da página seja carregado na tela. Essa métrica está relacionada à percepção de carregamento inicial, pois indica em quanto tempo o conteúdo principal da interface se torna visível ao usuário.

Outra métrica relevante é o Time to Interactive (TTI), que indica o tempo necessário para que a página se torne totalmente interativa, ou seja, capaz de responder de forma adequada às ações do usuário. Em aplicações com muitos componentes, gráficos ou bibliotecas externas, um TTI elevado pode indicar que a interface aparenta estar carregada, mas ainda não responde corretamente às interações.

O Cumulative Layout Shift (CLS), por sua vez, mede a estabilidade visual da página, identificando deslocamentos inesperados de elementos durante o carregamento. Em um *dashboard*, valores elevados de CLS podem prejudicar a experiência de uso, pois gráficos, filtros ou cartões informativos podem mudar de posição de forma inesperada, dificultando a leitura e a interação.

Além dessas métricas, o Interaction to Next Pain (INP) avalia a responsividade da interface após interações do usuário, como cliques, seleções ou acionamento de filtros. Essa métrica mede o tempo entre a interação realizada e a próxima atualização visual exibida pelo navegador, sendo importante para avaliar a fluidez da aplicação durante o uso.

Ferramentas como Lighthouse, React Profiler, Web Vitals API e DevTools possibilitam medir objetivamente esses indicadores e comparar diferentes abordagens de implementação. No contexto deste trabalho, essas ferramentas foram utilizadas para avaliar o comportamento das bibliotecas analisadas em relação ao carregamento, renderização e resposta da interface.

Em aplicações de *dashboards*, o desempenho também está relacionado à complexidade dos gráficos, ao volume de dados e à otimização interna das bibliotecas utilizadas. Bibliotecas mal otimizadas ou excessivamente pesadas podem gerar redução da taxa de quadros por segundo, conhecida como Frames Per Second (FPS), travamentos, *re-renders* desnecessários e maior consumo de memória. O FPS indica a quantidade de quadros exibidos por segundo na interface; valores baixos podem tornar animações, transições e interações menos fluidas.

Outro aspecto relevante é o tamanho do *bundle*, que corresponde ao conjunto de arquivos gerados após o processo de *build* da aplicação, incluindo código JavaScript, folhas de estilo, recursos estáticos e bibliotecas utilizadas. Quanto maior o *bundle*, maior tende a ser o tempo necessário para download, interpretação e execução dos arquivos pelo navegador, o que pode impactar diretamente o carregamento inicial da aplicação.

2.4 Bibliotecas de Visualização em React

Diversas bibliotecas de gráficos têm sido utilizadas para construir *dashboards* modernos, especialmente em aplicações React. Cada uma oferece características distintas de desempenho, flexibilidade e complexidade. A seguir são apresentadas três delas, que são consideradas nos experimentos realizados neste trabalho: Victory Charts, Recharts e Echarts.

2.4.1 Victory Charts

A biblioteca Victory Charts é reconhecida pela integração nativa com React e pela facilidade de customização. Sua abordagem declarativa possibilita controlar a renderização dos componentes de maneira direta, embora possa apresentar limitações de desempenho para grandes volumes de dados (NEARFORM, 2025b).

2.4.1.1 Arquitetura e Princípios de Design

Victory Charts implementa uma arquitetura modular baseada em componentes React, de tal forma que cada elemento visual (eixos, séries, legendas, grades) corresponde a um componente independente, que pode ser combinado livremente. Esse modelo favorece a customização granular sem sacrificar a consistência visual, pois os componentes compartilham sistemas comuns de coordenadas, escalas e temas. A biblioteca adota uma filosofia de design baseada em configurações padrão sensatas, que podem ser totalmente sobrescritas conforme a necessidade do desenvolvedor (NEARFORM, 2025b), e oferece configurações padrão que funcionam imediatamente, mas permite a sobreposição completa de qualquer aspecto do comportamento ou aparência .

Um dos aspectos mais distintos da arquitetura do Victory Chart é seu sistema de temas unificado, que assegura coerência estética em diferentes tipos de gráficos dentro de um mesmo *dashboard*. Quando um tema é aplicado ao componente Victory Chart, ele é automaticamente propagado para todos os componentes filhos (`VictoryAxis`, `VictoryLine`, `VictoryBar` etc.), mantendo paletas de cores, estilos de tipografia e propriedades de espaçamento harmoniosas. Essa abordagem reduz significativamente o esforço de desenvolvimento necessário para criar visualizações coesas (NEARFORM, 2025a).

2.4.1.2 Componentes Principais e sua Aplicação

A biblioteca oferece um conjunto abrangente de componentes especializados, cada um responsável por um aspecto específico da visualização:

VictoryChart - Componente contêiner que estabelece o contexto compartilhado para todos os elementos gráficos. Define propriedades fundamentais como `domain` (intervalo dos dados), `scale` (sistema de coordenadas), `padding` (margens internas) e `theme` (tema visual). Sua função primordial é orquestrar a interoperabilidade entre os demais componentes, garantindo que todos compartilhem as mesmas referências espaciais.

Componentes de Dados - `VictoryLine` (linhas), `VictoryBar` (barras), `VictoryArea` (áreas), `VictoryScatter` (pontos dispersos), `VictoryPie` (pizza) e `VictoryHistogram` (histogramas) representam os principais tipos de visualização disponíveis. Cada um aceita dados por meio da `prop data`[1] e oferecem propriedades específicas para sua modalidade (ex: `barWidth` para `VictoryBar`, `interpolation` para `VictoryLine`).

Componentes de Estrutura - `VictoryAxis` define os eixos cartesianos, com propriedades especializadas para os eixos dependentes (`dependentAxis`) e independentes, permitindo organizar visualmente os dados e facilitar sua interpretação. Já o `VictoryGrid` adiciona linhas de grade que auxiliam na leitura de valores e comparação entre catego-

[1] A `prop data` corresponde à propriedade responsável por fornecer os dados utilizados na renderização do gráfico. Esses dados são estruturados em listas ou objetos contendo valores, categorias e demais informações necessárias para a construção da visualização.

rias, reduzindo o esforço cognitivo do usuário durante a análise visual. O componente `VictoryLegend` implementa legendas interativas que contribuem para identificação rápida das informações apresentadas, enquanto o `VictoryLabel` possibilita a inserção de textos customizados em diferentes regiões do gráfico, favorecendo maior contextualização e clareza na apresentação dos dados.

Componentes de Interação - O `VictoryTooltip` exibe informações detalhadas ao passar o cursor sobre elementos gráficos, permitindo ao usuário acessar valores específicos sem sobrecarregar visualmente a interface. Já os componentes `VictorySelectionContainer` e `VictoryZoomContainer` habilitam interações avançadas, como seleção de regiões e aplicação de *zoom*, recursos importantes em *dashboards* analíticos exploratórios, especialmente em cenários com grande volume de dados ou necessidade de análise detalhada (NEARFORM, 2025c). Essas funcionalidades contribuem para uma experiência mais dinâmica e interativa, alinhando-se aos princípios de usabilidade e exploração visual discutidos na literatura de IHC.

2.4.1.3 Suporte a Visualizações Avançadas

Para além dos gráficos cartesianos tradicionais, Victory Charts oferece suporte robusto a visualizações polares e radiais, a partir do componente `VictoryPolarAxis` e da propriedade polar do VictoryChart. Essa capacidade permite a criação de gráficos radares e visualizações angulares que são particularmente úteis para comparações multidimensionais. A propriedade `innerRadius` possibilita a criação de gráficos de rosca (*donut charts*), enquanto `startAngle` e `endAngle` controlam o arco de exibição para criar visualizações parciais.

Para cenários que exigem visualizações multivariadas ou em camadas, Victory oferece componentes composicionais como `VictoryGroup` (para agrupar múltiplas séries), `VictoryStack` (para gráficos de barras ou áreas empilhadas) e `VictoryErrorBar` (para representar intervalos de confiança ou margens de erro). Essa flexibilidade composicional é uma das características mais poderosas da biblioteca, permitindo a construção de visualizações complexas por meio da combinação de elementos simples.

2.4.2 Recharts

A biblioteca Recharts é amplamente adotada em *dashboards* corporativos. A Recharts utiliza D3.js internamente e oferece componentes robustos para gráficos comuns. É conhecida por sua simplicidade e bom desempenho em cenários moderados, sendo frequentemente usada em aplicações que exigem atualizações em tempo real (RECHARTS..., 2025).

2.4.2.1 Arquitetura e estrutura de design

O Recharts é uma biblioteca de visualização de dados construída especificamente para o ecossistema React, caracterizando-se por sua abordagem componentizada e declarativa que se integra naturalmente com o fluxo de desenvolvimento React moderno. Diferente de outras bibliotecas que impõem uma estrutura rígida, o Recharts adota uma estrutura de gráficos compostos por componentes, em que cada elemento do gráfico é um componente React independente, que pode ser combinado, configurado e reutilizado de forma flexível. Essa arquitetura alinha-se perfeitamente com o paradigma do React, permitindo que desenvolvedores construam visualizações complexas a partir da composição de componentes simples e bem definidos.

Um dos princípios fundamentais do Recharts é seu compromisso com a simplicidade e acessibilidade. A biblioteca foi projetada para ser intuitiva para desenvolvedores familiarizados com React, minimizando a curva de aprendizado por meio de uma API que espelha os padrões já estabelecidos no ecossistema. Ao mesmo tempo, oferece capacidades avançadas disponibilizando propriedades e componentes especializados, seguindo a estrutura do React de "composição sobre configuração". Essa abordagem resulta em código mais legível, manutenível e alinhado com as práticas modernas de desenvolvimento *frontend*.

O Recharts implementa um sistema de renderização baseado em Scalable Vector Grafics (SVG) nativo, aproveitando as vantagens de escalabilidade, nitidez e interatividade que essa tecnologia oferece. Diferente de soluções baseadas em Canvas, a abordagem SVG permite que cada elemento do gráfico seja manipulado individualmente no Document Object Model (DOM), facilitando a implementação de interações complexas, animações suaves e estilização Cascading Style Sheets (CSS). Essa escolha arquitetural posiciona o Recharts como uma solução ideal para *dashboards* que exigem alta interatividade e responsividade.

2.4.2.2 Componentes Principais e Padrões de Composição

O Recharts organiza seus componentes em uma hierarquia lógica que reflete a estrutura típica de um gráfico:

- <`RechartsSurface`> - Componente fundamental que define a área de renderização e estabelece o contexto de coordenadas para todos os elementos filhos. Funciona como o contêiner raiz que orquestra o sistema de escalas, transformações e eventos.
- <`ChartContainer`> - Componente de alto nível que encapsula a lógica comum de configuração de gráficos, incluindo dimensionamento responsivo, manipulação de redimensionamento da janela e propagação de propriedades para componentes filhos.

- <`ResponsiveContainer`> - Componente essencial para *dashboards* modernos, garante que os gráficos se adaptem dinamicamente a diferentes tamanhos de tela e dispositivos. Implementa um *wrapper* que calcula proporções e dimensões baseadas no contêiner pai, seguindo princípios de design responsivo.

O Recharts oferece um conjunto abrangente de componentes para diferentes tipos de visualização:

- `LineChart` / `Line` - Para visualização de séries temporais e tendências. Suporta múltiplas linhas com estilos distintos, marcadores personalizáveis e diferentes algoritmos de interpolação.
- `BarChart` / `Bar` - Para comparações categóricas e distribuições. Oferece controle granular sobre largura das barras, espaçamento, cantos arredondados e padrões de preenchimento.
- `AreaChart` / `Area` - Para representação de volumes acumulados e séries com contexto quantitativo. Permite gradientes de preenchimento, transparências e múltiplas áreas sobrepostas ou empilhadas.
- `PieChart` / `Pie` - Para visualização de proporções e composições. Suporta gráficos de *donut*, rótulos adaptativos, destaque de segmentos e animações de entrada.
- `ScatterChart` / `Scatter` - Para análise de correlações e identificação de *outliers*. Oferece símbolos personalizáveis, tamanhos variáveis baseados em dados e agrupamento visual.
- `RadarChart` / `Radar` - Para comparações multidimensionais em sistemas de coordenadas polares. Permite visualização de perfis e análise de múltiplas métricas simultaneamente.

2.4.2.3 Sistema de Estilização e Temas

O Recharts implementa um sistema de estilização flexível e consistente que opera em três níveis distintos:

Estilização por Propriedades cada componente aceita propriedades de estilo que seguem uma convenção consistente:

1. `stroke` - Cor de contorno para linhas e bordas;
2. fill - Cor de preenchimento para áreas e formas sólidas;
3. strokeWidth - Espessura de linhas e contornos;

4. strokeDasharray - Padrão de traços para linhas tracejadas;

Sistema de Temas para garantir consistência visual em *dashboards* com múltiplos gráficos:

Temas predefinidos - Inclui temas como `light`, `dark` e `colorblind` otimizados para diferentes contextos.

Temas Customizáveis - Permite definir paletas de cores, famílias de fontes e estilos padrão que se propagam por todos os componentes.

Provider de Tema - Componente <`RechartsThemeProvider`> que aplica configurações de tema globalmente.

Estilização Condicional para criar visualizações reativas que respondem a dados ou estado:

Funções como Propriedades - Aceita funções que recebem dados e retornam estilos, permitindo formatação condicional baseada em valores.

Seletores de Estado - Suporte a estados como `active`, `selected` e `hovered` com estilos diferenciados.

Gradientes e Padrões- Suporte avançado a gradientes SVG e padrões de preenchimento para enriquecimento visual.

2.4.3 ECharts

Dentre as soluções mais poderosas em termos de desempenho, ECharts se destaca por ser altamente otimizado, suportando animações complexas e grandes volumes de dados. Sua versão para React permite integrações eficientes com escalabilidade visual, conforme apontado por Chopra (2021) no estudo sobre *frameworks* de visualização para *dashboards*.

2.4.3.1 Estratégias de Integração: *Wrapper* Oficial vs. Componente Personalizado

A integração do Apache ECharts, uma biblioteca JavaScript imperativa, dentro do paradigma declarativo do React requer uma camada de adaptação. Existem duas estratégias predominantes para essa integração, cada uma com seus próprios *trade-offs* em termos de facilidade de uso, controle e manutenção.

A primeira e mais direta estratégia é a adoção do pacote `echarts-for-react`, descrito em sua documentação como o *wrapper* React mais simples e melhor para o Apache ECharts. Esse componente pronto para uso abstrai a complexidade do ciclo de vida do ECharts, oferecendo uma interface baseada em *props*[2]. Sua utilização básica é bastante

[2] Props (abreviação de "properties" ou propriedades) são o principal mecanismo do React para passar dados de um componente pai para um componente filho. São imutáveis (*read-only*) dentro do componente

simplificada: após a instalação das dependências (`echarts e echarts-for-react`), o desenvolvedor importa o componente ReactECharts e passa um objeto de configuração `option` para renderizar o gráfico. O pacote oferece uma Application Programming Interface (API) completa, incluindo suporte a temas, eventos (via ***`prop onEvents`***), estado de carregamento e redimensionamento automático (ECHARTS-FOR-REACT..., 2025; BAO; LI; ZHOU, 2020; APACHE FOUNDATION, 2025; JSDOCS..., 2025).

Contudo, a documentação oficial e os estudos técnicos da comunidade de desenvolvimento apontam que, em alguns contextos, pode haver problemas com o redimensionamento responsivo do gráfico ou preocupações quanto à manutenção ativa da biblioteca (VAIBHAV, 2023; JSDOCS, 2025). Como alternativa, a segunda estratégia ganha relevância: implementar um componente *wrapper* personalizado. Essa abordagem, detalhada em tutoriais como os de Vaibhav (2023), proporciona um controle fino sobre a instância do ECharts. A estrutura central envolve o uso de um `useRef` para armazenar a referência do elemento DOM container, a inicialização do gráfico dentro de um `useEffect` com `echarts.init`, e a atualização via `setOption` em resposta a mudanças nos dados ou configurações. A principal vantagem reside na eliminação de uma dependência externa e na capacidade de customizar extensivamente o comportamento do componente — como a implementação de um padrão de projeto *Observer* (GAMMA *et al.*, 1994) para redimensionamento (ResizeObserver) ou a gestão otimizada de eventos — para atender necessidades específicas do *dashboard*.

A decisão entre uma estratégia e outra deve considerar o escopo do projeto, a necessidade de controle e a expertise da equipe. O `echarts-for-react` oferece velocidade no desenvolvimento, enquanto o *wrapper* personalizado oferece máxima flexibilidade e otimização.

2.4.3.2 Arquitetura de Configuração e Principais Recursos

Independentemente da estratégia de integração escolhida, o coração da visualização com ECharts é o objeto `option`. Esse objeto segue uma arquitetura declarativa e altamente modular, de tal forma que todas as propriedades do gráfico, desde eixos e séries até *tooltips* e legendas, são centralmente configuradas. Por exemplo, um gráfico de linhas básico é definido por uma `option` contendo configurações para `xAxis`, `yAxis` e uma series do tipo '`line`'.

A biblioteca se destaca por uma série de recursos avançados essenciais para *dashboards* profissionais:

- Múltiplos Tipos de Gráfico - Suporta mais de 20 tipos, como linha, barra, dispersão (*scatter*), pizza e radar, que podem ser combinados livremente em um único sistema

que as recebe e funcionam como os argumentos de uma função.

de coordenadas.

- Interatividade Nativa - A configuração de eventos como `click` ou `legend selectchanged` por meio da `prop onEvents` (no *wrapper* oficial) ou do método `chart.on()` (no *wrapper* personalizado) permite criar *dashboards* exploratórios de tal forma que as ações do usuário podem filtrar dados ou acionar *callbacks*.
- Extensibilidade - O ecossistema do ECharts inclui extensões oficiais, como `echarts-liquidfill` para gráficos de preenchimento líquido e `wordcloud`, que podem ser importadas e registradas para estender as capacidades da biblioteca.

3 Trabalhos Relacionados

A construção de *dashboards* interativos e a comparação entre ferramentas de visualização de dados têm sido temas recorrentes na literatura, especialmente em contextos envolvendo dados públicos, microdados educacionais e informações científicas de grande complexidade. Os trabalhos analisados neste capítulo apresentam diferentes abordagens para a construção de visualizações e oferecem fundamentos importantes para o desenvolvimento do presente estudo.

3.1 Estudo Comparativo no Projeto VODAN BR

Rosa e Pestana (2025) realizaram um estudo comparativo entre ferramentas de visualização de dados com o objetivo de identificar a solução mais adequada para a criação de um *dashboard* destinado ao projeto VODAN BR, o qual integra a iniciativa internacional Virus Outbreak Data Network. O projeto lida com dados clínicos complexos e anonimizados relacionados à COVID-19, provenientes de múltiplas instituições de saúde.

Rosa (2025) avaliou cinco plataformas de *Business Intelligence* (BI), termo utilizado para se referir a ferramentas voltadas à coleta, organização, análise e visualização de dados para apoio à tomada de decisão. As plataformas analisadas foram FVIDI, Apache Superset, Metabase, Redash e Looker Studio. O Apache Superset, o Metabase e o Redash são ferramentas voltadas à criação de painéis analíticos e visualizações de dados a partir de diferentes fontes, enquanto o Looker Studio é uma solução do Google voltada à construção de relatórios e *dashboards* interativos.

A avaliação considerou critérios como clareza visual, facilidade de uso, flexibilidade das visualizações e compatibilidade com dados semânticos. Dados semânticos são dados estruturados de forma a explicitar relações e significados entre as informações, permitindo melhor interpretação por sistemas computacionais. Nesse contexto, o estudo utilizou como referência tecnologias como RDF e SPARQL. O Resource Description Framework (RDF) é um modelo utilizado para representar dados na Web de forma estruturada por meio de relações entre entidades, enquanto SPARQL é uma linguagem de consulta utilizada para recuperar e manipular dados armazenados nesse formato.

A metodologia do estudo envolveu a definição de critérios de avaliação, a análise comparativa entre as plataformas e o desenvolvimento de um protótipo. O trabalho concluiu que o Looker Studio foi a ferramenta mais adequada ao contexto do VODAN BR, uma rede brasileira associada à iniciativa *Virus Outbreak Data Network*, voltada à organização e disponibilização de dados relacionados à saúde. Segundo o estudo, a escolha do Looker

Studio ocorreu devido à sua boa integração com fontes de dados, facilidade de uso e opções avançadas de visualização. Esse trabalho se relaciona com a presente pesquisa por também realizar uma comparação entre ferramentas de visualização de dados em um contexto aplicado, embora o foco deste TCC esteja em bibliotecas *frontend* específicas do ecossistema React.

O trabalho de Rosa e Pestana (2025) destaca a importância da avaliação comparativa de ferramentas para visualização de dados, especialmente em contextos que envolvem grandes volumes de informações e necessidade de integração entre diferentes fontes de dados. Entretanto, o estudo concentra-se na análise de plataformas de Business Intelligence (BI), e não em *frameworks frontend* voltados ao desenvolvimento em React, o que limita sua aplicabilidade em cenários focados em desempenho técnico, renderização de componentes e experiência do usuário em aplicações web interativas.

3.2 Ferramentas de Baixo Custo para *dashboards* Públicos

Arruda e Mesquita (2024) apresentam uma abordagem voltada para a criação de *dashboards* utilizando ferramentas de baixo custo e de implementação acessível, com base nos microdados educacionais do Instituto Nacional de Estudos e Pesquisas Educacionais Anísio Teixeira (INEP), especificamente relacionados ao Censo da Educação Superior (CENSUP). O INEP é o órgão responsável por produzir e divulgar estatísticas, indicadores e avaliações educacionais no Brasil, enquanto o CENSUP é uma pesquisa estatística realizada anualmente sobre instituições de educação superior, cursos, alunos e docentes. Os microdados, por sua vez, correspondem a conjuntos detalhados de informações disponibilizadas para análise, permitindo estudos mais aprofundados sobre os dados educacionais.

O estudo discute alternativas *open source*, isto é, ferramentas de código aberto que podem ser utilizadas, estudadas, modificadas e distribuídas de forma mais acessível, para lidar com grandes volumes de dados. Essa discussão ressalta as limitações financeiras e estruturais enfrentadas por instituições públicas e universidades, que muitas vezes precisam adotar soluções com menor custo de implantação e manutenção.

A metodologia envolveu três principais componentes:

- tratamento de dados em Python (Pandas) para padronização e integração;
- uso de Google Sheets e Google Apps Script para disponibilização dos dados e construção de APIs acessíveis;
- desenvolvimento de um *dashboard* em React, utilizando ApexCharts, TailwindCSS e Material-UI, hospedado na plataforma Vercel;

Os resultados demonstraram que é possível construir *dashboards* funcionais, acessíveis e escaláveis com ferramentas gratuitas, sendo uma alternativa viável a plataformas de BI pagas. O foco do estudo, no entanto, encontra-se mais na viabilidade econômica e menos na análise de desempenho técnico das bibliotecas utilizadas.

3.3 *ECharts: A Declarative Framework for Rapid Construction of Web-Based Visualization*

O artigo de Bao, Li e Zhou (2020) discute o uso de um *framework* declarativo voltado à construção rápida de visualizações web com alto desempenho. Os autores descrevem recursos como renderização otimizada, uso de Canvas, suporte a animações e grande variedade de gráficos complexos, destacando a eficiência da biblioteca para aplicações de grande escala.

A pesquisa evidencia vantagens do modelo declarativo e demonstra como ECharts reduz o esforço de desenvolvimento e aumenta o desempenho comparado a bibliotecas imperativas. Apesar disso, o estudo analisa apenas o ECharts isoladamente, sem compará-lo com outras bibliotecas em um mesmo contexto, como será feito neste TCC com React, Victory Charts, Recharts e ECharts.

3.4 *Concept-Annotated Examples for Library Comparison*

O estudo de Elmqvist *et al.* (2022) apresenta uma abordagem baseada em exemplos anotados com conceitos visuais para comparar diferentes bibliotecas de visualização. O método proposto busca superar comparações superficiais, permitindo análise de expressividade, flexibilidade e limitações estruturais de cada biblioteca.

Os autores destacam diferenças claras entre modelos declarativos e imperativos, como Vega-Lite *versus* D3.js, apontando como cada paradigma influencia a construção de visualizações interativas. Esse tipo de análise contribui para comparações mais rigorosas entre *frameworks*. No entanto, o trabalho mantém foco em características conceituais das bibliotecas, sem avaliar desempenho técnico ou aplicações reais, enquanto o presente TCC incorpora métricas empíricas de desempenho e usabilidade.

3.5 O impacto da performance de *frontend* na experiência do usuário

Alves (2025) analisou como o desempenho de aplicações web influencia diretamente a experiência do usuário, utilizando métricas Web Vitals amplamente adotadas como LCP,

CLS, TTI, INP. O trabalho avaliou sites com diferentes níveis de otimização utilizando ferramentas como Lighthouse, PageSpeed Insights e Chrome DevTools.

Os resultados demonstram que otimizações como compressão de imagens, *lazy-loading*, minificação, cache e uso de Content Delivery Networks (CDNs) exercem impacto significativo na percepção de fluidez e estabilidade visual. A pesquisa reforça que desempenho é parte essencial da experiência do usuário.

Esse estudo é altamente relevante para o presente TCC, pois fundamenta a importância das métricas de desempenho adotadas na etapa de avaliação comparativa entre bibliotecas de visualização.

De modo geral, os trabalhos relacionados apresentados contribuem para a fundamentação deste trabalho ao evidenciarem a relevância da visualização de dados, da construção de *dashboards*, da comparação entre ferramentas e da avaliação de desempenho em aplicações *web*. O estudo de Rosa e Pestana (2025) aproxima-se deste trabalho ao realizar uma comparação entre ferramentas de visualização em um contexto aplicado, embora esteja voltado a plataformas de *Business Intelligence* (BI). O trabalho de Arruda e Mesquita (2024) contribui ao demonstrar a viabilidade de soluções acessíveis para criação de *dashboards* em instituições públicas, enquanto o estudo de Bao, Li e Zhou (2020) evidencia características técnicas de uma biblioteca específica de visualização. Já a abordagem de Elmqvist *et al.* reforça a importância de comparações estruturadas entre bibliotecas, e o trabalho de Alves (2025) fundamenta a análise de desempenho *frontend* por meio de métricas objetivas.

Diferentemente desses estudos, o presente trabalho concentra-se na comparação prática entre bibliotecas de visualização do ecossistema React, especificamente Victory Charts, Recharts e ECharts, aplicadas a um mesmo *dashboard* no contexto da SEAPA-MG. Assim, a principal contribuição deste TCC está em avaliar essas bibliotecas sob as mesmas condições de implementação, dados e ambiente experimental, considerando tempo de renderização, métricas de desempenho, tamanho do *bundle*, recursos disponíveis, complexidade de desenvolvimento e aspectos técnicos observáveis da interação.

4 Metodologia

A metodologia adotada neste trabalho foi planejada para integrar práticas de desenvolvimento *frontend*, avaliação comparativa de desempenho e análise de usabilidade, fundamentando-se nos princípios da visualização de dados, da Interação Humano-Computador (IHC) e da Engenharia de Software. A estrutura metodológica segue uma abordagem aplicada e comparativa, combinando elementos qualitativos e quantitativos, de modo a responder à questão central da pesquisa.

4.1 Questão de Pesquisa

A questão que orienta o presente estudo é:

> *"Quais frameworks e bibliotecas frontend oferecem melhor equilíbrio entre desempenho técnico, fluidez de interação e escalabilidade para a criação de dashboards interativos voltados à visualização de dados da regularização fundiária da SEAPA-MG?"*

Essa pergunta direciona o desenvolvimento, a análise e a comparação das diferentes soluções implementadas para determinar quais tecnologias demonstram o melhor equilíbrio entre desempenho técnico, facilidade de uso e eficiência visual.

4.2 Natureza e Abordagem Metodológica

A pesquisa possui natureza aplicada, pois busca desenvolver um produto funcional, um *dashboard* interativo, para o uso no contexto real da SEAPA-MG. Sua abordagem é predominantemente qualitativa e exploratória, uma vez que envolve interpretação de critérios de usabilidade, design e percepção do usuário.

No entanto, também incorpora elementos quantitativos, especialmente na mensuração de métricas objetivas, como tempo de renderização, carregamento e consumo de recursos. Além disso, utiliza o método comparativo, fundamental para avaliar diferentes bibliotecas *frontend* na implementação de um mesmo conjunto de visualizações, conforme sugerido por autores como Sharp, Rogers e Preece (2019) e Dix *et al.* (2004).

4.3 Etapas da Metodologia

A metodologia está organizada em cinco etapas principais: (1) revisão bibliográfica; (2) definição de critérios; (3) implementação; (4) avaliação comparativa e (5) análise dos resultados.

4.3.1 Etapa 1 — Levantamento e Revisão Bibliográfica

Esta etapa tem como objetivo investigar e sistematizar os fundamentos teóricos sobre visualização de dados, usabilidade, Interação Humano-Computador (IHC) e desempenho em aplicações *frontend*, fornecendo embasamento conceitual para o desenvolvimento e avaliação dos *dashboards* propostos neste trabalho.

Nesse sentido, a revisão bibliográfica foi conduzida por meio de uma busca sistemática em livros clássicos da área e em bases de dados científicas reconhecidas, com foco em publicações nacionais e internacionais relevantes para os temas do estudo. As principais bases consultadas foram:

- Google Scholar;
- IEEE Xplore;
- ACM Digital Library;
- ScienceDirect.

As buscas foram realizadas entre os meses de setembro e outubro de 2025, utilizando combinações das seguintes palavras-chave, em português e inglês:

- “visualização de dados” / “data visualization”;
- “dashboards interativos”;
- “Interação Humano-Computador” / “Human-Computer Interaction”;
- “usabilidade” / “usability”;
- “desempenho de aplicações web” / “front-end performance”;
- “React dashboards” e “data visualization libraries”.

A busca inicial retornou aproximadamente 80 publicações. Em uma primeira etapa, foi realizada a leitura dos títulos e resumos, com o objetivo de excluir trabalhos fora do escopo do estudo, duplicados ou excessivamente focados em aspectos não relacionados à

visualização, usabilidade ou desempenho. Após essa triagem inicial, cerca de 35 trabalhos foram selecionados para leitura exploratória.

Na etapa seguinte, foram priorizados estudos que apresentassem aplicações práticas, análises comparativas, avaliações de desempenho ou discussões conceituais consolidadas sobre visualização de dados e IHC. Ao final do processo, aproximadamente 15 referências foram consideradas centrais para a fundamentação teórica do trabalho.

Entre os principais referenciais teóricos adotados destacam-se:

- Sharp, Rogers e Preece (2019), que fundamenta os princípios de design de interação e experiência do usuário;
- Nielsen (1995), Dix *et al.* (2004), referências clássicas sobre conceitos e métodos de Interação Humano-Computador;
- FEW (2014) e Pentland (2013), que abordam a comunicação visual, o papel dos *dashboards* e a compreensão de dados por meio de representações visuais;
- Trabalhos recentes sobre *dashboards* e visualização aplicada, como Rosa e Pestana (2025), Arruda e Mesquita (2024), Alves (2025) e Chopra (2021), que fornecem subsídios técnicos e metodológicos alinhados ao contexto deste estudo.

Como resultado desta etapa, obteve-se uma base teórica consolidada que orienta tanto as decisões de projeto do *dashboard* quanto a definição dos critérios técnicos e de usabilidade utilizados nas etapas posteriores da metodologia.

4.3.2 Etapa 2 — Definição dos Critérios de Avaliação

Esta etapa tem como objetivo estabelecer critérios qualitativos e quantitativos para comparar diferentes bibliotecas de visualização.

Nesse sentido, foram definidos critérios baseados na literatura e em Web Vitals[1]:

- tempo de renderização dos gráficos;
- Time to Interactive (TTI);
- tamanho do *bundle* e uso de memória;
- fluidez de interação e responsividade;

[1] Web Vitals é um conjunto de métricas propostas pelo Google para avaliar a experiência do usuário em aplicações web, considerando aspectos relacionados ao desempenho, responsividade e estabilidade visual da interface. Entre as principais métricas estão Largest Contentful Paint (LCP), Interaction to Next Paint (INP) e Cumulative Layout Shift (CLS).

- análise qualitativa das interfaces com base em princípios de usabilidade e heurísticas de Nielsen (NIELSEN, 1995).

4.3.3 Etapa 3 — Implementação do *Dashboard*

Esta etapa tem como objetivo desenvolver *dashboards* interativos em React utilizando diferentes bibliotecas de visualização. Nesse sentido, foram realizados:

- Desenvolvimento do *frontend* em React, seguindo práticas modernas da biblioteca.
- Implementação inicial com a biblioteca Victory Charts, devido à sua integração nativa com React.
- Criação de implementações alternativas com Recharts e ECharts, ambas amplamente utilizadas em *dashboards* profissionais.
- Uso de dados anonimizados do SEAPA-MG em formato JSON.

Esta etapa tem como resultado protótipo funcional construído em três variações, permitindo comparações diretas de desempenho e usabilidade.

4.3.4 Etapa 4 — Avaliação de Desempenho

Esta etapa tem como objetivo mensurar o desempenho técnico das aplicações e analisar qualitativamente características relacionadas à interação, clareza visual e experiência de uso das interfaces implementadas.

Para isso, foram utilizadas ferramentas de análise de desempenho e métodos de avaliação interpretativa fundamentados nos princípios de Interação Humano-Computador (IHC), conforme descrito a seguir:

- Uso das ferramentas Lighthouse, React Profiler, Web Vitals API e *Memory Snapshots* do DevTools;
- Execução de testes controlados de *benchmarking*;

Como resultado desta etapa, foi produzido um relatório comparativo contendo métricas objetivas e análises qualitativas capazes de evidenciar vantagens, limitações e diferenças técnicas entre as bibliotecas analisadas.

4.3.5 Etapa 5 — Discussão dos Resultados e Conclusões

Esta etapa tem como objetivo interpretar os achados e identificar a solução mais adequada para o contexto da SEAPA-MG.

Nesse sentido foi realizada uma análise integrada dos dados quantitativos e qualitativos obtidos nas etapas anteriores, correlacionando desempenho, usabilidade e escalabilidade.

Como resultado a produção de um documento final contendo recomendações técnicas e diretrizes práticas para o desenvolvimento de *dashboards* eficientes no setor público.

5 Implementação e Experimentos

Este capítulo apresenta o processo de implementação do *dashboard*, bem como os experimentos realizados para avaliar o desempenho, usabilidade e eficiência das bibliotecas de visualização analisadas. São descritos o ambiente experimental, as estratégias adotadas na construção da aplicação, os critérios de medição e a análise dos resultados obtidos.

5.1 Ambiente Experimental

Os experimentos foram conduzidos em ambiente local, utilizando um computador com sistema operacional Linux (XUbuntu), processador Intel Core i5 de 10ª geração e 8GB de memória RAM. O navegador utilizado para a coleta das métricas foi o Opera, com ferramentas de desenvolvimento integradas, como Lighthouse e Performance Panel.

Para garantir maior controle sobre os testes, foi utilizado um *backend* local fixo, responsável por fornecer os dados dos gráficos por meio de rotas padronizadas. Essa abordagem permitiu que todas as implementações compartilhassem exatamente as mesmas fontes de dados, garantindo a comparabilidade dos resultados.

Além disso, foi realizado o *build* da aplicação utilizando o Vite[4] em modo de produção, sendo os testes executados sobre os arquivos gerados na pasta `dist`, de modo a refletir o comportamento real da aplicação em ambiente de produção.

5.2 Implementação do *Dashboard*

O *dashboard* foi desenvolvido utilizando React, seguindo uma arquitetura baseada em componentes reutilizáveis. A estrutura da aplicação foi mantida constante entre os experimentos, sendo alterada apenas a biblioteca responsável pela renderização dos gráficos.

A interface foi composta por:

- um conjunto de filtros para manipulação dos dados;
- um gráfico de pizza;
- três gráficos de barras;

[4] Vite é uma ferramenta de desenvolvimento *frontend* voltada para aplicações *web* modernas. Desenvolvida com foco em desempenho, utiliza módulos ES, sigla para *ECMAScript*, que é o padrão oficial que define a linguagem JavaScript. Esses módulos permitem organizar o código em arquivos reutilizáveis, por meio de comandos como `import` e `export`. Durante o desenvolvimento, o Vite utiliza esses módulos nativos do navegador e, no processo de *build*, aplica ferramentas de empacotamento otimizadas, proporcionando inicialização rápida e menor tempo de atualização da aplicação.

Durante a implementação, buscou-se manter a consistência visual entre as diferentes bibliotecas, garantindo:

- mesma disposição dos gráficos em tela;
- Uso de *tooltips* em todos os gráficos;
- legendas customizadas;
- mesma estrutura de dados para alimentação dos componentes.

5.3 Base de Dados para Testes

Devido à limitação de volume de dados disponíveis no sistema original da SEAPA-MG, foi necessário realizar a criação de uma base de dados local para simulação dos cenários de uso.

Para isso, foi populado um banco de dados com mais de 10.000 registros, distribuídos de forma a alimentar todos os gráficos do *dashboard*. Essa estratégia permitiu a realização de testes mais consistentes, especialmente na análise de desempenho e renderização.

5.4 Coleta de Métricas

A coleta de métricas foi realizada utilizando diferentes abordagens, com o objetivo de capturar tanto aspectos técnicos quanto perceptivos da aplicação.

5.4.1 Tempo de Renderização

O tempo de renderização dos gráficos foi medido diretamente no código, utilizando a API `performance.now()`, que permite obter medições em milissegundos com alta precisão.

O procedimento consistiu em registrar o tempo imediatamente antes da renderização do gráfico e calcular o intervalo de tempo até a aplicação das configurações visuais.

5.4.2 Métricas de Performance (*Lighthouse*)

Foram utilizadas métricas baseadas em *Web Vitals*, obtidas por meio da ferramenta *Lighthouse*, incluindo:

- *Largest Contentful Paint* (LCP);
- *First Contentful Paint* (FCP);
- *Total Blocking Time* (TBT);

- *Cumulative Layout Shift* (CLS).

5.4.3 Interação e Responsividade

A fluidez da interação foi avaliada utilizando a métrica *Interaction to Next Paint* (INP), disponível no painel de performance do navegador. Essa métrica representa o tempo de resposta da interface após uma interação do usuário.

5.4.4 Tamanho do *bundle*

O tamanho do *bundle* foi obtido utilizando a ferramenta *Rollup Visualizer*, considerando o nó raiz da aplicação, que representa o total de código carregado pelo navegador.

5.5 Resultados

Esta seção apresenta os resultados obtidos a partir da implementação e avaliação comparativa das bibliotecas de visualização analisadas neste trabalho. Os experimentos foram conduzidos utilizando versões equivalentes do *dashboard* desenvolvidas com Victory Charts, Recharts e ECharts, mantendo a mesma estrutura de dados, organização visual e regras de negócio, com o objetivo de garantir maior consistência na comparação.

As análises são baseadas em métricas relacionadas ao desempenho técnico, fluidez de interação, clareza visual e aspectos qualitativos da implementação das interfaces, incluindo tempo de renderização, métricas Web Vitals e características observáveis de interação. Os resultados apresentados buscam evidenciar diferenças de comportamento entre as bibliotecas, permitindo identificar vantagens, limitações e possíveis impactos no desenvolvimento de *dashboards* interativos para aplicações governamentais orientadas a dados.

5.5.1 Tempo de Renderização

Foram realizadas 30 execuções para cada implementação analisada, buscando reduzir interferências ocasionais do ambiente de execução e aumentar a confiabilidade estatística dos resultados. Para os gráficos de barras, considerando que a *dashboard* possuía três componentes distintos desse tipo, os valores obtidos foram agrupados para cálculo da média geral e do desvio padrão. Os resultados são apresentados considerando média aritmética e dispersão dos dados.

Observa-se que a biblioteca Victory apresentou os menores tempos médios de renderização entre as bibliotecas avaliadas, tanto para gráficos de pizza quanto para gráficos de barras, indicando melhor desempenho bruto de renderização nos cenários analisados.

Tabela 1 – Tempo médio de renderização dos gráficos

Biblioteca	Tipo de Gráfico	Tempo Médio (ms)	Desvio Padrão (ms)
Recharts	Pizza	73,06	6,36
Recharts	Barras	73,18	6,61
Victory	Pizza	13,81	4,75
Victory	Barras	13,08	4,53
ECharts	Pizza	39,22	3,15
ECharts	Barras	16,78	8,48

Fonte: Produzido pelo autor.

O ECharts apresentou desempenho de 16,78 ms, próximo ao de menor valor de 13,81 do Victory Charts nos gráficos de barras, embora tenha sido identificada uma variação significativa entre os tempos obtidos nos diferentes componentes da *dashboard*. O primeiro gráfico de barras apresentou tempo médio de aproximadamente 25,7 ms, enquanto os outros dois gráficos registraram tempos significativamente menores, em torno de 7,1 ms e 7,4 ms, respectivamente.

Essa diferença pode estar relacionada à complexidade distinta entre os gráficos, quantidade de elementos renderizados, configurações visuais específicas ou ao volume de informações exibidas em cada componente. Dessa forma, os resultados indicam que o desempenho do ECharts pode variar de acordo com o contexto de utilização e nível de complexidade da visualização implementada.

5.5.2 Métricas de desempenho

A Tabela 2 apresenta os resultados obtidos via Lighthouse os quais indicaram o desempenho para todas as bibliotecas em ambiente simulado de produção.

Tabela 2 – Comparação das métricas de desempenho em tempo real

Biblioteca	LCP (s)	INP (ms)
Recharts	0.45	80
Victory	0.50	88
ECharts	0.22	48

Fonte: Produzido pelo autor.

Os resultados apresentados na Tabela 2 indicam que o ECharts apresentou melhor desempenho nas métricas de carregamento e interação, com menores valores de LCP e INP. Por outro lado, as bibliotecas Recharts e Victory apresentaram desempenho semelhante, com valores muito superiores, quase duas vezes maior. Esse comportamento pode estar relacionado às estratégias internas de renderização e otimização adotadas pelo ECharts,

que utiliza mecanismos gráficos altamente otimizados e estruturas voltadas a cenários com grande quantidade de interações e elementos visuais.

Embora a biblioteca Victory Charts tenha apresentado os menores tempos médios de renderização, observou-se que métricas relacionadas à interação e responsividade percebida, como o INP, favoreceram o ECharts. Esse resultado sugere que desempenho bruto de renderização e fluidez percebida pelo usuário não necessariamente apresentam comportamento equivalente, especialmente em aplicações com forte componente interativo.

5.5.3 Tamanho do *Bundle*

Com o objetivo de analisar o impacto das bibliotecas no tamanho final da aplicação, foram realizadas medições do *bundle* gerado em ambiente de produção após o processo de *build*. A análise considerou tanto o tamanho total renderizado quanto o tamanho compactado utilizando compressão *gzip*, permitindo avaliar o impacto potencial no carregamento inicial da aplicação e na transferência de dados pela rede.

A Tabela 3 apresenta os resultados obtidos para cada uma das bibliotecas analisadas. A biblioteca Victory apresentou o menor tamanho de *bundle*, enquanto o ECharts apresentou o maior, devido à sua maior complexidade e conjunto de funcionalidades.

Tabela 3 – Comparação do tamanho do *bundle*

Biblioteca	**Renderizado**	**Gzip**
Recharts	10.11 MB	2.87 MB
Victory	9.81 MB	2.78 MB
ECharts	12.06 MB	3.37 MB

Fonte: Produzido pelo autor.

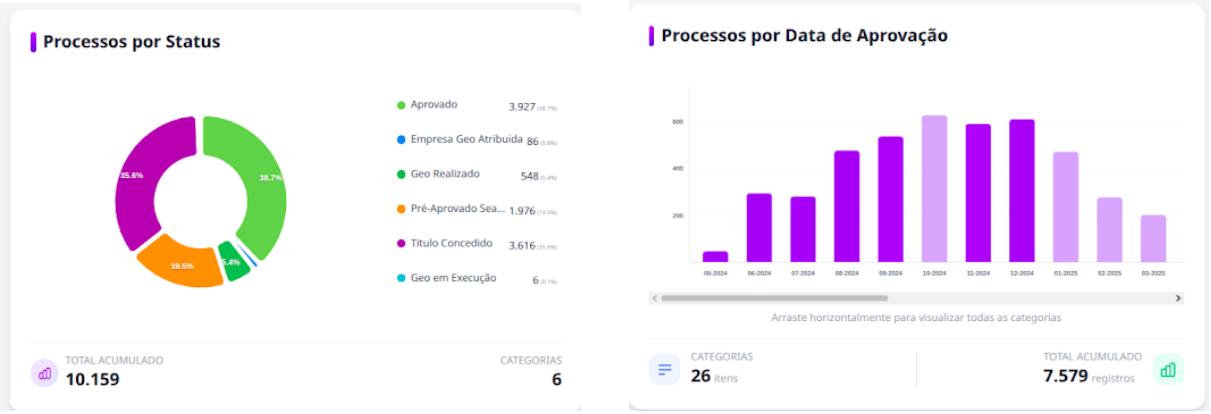
5.5.4 Análise Comparativa das Bibliotecas de Visualização

Com o objetivo de garantir maior consistência na comparação entre as bibliotecas analisadas, as implementações desenvolvidas mantiveram a mesma estrutura visual, organização espacial dos componentes, conjunto de dados e regras de negócio. Dessa forma, as diferenças observadas concentram-se principalmente nos aspectos técnicos relacionados à implementação, desempenho e recursos oferecidos por cada biblioteca.

As Figuras 1, 2 e 3 apresentam exemplos dos gráficos implementados utilizando as bibliotecas Victory Charts, Recharts e ECharts, respectivamente. As implementações buscaram manter equivalência visual e estrutural entre os gráficos, permitindo que a comparação fosse realizada com foco nos aspectos técnicos das bibliotecas analisadas.

Durante o desenvolvimento das implementações, observou-se que a biblioteca Recharts apresentou maior simplicidade estrutural e menor verbosidade de código, utilizando

Figura 1 – Implementação dos gráficos utilizando Victory Charts



(a) Gráfico de pizza (b) Gráfico de barras

Fonte: Produzido pelo autor.

Figura 2 – Implementação dos gráficos utilizando Recharts

(a) Gráfico de pizza (b) Gráfico de barras

Fonte: Produzido pelo autor.

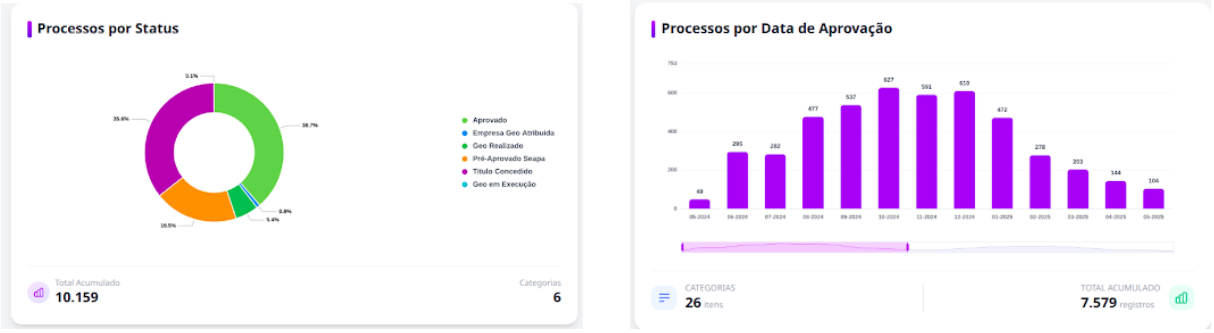
uma abordagem declarativa fortemente integrada ao ecossistema React. Recursos como legendas, responsividade e *tooltips* puderam ser implementados diretamente por meio da composição de componentes, favorecendo a legibilidade e manutenção do código-fonte.

A abordagem declarativa da Recharts pode ser observada no Código 5.1, no qual os elementos do gráfico são estruturados diretamente como componentes React independentes, reduzindo a necessidade de configurações imperativas adicionais.

Código 5.1 – Estrutura declarativa utilizando Recharts

```
<ResponsiveContainer width="100%" height="120%">
  <BarChart data={data}>
    <XAxis dataKey="name" />
    <YAxis />
    <Tooltip />
```

Figura 3 – Implementação dos gráficos utilizando ECharts



(a) Gráfico de pizza (b) Gráfico de barras

Fonte: Produzido pelo autor.

```
    <Bar dataKey="value" />
  </BarChart>
</ResponsiveContainer>
```

Esse comportamento pode estar relacionado à forte integração declarativa da biblioteca com o ecossistema React, favorecendo reutilização de componentes e reduzindo a necessidade de configurações imperativas adicionais.

A biblioteca Victory Charts apresentou uma arquitetura modular baseada em componentes independentes, oferecendo elevada flexibilidade para customização das visualizações. Entretanto, determinadas funcionalidades exigiram maior nível de configuração manual para reprodução de comportamentos interativos e ajustes de layout.

A modularidade da Victory Charts pode ser observada no Código 5.2, no qual diferentes componentes independentes são utilizados para configuração detalhada dos elementos visuais do gráfico, permitindo elevado nível de customização.

Código 5.2 – Customização modular utilizando Victory Charts

```
<VictoryAxis
  tickFormat={(tick) => formatTickLabel(tick)}
  tickLabelComponent={
    <VictoryLabel
      angle={0}
      textAnchor="middle"
      verticalAnchor="start"
    />
  }
/>
```

Já a biblioteca ECharts apresentou maior quantidade de recursos nativos voltados à construção de *dashboards* analíticos, incluindo suporte avançado a animações, interações,

efeitos visuais e mecanismos de renderização otimizados.

Em contrapartida, a implementação exigiu maior complexidade estrutural e maior verbosidade de configuração, principalmente devido ao uso extensivo do objeto `option` para definição dos gráficos. Essa abordagem pode ser observada no Código 5.3, no qual diferentes configurações são centralizadas em um único objeto de definição.

Código 5.3 – Configuração baseada em objeto utilizando ECharts

```
const option: echarts.EChartsOption = {
  tooltip: {
    trigger: "axis"
  },
  dataZoom: [
    {
      type: "inside"
    }
  ],
  xAxis: {
    type: "category"
  },
  yAxis: {
    type: "value"
  },
  series: [
    {
      type: "bar"
    }
  ]
};
```

Essa estrutura oferece elevado nível de controle sobre os elementos gráficos e interações, porém aumenta a verbosidade e complexidade estrutural da implementação. Apesar dessas vantagens, o ECharts apresentou maior tamanho de *bundle* e maior complexidade estrutural durante o desenvolvimento, indicando um *trade-off* entre desempenho, flexibilidade e simplicidade de implementação.

A Tabela 4 apresenta uma síntese comparativa das principais características observadas durante o desenvolvimento das implementações.

Os resultados observados indicam que a escolha da biblioteca depende diretamente do contexto de utilização da aplicação. A Recharts favorece simplicidade e produtividade durante o desenvolvimento, enquanto a ECharts oferece maior robustez para cenários analíticos mais complexos e ricos em interações. Já a Victory Charts apresenta uma abordagem intermediária, equilibrando modularidade, desempenho e flexibilidade de customização.

Tabela 4 – Comparação técnica entre as bibliotecas analisadas

Critério	**Victory**	**Recharts**	**ECharts**
Facilidade de configuração	Média	Alta	Baixa
Simplicidade do código	Média	Alta	Baixa
Suporte nativo a interações	Médio	Médio	Alto
Customização visual	Alta	Média	Alta
Suporte a animações	Médio	Médio	Alto
Facilidade de implementação	Média	Alta	Média
Recursos analíticos avançados	Médio	Baixo	Alto

Fonte: Produzido pelo autor.

5.6 Considerações Finais do Capítulo

Os experimentos realizados demonstram que não existe uma solução única ideal para todos os cenários. A escolha da biblioteca de visualização deve considerar o contexto da aplicação, o volume de dados, os requisitos de desempenho e a experiência do usuário.

Os resultados obtidos neste capítulo servirão de base para a discussão dos resultados para as conclusões finais do trabalho, apresentadas no próximo capítulo.

6 Discussão dos Resultados

Este capítulo tem como objetivo analisar criticamente os resultados obtidos nos experimentos, relacionando-os com o referencial teórico apresentado e com a questão de pesquisa proposta. A discussão considera principalmente o desempenho técnico das bibliotecas, o comportamento interativo observado nas interfaces implementadas, a complexidade de desenvolvimento e a adequação das soluções ao contexto do sistema de regularização fundiária da Secretaria de Estado de Agricultura, Pecuária e Abastecimento de Minas Gerais (SEAPA-MG).

6.1 Análise Geral dos Resultados

Para sintetizar os resultados obtidos ao longo dos experimentos, apresenta-se a Tabela 5, que consolida as principais métricas avaliadas.

Tabela 5 – Matriz comparativa final das bibliotecas de visualização

Critério	**Victory Charts**	**Recharts**	**ECharts**
Tempo médio de renderização (ms)	**13,45**	73,12	28
LCP / INP	0,50 s / 88 ms	0,45 s / 80 ms	**0,22 s / 48 ms**
Tamanho do *bundle* (MB - gzip)	**2,78**	2,87	3,37
Recursos de interação e visualização	Intermediários	Intermediários	**Avançados**
Complexidade de implementação	Média	**Baixa**	Alta

Fonte: Produzido pelo autor.

A Tabela 5 consolida os principais resultados obtidos ao longo dos experimentos, permitindo uma visão integrada das características de cada biblioteca. Observa-se que a biblioteca Victory Charts apresentou o menor tempo médio de renderização e o menor tamanho de *bundle*, indicando maior eficiência em métricas técnicas de baixo nível. Esse comportamento sugere que a biblioteca possui menor sobrecarga estrutural, sendo adequada para cenários em que leveza e simplicidade sejam fatores prioritários.

A biblioteca Recharts, por sua vez, destacou-se pela menor complexidade de implementação. Sua abordagem baseada em componentes declarativos favoreceu a leitura do código, a manutenção e a construção dos gráficos com menor quantidade de configuração.

Entretanto, nos testes realizados, apresentou maior tempo médio de renderização em comparação com as demais bibliotecas, o que indica menor desempenho bruto no cenário analisado.

Já a biblioteca ECharts apresentou os melhores resultados nas métricas LCP e INP, além de oferecer maior quantidade de recursos nativos para interação, configuração e construção de visualizações analíticas. Apesar de apresentar maior tamanho de *bundle* e maior complexidade de configuração, seus resultados indicam melhor comportamento em termos de carregamento percebido, resposta da interface e suporte a cenários mais complexos.

Esses resultados reforçam a existência de um *trade-off* entre desempenho bruto de renderização, tamanho final da aplicação, recursos disponíveis e complexidade de desenvolvimento. Dessa forma, a escolha da biblioteca não deve ser feita a partir de uma única métrica isolada, mas sim considerando o contexto da aplicação, o volume de dados, os requisitos de interação e as necessidades futuras de manutenção e expansão do sistema.

6.2 Desempenho Técnico e Comportamento Interativo

Os resultados das métricas de desempenho indicam que a biblioteca ECharts apresentou melhor comportamento em termos de carregamento e resposta da interface, com menores valores de LCP e INP. O LCP está relacionado ao tempo necessário para que o principal conteúdo visual da tela seja carregado, enquanto o INP mede a resposta da aplicação após interações do usuário. Assim, os resultados obtidos sugerem que o ECharts proporcionou uma interface mais responsiva no cenário experimental analisado.

Esse comportamento pode estar relacionado ao modelo de renderização utilizado pela biblioteca, baseado em *canvas*, que tende a reduzir a manipulação direta do DOM e favorecer maior eficiência em cenários com gráficos mais complexos ou maior volume de elementos visuais. Tal resultado está alinhado às discussões de Alves (2025), que destaca que o desempenho do *front-end* impacta diretamente a responsividade, a estabilidade visual e a fluidez de aplicações *web*.

Por outro lado, a biblioteca Victory Charts apresentou o menor tempo médio de renderização e o menor tamanho de *bundle*, indicando uma solução mais leve do ponto de vista computacional. No entanto, embora esses resultados sejam positivos, eles não indicam, isoladamente, que a biblioteca seja a mais adequada para todos os cenários. Em aplicações de *dashboards*, especialmente aquelas voltadas à análise de dados institucionais, também é necessário considerar recursos de interação, capacidade de customização, escalabilidade das visualizações e facilidade de expansão futura.

A Recharts apresentou maior tempo médio de renderização nos experimentos, mas

demonstrou boa simplicidade estrutural. Sua principal vantagem está na integração declarativa com React, permitindo que gráficos sejam implementados por meio da composição de componentes. Dessa forma, mesmo não apresentando os melhores resultados técnicos de desempenho, a biblioteca se mostra interessante em cenários nos quais produtividade, legibilidade do código e manutenção sejam prioridades.

6.3 Análise Técnica das Interfaces e Implementações

A análise das interfaces implementadas foi conduzida de forma técnica e interpretativa, considerando aspectos observáveis como organização visual dos componentes, resposta das interações, clareza das informações apresentadas e complexidade de implementação. Essa análise não teve como objetivo realizar uma avaliação formal de usabilidade com usuários finais, mas sim complementar as métricas quantitativas com observações sobre o comportamento das bibliotecas durante o desenvolvimento e uso das interfaces.

Nesse sentido, a Recharts apresentou maior concisão de código e menor complexidade de configuração. A construção dos gráficos por meio de componentes React tornou a implementação mais direta, favorecendo a legibilidade e a manutenção. Esse aspecto é relevante em projetos nos quais a equipe de desenvolvimento precisa alterar, expandir ou adaptar os gráficos com frequência.

A Victory Charts apresentou boa flexibilidade, mas exigiu maior quantidade de ajustes manuais em determinados pontos, especialmente em elementos como rótulos, eixos, legendas e interações. Embora tenha apresentado bons resultados técnicos de desempenho, sua implementação pode demandar maior esforço em cenários que exigem comportamentos interativos mais elaborados.

O ECharts, por sua vez, apresentou maior complexidade estrutural devido ao uso extensivo do objeto `option`, responsável por concentrar as configurações dos gráficos. Apesar disso, essa mesma estrutura oferece uma quantidade maior de recursos nativos, incluindo interações avançadas, ferramentas de navegação, animações, mecanismos de zoom, customização de *tooltips* e suporte a diferentes tipos de visualizações. Essa característica torna a biblioteca mais adequada para *dashboards* analíticos que exigem maior riqueza de recursos e potencial de expansão.

6.4 *Trade-off* entre Desempenho, Interatividade e Complexidade

Os resultados obtidos evidenciam a existência de um *trade-off* entre desempenho técnico, recursos de interação e complexidade de implementação. Bibliotecas mais leves, como a Victory Charts, tendem a apresentar menor tempo de renderização e menor

tamanho de *bundle*, porém podem exigir maior esforço manual para alcançar determinados comportamentos visuais e interativos.

A Recharts posiciona-se como uma solução de implementação mais simples, com boa integração ao ecossistema React e menor verbosidade em diversos cenários. No entanto, nos experimentos realizados, apresentou desempenho inferior em relação ao tempo médio de renderização, o que pode limitar sua adequação em *dashboards* com maior volume de dados ou maior quantidade de elementos gráficos.

Já o ECharts apresentou maior custo inicial em termos de tamanho de *bundle* e complexidade de configuração, mas compensou esse custo com melhor desempenho nas métricas LCP e INP, além de maior quantidade de recursos nativos para construção de interfaces analíticas. Em um sistema institucional, no qual a tendência é que novas visualizações, filtros e formas de análise sejam incorporados ao longo do tempo, essa maior robustez pode representar uma vantagem significativa.

Dessa forma, os resultados indicam que cada biblioteca apresenta características mais adequadas para diferentes cenários de utilização. A Victory Charts pode ser indicada para *dashboards* mais leves e customizáveis. A Recharts pode ser adequada para projetos com foco em produtividade e simplicidade de manutenção. O ECharts, por sua vez, mostra-se mais apropriado para *dashboards* complexos, interativos e com maior demanda por recursos analíticos.

6.4.1 Limitações do Estudo

Entre as principais limitações deste estudo, destaca-se a ausência de avaliações formais com usuários finais. Dessa forma, as observações qualitativas realizadas não devem ser interpretadas como uma avaliação formal de usabilidade, mas sim como uma análise técnica das interfaces implementadas, baseada em aspectos observáveis de interação, organização visual e comportamento dos componentes.

Além disso, os experimentos foram realizados em ambiente local controlado, utilizando uma configuração específica de hardware e navegador, o que pode influenciar parcialmente os resultados obtidos. Em outros ambientes, com diferentes dispositivos, navegadores, condições de rede ou infraestrutura de hospedagem, os resultados podem apresentar variações.

Outra limitação refere-se ao escopo das implementações analisadas, que envolveram apenas um modelo específico de *dashboard* desenvolvido no ecossistema React. Dessa forma, os resultados observados podem variar em cenários com diferentes arquiteturas, volumes de dados, tipos de gráficos, regras de negócio ou requisitos de interação.

Apesar dessas limitações, os resultados obtidos oferecem evidências relevantes sobre o comportamento das bibliotecas analisadas em um cenário aplicado de visualização de

dados, contribuindo para decisões técnicas relacionadas ao desenvolvimento de *dashboards* interativos em sistemas institucionais.

6.5 Resposta à Questão de Pesquisa

A questão de pesquisa deste trabalho buscou identificar quais bibliotecas *front-end* oferecem melhor equilíbrio entre desempenho técnico, fluidez de interação e escalabilidade para a criação de *dashboards* interativos voltados à visualização dos dados da regularização fundiária da Secretaria de Estado de Agricultura, Pecuária e Abastecimento de Minas Gerais (SEAPA-MG).

Com base nos resultados obtidos, conclui-se que a biblioteca ECharts é a solução mais indicada para o contexto da SEAPA-MG. Embora a Victory Charts tenha apresentado melhor desempenho bruto de renderização e menor tamanho de *bundle*, o ECharts demonstrou melhor equilíbrio para um cenário institucional que exige visualizações interativas, possibilidade de expansão, maior variedade de recursos gráficos e bom desempenho percebido durante o uso da interface.

A escolha do ECharts justifica-se principalmente pelos melhores resultados nas métricas LCP e INP, que indicam melhor comportamento em relação ao carregamento percebido e à resposta da interface após interações. Além disso, a biblioteca oferece recursos nativos mais avançados para construção de *dashboards* analíticos, como suporte a zoom, filtros visuais, animações, personalização de *tooltips*, diferentes tipos de séries e maior controle sobre os elementos gráficos.

No contexto da SEAPA-MG, em que o sistema pode evoluir para contemplar novos indicadores, filtros, relatórios e formas de análise dos processos de regularização fundiária, a robustez e a flexibilidade do ECharts tornam-se fatores relevantes. Ainda que seu maior tamanho de *bundle* represente uma desvantagem, esse impacto pode ser mitigado futuramente com estratégias como *lazy loading*, *code splitting* e otimização do carregamento dos componentes.

Dessa forma, a resposta à questão de pesquisa indica o ECharts como a biblioteca mais adequada ao cenário analisado, por oferecer o melhor equilíbrio entre desempenho percebido, recursos de interação, escalabilidade e capacidade de adaptação às necessidades futuras do sistema da Secretaria de Estado de Agricultura, Pecuária e Abastecimento de Minas Gerais (SEAPA-MG). A Victory Charts permanece como alternativa relevante para cenários mais leves e com maior restrição de tamanho de aplicação, enquanto a Recharts se destaca em contextos nos quais simplicidade de implementação e manutenção sejam os critérios prioritários.

7 Conclusão

Este trabalho teve como objetivo realizar uma avaliação comparativa de bibliotecas *front-end* para a construção de *dashboards* interativos, considerando critérios de desempenho técnico, fluidez de interação, tamanho do *bundle*, complexidade de implementação e recursos disponíveis no contexto do sistema de regularização fundiária da Secretaria de Estado de Agricultura, Pecuária e Abastecimento de Minas Gerais (SEAPA-MG).

A partir dos experimentos realizados, foi possível observar que não existe uma biblioteca que se destaque de forma absoluta em todos os critérios analisados. Os resultados evidenciaram diferenças relevantes entre as abordagens, especialmente no que diz respeito ao tempo de renderização, ao carregamento percebido, à resposta da interface, ao tamanho final da aplicação e à complexidade exigida durante o desenvolvimento.

A biblioteca Victory Charts destacou-se pelo menor tempo médio de renderização e pelo menor tamanho de *bundle*, evidenciando boa eficiência computacional. Esses resultados indicam que a biblioteca pode ser adequada para aplicações que priorizam leveza, simplicidade estrutural e menor custo inicial de carregamento. No entanto, durante a implementação, observou-se que determinados recursos de interação e customização exigem maior configuração manual, o que pode aumentar o esforço de desenvolvimento em *dashboards* mais complexos.

A Recharts, por sua vez, apresentou como principal vantagem a simplicidade de implementação. Sua abordagem baseada em componentes declarativos no ecossistema React favorece a legibilidade do código, a manutenção e a produtividade no desenvolvimento. Apesar disso, nos experimentos realizados, a biblioteca apresentou tempos médios de renderização superiores aos das demais soluções, o que pode limitar sua adequação em cenários com maior volume de dados ou maior exigência de desempenho.

Já a biblioteca ECharts apresentou os melhores resultados nas métricas relacionadas ao carregamento percebido e à resposta da interface, como Largest Contentful Paint (LCP) e Interaction to Next Paint (INP). Além disso, demonstrou maior variedade de recursos nativos para construção de visualizações analíticas, como suporte a interações avançadas, animações, zoom, personalização de *tooltips* e maior controle sobre os elementos gráficos. Por outro lado, apresentou maior tamanho de *bundle* e maior complexidade de configuração, principalmente devido ao uso extensivo do objeto `option`.

Esses resultados reforçam a existência de um *trade-off* entre desempenho bruto de renderização, carregamento inicial, recursos de interação e complexidade de desenvolvimento. Assim, a escolha da tecnologia mais adequada deve considerar o contexto específico da aplicação, os requisitos do sistema, o volume de dados, a necessidade de expansão

futura e o perfil da equipe responsável pela manutenção.

No contexto da SEAPA-MG, conclui-se que o ECharts se apresenta como a biblioteca mais indicada entre as soluções analisadas. Embora a Victory Charts tenha apresentado melhor desempenho bruto de renderização e menor tamanho de *bundle*, o ECharts demonstrou melhor equilíbrio para um sistema institucional que pode evoluir para contemplar novos indicadores, filtros, relatórios e formas de análise dos processos de regularização fundiária. Sua robustez, variedade de recursos e melhor comportamento nas métricas LCP e INP tornam a biblioteca uma alternativa mais adequada para *dashboards* analíticos, interativos e com potencial de expansão.

Um dos principais desafios enfrentados durante o desenvolvimento deste trabalho foi a limitação de dados disponíveis no ambiente produtivo da SEAPA-MG. A baixa quantidade de registros reais impossibilitou a realização de testes mais robustos em cenários de grande volume de dados, o que motivou a criação de uma base de dados local para simulação. Embora essa estratégia tenha permitido a execução dos experimentos, os resultados obtidos podem não refletir completamente o comportamento das bibliotecas em ambientes reais de alta escala.

Além disso, destaca-se que os experimentos foram realizados em ambiente local controlado, utilizando uma configuração específica de hardware, sistema operacional e navegador. Dessa forma, o desempenho observado pode variar em cenários com maior complexidade de dados, maior número de usuários simultâneos, diferentes condições de infraestrutura, dispositivos menos robustos ou ambientes reais de hospedagem.

Como contribuição, este trabalho apresenta uma análise comparativa aplicada entre bibliotecas de visualização de dados no ecossistema React, fornecendo subsídios técnicos para a escolha de ferramentas em aplicações governamentais e sistemas baseados em dados. A comparação realizada considerou métricas quantitativas de desempenho e aspectos técnicos observáveis das implementações, contribuindo para decisões relacionadas à construção de *dashboards* interativos.

Como trabalhos futuros, sugere-se:

- Realizar experimentos com bases de dados reais em maior escala, refletindo cenários mais próximos do ambiente de produção;
- Avaliar o comportamento das bibliotecas em ambientes com múltiplos usuários simultâneos, considerando aspectos de concorrência, escalabilidade e infraestrutura;
- Investigar o impacto de diferentes estratégias de otimização, como *lazy loading*, divisão de código (*code splitting*) e uso de Content Delivery Networks (CDNs);
- Expandir a análise para outras bibliotecas de visualização e *frameworks* emergentes,

permitindo uma comparação mais ampla entre soluções disponíveis no ecossistema *front-end*;

- Realizar estudos com usuários finais, aplicando métodos formais de avaliação de usabilidade, como testes de tarefas, entrevistas e questionários de experiência do usuário, de modo a complementar a análise técnica realizada neste trabalho.

Por fim, conclui-se que a escolha de bibliotecas de visualização de dados deve ser orientada por uma análise multidimensional, que considere desempenho técnico, recursos disponíveis, facilidade de manutenção, complexidade de implementação e necessidades específicas do contexto de uso. No caso analisado, o ECharts mostrou-se a alternativa mais adequada para apoiar a evolução dos *dashboards* do sistema de regularização fundiária da SEAPA-MG, enquanto Victory Charts e Recharts permanecem como opções relevantes para cenários com prioridades distintas.

Referências

AGÊNCIA SP. *Governo de São Paulo usa dados de satélites para acelerar respostas ambientais e aprimorar planejamento territorial*. 2024. Disponível em: <https://www.agenciasp.sp.gov.br/governo-de-sao-paulo-usa-dados-de-satelites-para-acelerar-respostas-ambientais-e-aprimorar-planejamento-territorial/>. Acesso em: 10 jun. 2025.

ALVES, A. A. d. S. *O impacto da performance de front-end na experiência do usuário: análise técnica e recomendações para otimização*. Dissertação (Mestrado) — FATEC São José do Rio Preto, São José do Rio Preto, 2025. Disponível em: <https://ric.cps.sp.gov.br/handle/123456789/37404>. Acesso em: 12 set. 2025.

APACHE FOUNDATION. *Apache ECharts – Options Documentation*. [S.l.], 2025. Disponível em: <https://echarts.apache.org/en/option.html>. Acesso em: 16 set. 2025.

APACHE SOFTWARE FOUNDATION. *Apache ECharts: Documentation and API Reference*. [S.l.], 2025. Disponível em: <https://echarts.apache.org/en/option.html>. Acesso em: 15 set. 2025.

ARRUDA, L. I. d.; MESQUITA, M. P. Ferramentas de baixo custo para integração e geração de dashboards de visualização de dados. In: *Anais da 13ª Escola Regional de Informática de Mato Grosso*. [S.l.]: Sociedade Brasileira de Computação, 2024. p. 183–185.

BAO, J.; LI, S.; ZHOU, Y. Echarts: A declarative framework for rapid construction of web-based visualization. *Journal of Visual Languages and Computing*, 2020.

BRASIL. *Gestão detalha princípios, objetivos e iniciativas da nova Estratégia Federal de Governo Digital*. 2024. Disponível em: <https://www.gov.br/governodigital/pt-br/noticias/gestao-detalha-principios-objetivos-e-iniciativas-da-nova-estrategia-federal-de-governo-digital>. Acesso em: 10 jun. 2025.

CHOPRA, E. P. Creating live dashboards for data visualization: Flask vs. react. *TIJER - International Research Journal*, v. 8, n. 9, p. 1–10, 2021. Disponível em: <http://www.tijer.org>.

DATA Visualization in React – Complete Guide. 2025. Disponível em: <https://www.ignek.com/blog/data-visualization-in-react/>. Acesso em: 16 set. 2025.

DIX, A. et al. *Human-Computer Interaction*. 3. ed. Harlow: Pearson, 2004.

ECHARTS-FOR-REACT – NPM Package. 2025. Disponível em: <https://www.npmjs.com/package/echarts-for-react>. Acesso em: 16 set. 2025.

ELMQVIST, N. et al. Concept-annotated examples for library comparison. *IEEE VIS Conference Proceedings*, 2022.

FEW, S. *Information Dashboard Design: Displaying Data for At-a-Glance Monitoring*. 2. ed. Burlingame: Analytics Press, 2014. ISBN 978-1-937737-00-9.

JSDOCS. *echarts-for-react API Documentation*. [S.l.], 2025. Disponível em: <https://www.jsdocs.io/package/echarts-for-react>. Acesso em: 15 set. 2025.

JSDOCS – echarts-for-react. 2025. Disponível em: <https://www.jsdocs.io/package/echarts-for-react>. Acesso em: 16 set. 2025.

MACHADO, M.; CARVALHO, R.; SILVA, F. Explorando, comparando e coordenando múltiplas fontes de dados em uma ferramenta de visualização de informação. In: *Anais do Simpósio Brasileiro de Sistemas de Informação (SBSI)*. [S.l.: s.n.], 2022.

NEARFORM. *Victory Chart API Documentation*. [S.l.], 2025. Disponível em: <https://commerce.nearform.com/open-source/victory/docs/api/victory-chart/>. Acesso em: 14 set. 2025.

NEARFORM. *Victory Charts Documentation – Introduction*. [S.l.], 2025. Disponível em: <https://commerce.nearform.com/open-source/victory/docs/introduction/>. Acesso em: 14 set. 2025.

NEARFORM. *Victory Native Documentation*. [S.l.], 2025. Disponível em: <https://commerce.nearform.com/open-source/victory-native/>. Acesso em: 14 set. 2025.

NIELSEN, J. *Usability Engineering*. Boston: Academic Press, 1995. ISBN 978-0-12-518405-2.

NPM. *echarts-for-react: A Simple React Wrapper for Apache ECharts*. [S.l.], 2025. Disponível em: <https://www.npmjs.com/package/echarts-for-react>. Acesso em: 15 set. 2025.

PENTLAND, A. *Social Physics: How Good Ideas Spread—The Lessons from a New Science*. New York: Penguin Press, 2013. ISBN 978-1-59420-565-1.

RECHARTS – Redefined chart library built with React. 2025. Disponível em: <https://recharts.org/>. Acesso em: 16 set. 2025.

ROSA; PESTANA, F. *Estudo comparativo de ferramentas de visualização e construção de dashboard para dados do VODAN BR*. Dissertação (Mestrado) — Universidade Federal do Rio de Janeiro, Rio de Janeiro, 2025. Disponível em: <http://hdl.handle.net/11422/26982>.

SECRETARIA DO PLANEJAMENTO E GESTÃO DO ESTADO DO CEARÀ. *Seplag desenvolve BI para monitorar benefícios sociais concedidos pelo Governo do Ceará*. 2025. Disponível em: <https://www.seplag.ce.gov.br/2025/02/05/seplag-desenvolve-bi-para-monitorar-beneficios-sociais-concedidos-pelo-governo-do-ceara/>. Acesso em: 10 jun. 2025.

Sharp, H.; Rogers, Y.; Preece, J. *Interaction Design: Beyond Human–Computer Interaction*. 6. ed. Hoboken: Wiley, 2019.

T., V. Visualizing data in react with apache echarts + react query. *Medium*, 2024. Disponível em: <https://medium.com/@vaibhav11t/visualizing-data-in-react-with-apache-echarts-react-query-b13c836caafa>. Acesso em: 16 set. 2025.

VAIBHAV. Visualizing data in react with apache echarts and react query. *Medium*, 2023. Disponível em: <https://medium.com/@vaibhav11t/visualizing-data-in-react-with-apache-echarts-react-query-b13c836caafa>. Acesso em: 15 set. 2025.