CENTRO ESTADUAL DE EDUCAÇÃO TECNOLÓGICA PAULA SOUZA

Faculdade de Tecnologia de Mauá

Curso Superior de Tecnologia em Desenvolvimento de Software Multiplataforma

Gustavo Santana Cardoso

Vincenzo Bonatti

Walber Cesar de Sousa Santos

Soundsnap: Rede Social de Capas de Álbuns Musicais

Mauá

2025

Gustavo Santana Cardoso
Vincenzo Bonatti
Walber Cesar de Sousa Santos

Soundsnap: Rede Social de Capas de Álbuns Musicais

Relatório técnico-científico apresentado à Fatec de Mauá, como exigência parcial para obtenção do título de Tecnólogo em Desenvolvimento de Software Multiplataforma, sob a orientação da profa. Me. Luana Lourenço.

Mauá
2025

RESUMO

O SoundSnap é uma plataforma web desenvolvida como uma rede social dedicada à visualização, descoberta e interação com capas de álbuns musicais, integrando funcionalidades sociais ao universo da música digital. Criado em React com JavaScript e utilizando a API do Spotify, o sistema permite que os usuários explorem um feed dinâmico de álbuns, visualizem informações detalhadas sobre artistas e construam um perfil personalizado por meio de curtidas e favoritos. O SoundSnap surge da evolução das formas de consumo musical, marcada pela transição das mídias físicas para os serviços de streaming, e atende a uma lacuna existente no mercado: a ausência de plataformas focadas na estética visual das capas de álbuns e na construção de identidade musical por meio da interação social. Assim, além de promover a descoberta de novos conteúdos, o SoundSnap também valoriza o papel cultural das capas e a relação do usuário com sua identidade musical. Do ponto de vista acadêmico e técnico, o projeto constitui um estudo de caso completo que envolve desenvolvimento web moderno, integração com APIs externas, uso de banco de dados SQLite e aplicação de técnicas de testes e metodologias de qualidade de software. O sistema do SoundSnap foi construído com foco em modularidade, escalabilidade e boas práticas de engenharia, utilizando metodologias como TDD para garantir confiabilidade nas funcionalidades essenciais. Foram realizados testes unitários, de integração, de performance e de segurança, assegurando que cada módulo funcionasse como esperado, que os dados fossem processados corretamente e que a experiência do usuário fosse estável e segura. Além disso, o SoundSnap passou por uma avaliação de acessibilidade utilizando o Axe DevTools, identificando ajustes necessários e reforçando o compromisso com a inclusão digital conforme as diretrizes da WCAG 2.1. Dessa forma, o SoundSnap se apresenta como uma solução funcional, culturalmente relevante e tecnicamente estruturada, unindo experiência musical, interação social e boas práticas de desenvolvimento para entregar uma plataforma inovadora e alinhada às demandas contemporâneas do ambiente digital.

Palavras-chave: música; redes sociais; capas de álbuns; desenvolvimento web; spotif

SUMÁRIO

1 INTRODUÇÃO

1.1 Contextualização

1.1.1 A evolução dos aplicativos e sites de música

O artigo de Xiaorui Guo (2023) analisa a transformação da indústria musical desde os meios físicos até o domínio das plataformas digitais de streaming, destacando como a tecnologia alterou os modelos de consumo, distribuição e lucro das empresas do setor.

No início, a era da mídia física era sustentada por produtos como discos de vinil, fitas cassete e CDs, que exigiam altos custos de produção, transporte e armazenamento. Esse modelo tornava o mercado restrito às grandes gravadoras e limitava o acesso de artistas independentes, uma vez que o controle sobre a distribuição e a divulgação era centralizado (Guo, 2023).

Com o avanço da tecnologia digital e da internet, iniciou-se uma revolução no consumo de música. O surgimento de arquivos em formato MP3 e plataformas de compartilhamento, como o Napster, reduziu drasticamente os custos de cópia e distribuição, dando origem a novas formas de circulação musical. Em seguida, serviços pagos como o iTunes criaram um modelo mais sustentável para o consumo digital (Guo, 2023). Essa fase marcou o declínio das vendas físicas e obrigou a indústria a adaptar-se, buscando novas estratégias para garantir a rentabilidade no ambiente virtual (United Nations Digital Library, 2021).

Posteriormente, a chegada dos serviços de streaming consolidou uma nova etapa no mercado musical. Aplicativos e sites como Spotify e Apple Music transformaram a maneira como as pessoas acessam conteúdo sonoro: em vez de comprar faixas individuais, os usuários passaram a pagar por assinaturas que garantem acesso ilimitado a catálogos inteiros. Essa mudança representou uma transição do modelo de posse para o de acesso, tornando o consumo mais flexível e ampliando o alcance global da música (Guo, 2023).

Além disso, o streaming possibilitou maior visibilidade para artistas independentes, embora ainda existam desigualdades na distribuição de lucros e na

exposição dentro das plataformas (United Nations Digital Library, 2021). Guo (2023) também destaca que os serviços de streaming trouxeram novas formas de curadoria, onde algoritmos e playlists personalizadas assumem papel central na descoberta de músicas e na relação entre artistas e ouvintes.

A autora ressalta as tendências tecnológicas emergentes que continuam a moldar a indústria musical. Entre elas estão o uso de inteligência artificial na recomendação de conteúdo e na composição, o blockchain para registro transparente de direitos autorais, o uso de NFTs como nova forma de monetização e as experiências imersivas com realidade virtual e aumentada (Guo, 2023). Tais inovações indicam que os aplicativos e sites de música seguirão evoluindo conforme a tecnologia amplia as possibilidades de criação e interação no ambiente digital.

A fundamentação teórica para uma rede social de músicas deve se apoiar em três pilares inter-relacionados: Redes Sociais Digitais, Consumo de Música e Identidade e Indústria Cultural. O primeiro pilar foca na estrutura da plataforma. As redes sociais digitais (RSDs) são definidas como sistemas que permitem a construção de um perfil, a articulação de uma lista de conexões e a visualização mútua dessas conexões (Boyd e Ellison, 2007; Recuero, 2009). No contexto da música, a plataforma atua como um espaço mediado que facilita o desenvolvimento de Capital Social, tanto na forma de Laços Fortes (conexões íntimas) quanto de Laços Fracos (conexões casuais). Estes últimos são cruciais para a descoberta de novas músicas e a circulação de informações musicais, conforme a teoria de Granovetter (1973).

Este pilar aborda a relação do usuário com o conteúdo musical. A música transcende seu valor de mercado e funciona como um emblema de identidade (DeNora, 2000), sendo vital para a construção e expressão da identidade pessoal e social do usuário. O consumo musical em plataformas digitais se insere na Cultura de Consumo (Douglas & Isherwood, 1979), onde as escolhas (como playlists e artistas seguidos) comunicam status e pertencimento social. A Ubiquidade (acesso em qualquer lugar) e o Hedonismo (prazer em ouvir) são fatores comportamentais chave que impulsionam a interação nessas plataformas.

As redes sociais são motores de viralização e promoção na indústria musical (Kozinets, 2001), transformando a forma como artistas e gravadoras interagem com o

público. O ambiente digital cria uma "abundância avassaladora" de escolhas, tornando a curadoria social e algorítmica essencial para a navegação do usuário.

O projeto Soundsnap surge como uma proposta de rede social dedicada à valorização e ao compartilhamento de capas de álbuns musicais. Utilizando a API do Spotify, a plataforma gera um feed interativo com álbuns aleatórios, exibindo suas capas, informações relacionadas e detalhes sobre os artistas. Além disso, os usuários do SoundSnap podem criar contas, curtir e favoritar seus álbuns preferidos, construindo assim um perfil musical personalizado.

O projeto SoundSnap está sendo desenvolvido como uma aplicação web em React com JavaScript, o que possibilita maior dinamismo na interface, modularidade no código e boa escalabilidade. A integração com banco de dados garante o armazenamento seguro das informações de usuários e a manutenção das interações sociais.

A proposta do SoundSnap se justifica pela carência de plataformas que unam interação social e valorização estética no universo musical. Embora existam serviços consolidados de streaming e redes sociais tradicionais, ainda há espaço para produtos que deem destaque à dimensão visual da música e à construção de identidade cultural dos usuários. Do ponto de vista acadêmico, o projeto contribui para a aplicação prática dos conhecimentos adquiridos no curso, especialmente em desenvolvimento web, integração com APIs, banco de dados e experiência do usuário (UX/UI).

As tecnologias web, em especial aquelas baseadas em linguagens modernas como o JavaScript e frameworks como o React, permitem a construção de sistemas dinâmicos, escaláveis e responsivos, que se destacam pela capacidade de atender tanto a demandas funcionais quanto estéticas. Essas ferramentas são amplamente utilizadas no mercado atual justamente por sua flexibilidade e integração com APIs e bancos de dados, o que as torna adequadas para projetos que buscam oferecer experiências interativas e personalizadas.

Nesse contexto, o projeto Soundsnap surge como uma iniciativa acadêmica que aplica tais conceitos, explorando a combinação entre desenvolvimento web, integração de serviços externos e valorização de aspectos culturais. A proposta reflete

a relevância das tecnologias digitais na criação de soluções inovadoras, que não apenas atendem a necessidades práticas, mas também ampliam as possibilidades de expressão e interação social no ambiente virtual.

1.2 Objetivo

Desenvolver uma rede social multiplataforma com o foco na área musical, realizando interações dos usuários com integração por meio de API ao Spotify, proporcionando uma interação no mundo da música. Utilizando tecnologias como, JavaScript e React, que possibilita maior dinamismo na interface, modularidade no código e boa escalabilidade.

1.2.1 Objetivos específicos

- Feed interativo com álbuns aleatórios exibindo suas capas, informações relacionadas e detalhes sobre os artistas.
- Os usuários podem criar contas, curtir e favoritar seus álbuns preferidos. • Construir um perfil musical personalizado

1.3 Justificativa

Estudar e aplicar técnicas de teste avançadas no desenvolvimento de software é crucial para construir sistemas mais robustos e eficientes. Por exemplo, a integração de inteligência artificial nos testes permite automatizar tarefas como a geração de dados, priorização de casos e análise de resultados, o que eleva a confiabilidade e a cobertura de testes (Khaliq, Farooq e Khan, 2022).

Além disso, mesmo em software de pesquisa, que muitas vezes tem alto grau de complexidade, os desenvolvedores reconhecem que testar seu código é vital para garantir resultados corretos, embora enfrentem desafios devido à falta de cultura e treinamento (Eisty e Carver, 2022).

Por outro lado, a aplicação de aprendizado adversarial no contexto de testes automatizados tem se mostrado uma estratégia promissora para aumentar a resiliência do sistema: gerando casos “difíceis” para expor vulnerabilidades e tornar o software mais robusto (Vitorino et al., 2023).

A implementação de pipelines de CI/CD com testes automatizados também é uma prática que traz benefícios concretos: permite feedback quase imediato sobre mudanças no código, reduz retrabalho e melhora a qualidade geral dos builds (Junior et al., 2025).

Mais inovador ainda, há propostas de usar otimização quântica combinada com aprendizado de máquina para priorizar quais testes executar primeiro, o que reduz o tempo de execução e ainda eleva a capacidade de detectar defeitos “críticos” (Bandarupalli, 2025).

Em síntese, essas estratégias, IA nos testes, aprendizado adversarial, pipelines automatizados e até otimização quântica, demonstram que investir em testes não é apenas uma questão de evitar bugs, mas de construir software mais confiável, sustentável e eficiente, tanto no curto quanto no longo prazo.

No contexto acadêmico, o projeto *Soundsnap* proporciona a oportunidade de aplicar na prática os conhecimentos adquiridos sobre desenvolvimento web e integração de tecnologias modernas, como o React com JavaScript e o consumo de dados por meio da API do Spotify. A implementação de testes de qualidade contribui para validar as funcionalidades do sistema, identificar falhas e assegurar a confiabilidade do software.

1.4 Problema de pesquisa

O problema de garantir qualidade e performance em software multiplataforma demanda uma combinação de boas práticas arquiteturais, automação de testes e escolha cuidadosa de frameworks e ferramentas.

Primeiro, é essencial compreender que frameworks multiplataforma variam bastante em termos de overhead de performance: para alguns frameworks, o “bridge”

que conecta o código cross-platform ao sistema nativo introduz latência perceptível; já para outros, o desempenho pode se aproximar (ou até superar) o nativo, dependendo da métrica e do uso (Biørn-Hansen et al., 2020).

Para apoiar essa escolha, pesquisas recentes analisam os frameworks mais adotados no mercado (como React Native, Flutter, MAUI) e destacam tanto a tendência de mercado quanto as lacunas de pesquisa em testes, evolução e manutenção desses sistemas (Jošt e Taneski, 2025).

Do ponto de vista da qualidade, é importante adotar ferramentas consolidadas: um estudo bibliométrico revelou que diversos projetos utilizam ferramentas de análise de código, métricas e automação para garantir a qualidade durante o desenvolvimento (Adigneri et al., 2022).

Para testes multiplataforma, práticas bem definidas são essenciais. Por exemplo, em ambientes Linux, Windows e Mac, recomenda-se o uso de testes automatizados, virtualização ou containerização e integração contínua (CI) para validar compatibilidade, usabilidade e performance entre plataformas (Soujanya Reddy, 2021).

Além disso, há inovação no uso de modelos de linguagem (LLMs) para gerar scripts de teste: essa abordagem pode facilitar a escrita e migração de testes entre plataformas diferentes, acelerando o processo e mantendo a consistência (Yu et al., 2023).

No que diz respeito à arquitetura, uma abordagem holística propõe o uso de uma linguagem de alto nível para abstrair os detalhes específicos das plataformas, gerando código que pode ser implantado em diferentes sistemas, reduzindo a duplicação e facilitando a manutenção (Blanco e Lucrédio, 2021).

Para otimizar a performance das aplicações, práticas como otimização de código (remover redundâncias, otimizar chamadas de API), compressão e uso adaptativo de imagens, além do monitoramento de rede (com estratégias de retry e qualidade adaptativa) são recomendadas (AppVin Technologies, s.d.).

No âmbito de testes, é vantajoso priorizar automação desde cedo, usando ferramentas como Appium ou Selenium para testar em múltiplos dispositivos e sistemas, e serviços como BrowserStack para emular diferentes ambientes (MoldStud / Crudu, 2024).

Por fim, para estruturar todo esse processo de forma confiável e eficiente, a arquitetura de desenvolvimento deve incorporar CI/CD e padronização de ferramentas/processos. O Azure Well-Architected Framework, por exemplo, recomenda definir padrões consistentes para práticas de garantia de qualidade, controle de versão, pipelines de automação, revisão de código e outros elementos operacionais (Microsoft, 2025)

Esse questionamento norteia o estudo e o desenvolvimento do *Soundsnap*, buscando compreender como técnicas de programação, metodologias de desenvolvimento e testes de qualidade podem ser aplicados para entregar uma solução robusta, escalável e alinhada às necessidades dos usuários.

1.5 Estrutura do relatório

Este relatório está organizado de forma a apresentar, de maneira clara e progressiva, todas as etapas envolvidas no desenvolvimento do SoundSnap, desde seus fundamentos teóricos até sua validação técnica. A Introdução reúne o contexto histórico da evolução dos meios de consumo musical, discute a transformação da indústria fonográfica até o surgimento das plataformas digitais, apresenta o objetivo central do projeto SoundSnap, sua justificativa e o problema de pesquisa que orienta o estudo — especialmente sobre práticas de qualidade e performance em sistemas multiplataforma. Em seguida, o capítulo de Desenvolvimento aprofunda os referenciais teóricos utilizados, contemplando literatura sobre redes sociais digitais, consumo musical e técnicas modernas de desenvolvimento de software, além de detalhar as metodologias adotadas, ferramentas empregadas e critérios de avaliação usados no estudo de caso. Nesse capítulo, também são descritas as funcionalidades da aplicação, a ficha técnica do sistema e uma análise abrangente das práticas de teste

implementadas, incluindo testes unitários, de integração, performance, segurança e avaliação de acessibilidade segundo as diretrizes da WCAG 2.1.

O relatório avança com a apresentação dos resultados obtidos durante as etapas de análise do SoundSnap, destacando pontos de eficácia, limitações encontradas e contribuições práticas do projeto para o aprendizado em engenharia de software. Por fim, nas Considerações Finais, são sintetizadas as principais conclusões do estudo, seguidas de recomendações para aprimoramento do sistema, limitações do trabalho e perspectivas futuras para evolução da plataforma, como expansão para dispositivos móveis, melhorias de acessibilidade e uso de algoritmos de recomendação. Dessa forma, a estrutura do relatório permite ao leitor compreender tanto o contexto conceitual quanto os aspectos técnicos e práticos que fundamentam o desenvolvimento do SoundSnap.

2 DESENVOLVIMENTO

2.1 Fundamentação teórica

2.1.1 Revisão de literatura

O desenvolvimento do aplicativo Soundsnap, uma plataforma que integra funcionalidades de streaming musical e rede social, exige uma combinação de metodologias, tecnologias e práticas de testes que garantam agilidade, qualidade e escalabilidade. Nesse contexto, as metodologias ágeis, especialmente o Scrum, tornam-se essenciais para organizar o fluxo de trabalho. O Scrum é um framework iterativo focado na entrega contínua de valor e na adaptação rápida às mudanças (Souza, 2022). A divisão do projeto em sprints curtas permite que funcionalidades essenciais do Soundsnap sejam desenvolvidas e validadas de forma incremental, incorporando feedback constante dos usuários. Essa flexibilidade está alinhada aos princípios do Manifesto Ágil, que prioriza “responder a mudanças mais que seguir um plano” (Thomaz, 2021). Além disso, práticas como definição de pronto, integração contínua e reuniões diárias reforçam a transparência e a melhoria contínua dentro da equipe (DevMedia, 2018).

Na camada de interface do aplicativo SoundSnap, a escolha do React (ou React Native, para mobile) se destaca por oferecer modularidade e reatividade, permitindo construir componentes reutilizáveis para áreas como listas de músicas, perfis e sistema de interação social. Segundo a documentação do MDN, o React facilita a criação de interfaces dinâmicas por meio do uso de componentes e do Virtual DOM, o que melhora o desempenho e simplifica o desenvolvimento (MDN Web Docs, 2023). Além disso, por ser uma biblioteca flexível e não um framework completo, o React permite combinar soluções específicas para gerenciamento de estado, rotas e testes, conforme discutido por Rossi (2021). Estudos acadêmicos reforçam a importância de padronização visual e reutilização de componentes, como demonstrado no trabalho de Alves (2022), mostrando que um design system em React melhora a consistência da interface e reduz retrabalho. Para o Soundsnap, isso significa uma interface fluida, responsiva e mais simples de manter, mesmo com expansão futura.

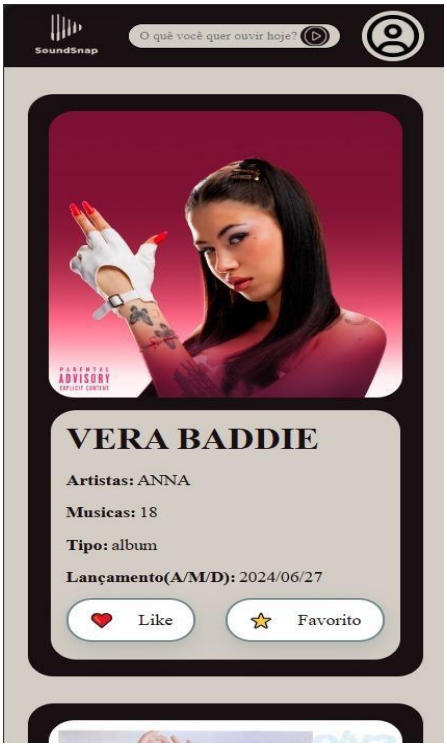


FIGURA 1 – Página inicial do SoundSnap mobile

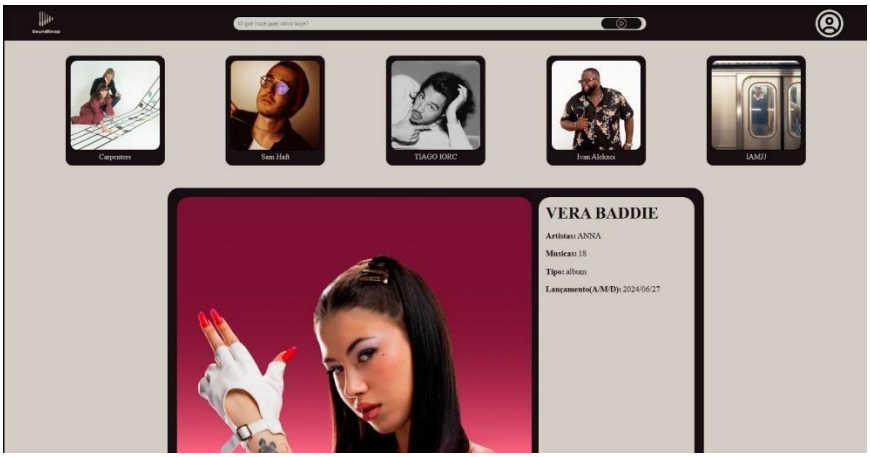


FIGURA 2 – Página inicial SoundSnap Web

No que diz respeito à qualidade do software, a adoção de TDD (Test-Driven Development) e BDD (Behavior-Driven Development) fortalece a confiabilidade do aplicativo. O TDD propõe criar testes antes do código, favorecendo design limpo, modular e menos propenso a falhas (Visure Solutions, 2020). Já o BDD amplia essa lógica ao priorizar testes que descrevem o comportamento esperado pelo usuário em linguagem natural, por exemplo, “Dado que o usuário gosta de uma música, quando ele clicar em curtir, então a curtida deve ser registrada”, o que melhora a comunicação

entre desenvolvedores, QA e stakeholders (Startbit, 2021). A combinação dessas práticas se encaixa bem na pirâmide de testes, que orienta manter muitos testes unitários na base e testes de aceitação no topo (Barbosa, 2022). Autores como Scarabelli (2020) destacam que o uso conjunto de TDD e BDD reduz bugs, aumenta a clareza dos requisitos e facilita futuras refatorações. Para um aplicativo como o Soundsnap, que exige alta estabilidade em funcionalidades, como interações em tempo real e manipulação de playlists, essas práticas são fundamentais para garantir uma experiência confiável e sem falhas.

Por fim, há autores que defendem que TDD e BDD são essencialmente complementares, pois ambos focam em garantir que “o software certo” seja produzido (Talking About Testing, 2023). No Soundsnap, essa integração permite validar tanto a lógica interna (por exemplo, cálculo de recomendações musicais) quanto o comportamento visível para o usuário, sustentando um ciclo de desenvolvimento robusto, escalável e centrado na experiência do usuário.

2.1.2 Técnicas de desenvolvimento de software

O desenvolvimento de software moderno combina várias técnicas para entregar valor rápido, manter qualidade e facilitar evolução. O desenvolvimento ágil é a base desse modelo: o Manifesto Ágil e suas práticas defendem ciclos curtos, feedback contínuo e priorização do valor ao usuário (Highsmith, 2001). Metodologias e frameworks como Scrum organizam o trabalho em sprints, dailies e revisões, permitindo que equipes priorizem entregas incrementais e ajustem o backlog conforme o retorno dos usuários (Sutherland, 2014; Souza, 2022). Práticas ágeis, quando bem definidas, aumentam a previsibilidade e a capacidade de adaptação do time (Thomaz, 2021; DevMedia, 2018).

Complementando a agilidade, Integração Contínua (CI) e Entrega Contínua / Continuous Delivery (CD) automatizam a compilação, testes e deploy, reduzindo riscos e acelerando a entrega de novas funcionalidades. A CI propõe que cada alteração de código seja integrada e testada automaticamente para detectar erros cedo (Fowler, 2006; Duvall, Matyas & Glover, 2007). A CD estende essa ideia com pipelines que possibilitam disponibilizar mudanças em produção de forma segura e

repetível, trazendo benefícios como menor tempo de recuperação e liberação frequente para os usuários (Humble & Farley, 2010; Fowler, 2006).

Para que essas práticas funcionem na prática, é essencial escrever código limpo. Código limpo favorece legibilidade, manutenção e facilita a colaboração entre desenvolvedores; características centrais para reduzir custos de evolução do sistema (Martin, 2008). Boas decisões arquiteturais e princípios de design (por exemplo, os princípios SOLID) ajudam a manter o código modular e testável, o que, por sua vez, melhora a eficácia de testes automatizados e pipelines CI/CD (Martin, 2017; Spinellis, 2006).

As práticas de teste (TDD e BDD) são aliados diretos de CI/CD e código limpo. O TDD incentiva escrever testes unitários antes do código, promovendo design orientado a testes e reduzindo regressões (Visure Solutions, 2020). O BDD foca em especificar o comportamento esperado em termos legíveis (Dado/Quando/Então), alinhando requisitos, QA e desenvolvimento (Startbit, 2021). A combinação TDD+BDD, organizada segundo a pirâmide de testes (muitos testes unitários, menos testes de integração e alguns testes de aceitação), fornece uma cobertura eficiente e ciclos de feedback rápidos (Barbosa, 2022; Scarabelli, 2020; Talking About Testing, 2023).

Por fim, documentar não é apenas bom senso: documentação clara reduz o custo de manutenção, facilita onboarding e garante que decisões arquiteturais e regras de negócio fiquem registradas (Forward & Lethbridge, 2002; Steinmacher et al., 2016). Estudos mostram que documentação acessível e atualizada melhora a compreensão do código e acelera a entrada de novos colaboradores (Pascarella et al., 2018). Integrar documentação no fluxo ágil — por exemplo, atualizando documentação como parte da Definition of Done — garante que ela não fique desatualizada com o tempo.

Em suma, a combinação de desenvolvimento ágil (Highsmith, 2001; Beck, 2004; Sutherland, 2014), CI/CD (Fowler, 2006; Humble & Farley, 2010; Duvall et al., 2007), código limpo e boa arquitetura (Martin, 2008; Martin, 2017; Spinellis, 2006), TDD/BDD (Visure Solutions, 2020; Startbit, 2021; Barbosa, 2022) e documentação ativa (Forward & Lethbridge, 2002; Pascarella et al., 2018; Steinmacher et al., 2016) cria um ambiente propício para projetar e manter aplicações complexas como o

Soundsnap — assegurando entregas frequentes, alta qualidade e menor custo de evolução.

2.1.3 Metodologias de teste de software

Para garantir a qualidade e confiabilidade da aplicação do SoundSnap, foi implementado uma estratégia abrangente de testes em três categorias distintas e complementares:

- Testes funcionais (caixa-preta) verificam se o sistema executa corretamente todas as funções esperadas, validando os requisitos do usuário sem conhecimento da estrutura interna do código.
- Testes estruturais (caixa-branca) avaliam a lógica interna, fluxos de controle e caminhos do código-fonte, garantindo que a implementação técnica esteja correta e otimizada.
- Testes baseados em defeitos indicam possíveis falhas, comportamentos incorretos e cenários de erro, simulando situações que podem comprometer a experiência do usuário.

Testes funcionais para garantir a experiência do usuário focam em validar que as principais funcionalidades do sistema atendam completamente aos requisitos especificados, garantindo uma experiência fluida e intuitiva. A descrição do teste de **cadastro e login** visa validar o processo completo de autenticação com e-mail e senha, incluindo validações de formato e segurança, e seu resultado esperado é ter um cadastro e login bem sucedidos para credenciais válidas, mensagens de erro apropriadas para e-mails duplicados ou inválidos. Já na **busca de artistas** visa testar a funcionalidade de pesquisas com nomes completos, parciais e caracteres especiais, avaliando a precisão dos resultados, com os resultados esperados sendo a exibição correta da lista de resultados relevantes ou mensagem clara “nenhum resultado encontrado” quando aplicável.

Os testes estruturais para avaliar a lógica interna, examinando profundamente o código-fonte, garantindo que todas as condições lógicas, caminhos de execução e estruturas internas funcionem corretamente e de forma eficiente. O **módulo de login** testa todas as condições da função login “senha incorreta”, “usuário inexistente”, “conta bloqueada” e “sessões expiradas”. No **feed de músicas** valida todos os

caminhos possíveis de carregamento do feed “sem conexão”, “recomendações”, “feed vazio” e “erros de servidor”.

Os testes baseados em defeitos prevenindo falhas e comportamentos inesperados, simula cenários problemáticos conhecidos e situações de uso inadequado, detectando falhas antes que afetem os usuários reais da plataforma.
Exemplos de simulações de cenários problemáticos:

- **Curtidas duplicadas:** usuário clica repetidamente no botão de curtir em rápida sucessão, com o resultado esperado sendo o sistema alternar corretamente entre curtir e descurtir, sem criar registros duplicados no banco de dados.
- **Injeção de código:** tentativas de inserir scripts maliciosos em campos de entrada de texto, com o resultado esperado do sistema sanitizar todas as entradas, bloqueando injeções de código e mantendo a segurança.

Cada teste será meticulosamente documentado em casos de testes estruturados, garantindo rastreamento, reprodutividade e facilidade de manutenção ao longo do ciclo de vida do projeto SoundSnap.

2.2 Metodologia

2.2.1 Tipo de pesquisa

Para a elaboração deste trabalho, adotou-se o estudo de caso como método de pesquisa. Essa abordagem foi selecionada por permitir uma análise aprofundada de um objeto específico — neste caso, o Soundsnap, um site desenvolvido pelo grupo com o objetivo de possibilitar aos usuários a descoberta de novos álbuns musicais, além de realizar ações como curtir, favoritar e explorar artistas.

O estudo de caso é uma metodologia amplamente utilizada em pesquisas aplicadas em tecnologia e desenvolvimento de software, pois possibilita investigar fenômenos contemporâneos dentro de seu contexto real. Segundo Yin (2015), essa técnica é indicada quando se busca compreender detalhadamente um produto, sistema ou processo, considerando suas características, funcionalidades e interação

com usuários ou ambiente de uso. Esse alinhamento torna o estudo de caso adequado para compreender as etapas, desafios e decisões envolvidas no desenvolvimento do Soundsnap.

A escolha do estudo de caso justifica-se pelo caráter prático do projeto e pela necessidade de analisar o comportamento da aplicação, desde sua concepção até as etapas de implementação e testes. Assim, a pesquisa concentra-se na observação direta do desenvolvimento da plataforma, incluindo a definição dos requisitos, arquitetura, tecnologias utilizadas e funcionalidades implementadas — como o sistema de descoberta de álbuns, o mecanismo de curtidas e o recurso de favoritos.

Além disso, o estudo de caso permite documentar e avaliar como o site responde aos objetivos propostos, incluindo aspectos de experiência do usuário, desempenho das funcionalidades e potencial de expansão futura. A análise do Soundsnap como estudo de caso possibilita também identificar melhorias, compreender limitações técnicas e apontar direções para evoluções futuras da plataforma.

Dessa forma, a utilização do estudo de caso não apenas sustenta o caráter científico do trabalho, mas também favorece uma compreensão integral do processo de desenvolvimento do Soundsnap, evidenciando sua relevância como produto tecnológico e como objeto de pesquisa

2.2.2 Procedimentos de análise

A interpretação dos dados obtidos durante as etapas de desenvolvimento e validação da aplicação foi realizada de maneira organizada, considerando três dimensões centrais: verificação das funcionalidades, avaliação da experiência do usuário e análise do desempenho técnico.

Os testes funcionais, fundamentados na abordagem de caixa-preta, têm como finalidade verificar se a aplicação executa adequadamente todas as funcionalidades especificadas, assegurando o cumprimento dos requisitos estabelecidos pelo usuário sem a necessidade de examinar sua estrutura interna. Complementarmente, os testes

estruturais, baseados na técnica de caixa-branca, concentram-se na análise da lógica interna do sistema, contemplando fluxos de controle, caminhos de execução e estruturas decisórias presentes no código-fonte. Essa etapa permite identificar inconsistências, redundâncias e fragilidades na implementação, contribuindo para a otimização da arquitetura do software. Ademais, os testes orientados à detecção de defeitos são empregados para antecipar possíveis falhas e comportamentos inesperados, por meio da simulação de cenários de erro que potencialmente comprometeriam a estabilidade da aplicação e a experiência do usuário.

A integração dessas três abordagens possibilita uma avaliação abrangente e rigorosa do sistema, promovendo maior confiabilidade, robustez e qualidade técnica ao produto final.

2.2.3 Critérios de avaliação

A avaliação do software desenvolvido foi conduzida com base em critérios técnicos amplamente utilizados na engenharia de software, de modo a assegurar a qualidade, confiabilidade e adequação da aplicação aos objetivos propostos. Para este estudo, foram considerados cinco critérios principais: funcionalidade, performance, segurança, usabilidade e manutenibilidade. Cada critério permitiu analisar dimensões específicas do Soundsnap, contribuindo para uma validação abrangente e rigorosa do sistema.

A funcionalidade foi avaliada verificando-se se todas as funcionalidades previstas no escopo do projeto operavam conforme os requisitos estabelecidos, incluindo a descoberta de álbuns, interação por meio de curtidas e favoritos, navegação entre páginas e exibição de informações dos artistas. Essa análise buscou confirmar que o software desempenha corretamente suas atividades essenciais, sem falhas ou comportamentos inesperados.

A performance contemplou aspectos relacionados ao tempo de resposta das interfaces, velocidade de carregamento dos conteúdos, eficiência no processamento das requisições e estabilidade em diferentes condições de uso. Foram observados eventuais atrasos, gargalos ou degradações de desempenho que pudessem comprometer a experiência do usuário ou a fluidez da navegação no SoundSnap.

O critério de segurança envolveu a verificação de mecanismos destinados a proteger dados e interações dos usuários, incluindo controles de acesso, proteção contra ações indevidas e tratamento adequado de entradas inválidas. Foram analisadas potenciais vulnerabilidades, riscos de uso indevido e conformidade com boas práticas de desenvolvimento seguro, considerando o contexto de uma aplicação web.

A usabilidade foi avaliada a partir da facilidade de interação dos usuários com a plataforma, clareza dos elementos de interface, organização das informações e consistência dos fluxos de navegação. Esse critério permitiu identificar possíveis dificuldades, ambiguidades ou obstáculos no uso cotidiano da aplicação, além de apontar oportunidades de aprimoramento da experiência do usuário.

Por fim, a manutenibilidade considerou a estruturação do código, modularidade, clareza na organização dos componentes e facilidade de realizar ajustes, correções ou expansões futuras. Esse critério é fundamental no contexto de projetos educacionais e profissionais, pois assegura que o software possa evoluir e ser aprimorado sem comprometer sua estabilidade.

A aplicação conjunta desses critérios permitiu uma avaliação completa do Soundsnap, garantindo que o sistema atenda aos parâmetros de qualidade esperados para um software multiplataforma, bem como aos objetivos estabelecidos para o presente estudo de caso.

2.3 Ficha técnica

Nome do sistema/software:	Soundsnap
Versão:	1.0
Desenvolvedores:	Gustavo Santana, Vincenzo Bonatti, Walber Cesar
Data de criação:	04 de setembro de 2025
Plataforma(s):	Web
Linguagens utilizadas:	Javascript
Frameworks:	React

Banco de dados:	SQLITE
Requisitos de hardware:	Navegador com suporte a HTML5
Requisitos de software:	Node.js
Ferramentas de desenvolvimento:	Visual Studio Code, Github,Git, Trello, Figma
Objetivo:	Conhecer novos álbuns de artistas, curtir e favoritar

Fonte: Estrutura adaptada pelas autoras.

2.4 Técnicas e testes de software e avaliação da acessibilidade

2.4.1 Testes unitários

No desenvolvimento do SoundSnap, os testes unitários foram essenciais para garantir que cada componente crítico do sistema funcionasse corretamente de forma isolada. Para isso, foi utilizada a ferramenta Jest, que se mostrou a mais adequada ao projeto por oferecer configuração simples, excelente suporte para JavaScript/Node.js e recursos nativos de *mocking*, fundamentais para simular comportamentos externos.

Os testes foram aplicados principalmente nos módulos do backend, responsáveis por funcionalidades como cadastro de usuários, autenticação, gerenciamento de artistas e álbuns, e geração de recomendações musicais. Cada função foi testada individualmente para verificar se retornava os valores esperados, tratava exceções corretamente e se comportava de maneira consistente em diferentes cenários. O uso de *mocks* permitiu simular respostas do banco de dados e APIs externas, garantindo que a lógica interna pudesse ser validada sem depender dos outros componentes do sistema.

Além disso, partes do SoundSnnap criadas em React Native também tiveram componentes testados com Jest, verificando se elementos de interface reagiam corretamente às ações do usuário, como clicar em botões, navegar entre telas ou buscar artistas e músicas.

Com essa estratégia, os testes unitários ajudaram a identificar falhas precocemente, facilitaram a manutenção do código e proporcionaram maior

segurança na evolução do SoundSnap. A utilização de Jest como ferramenta única permitiu padronização, agilidade e confiabilidade ao processo de desenvolvimento.

2.4.2 Testes de integração

No projeto *SoundSnap*, os testes de integração foram conduzidos de maneira incremental e apoiados pelo uso do **Git** como ferramenta central de versionamento e integração contínua. Adotou-se o modelo *Continuous Integration* (CI), no qual cada nova funcionalidade ou correção desenvolvida em um branch separado era integrada ao repositório principal por meio de *pull requests*. A prática de integração contínua permite identificar falhas de forma rápida, reduzindo retrabalho e conflitos de código (Fowler, 2006).

Inicialmente, o desenvolvedor implementava a nova funcionalidade ou correção em um branch dedicado. Em seguida, antes do *merge*, eram executados testes unitários e testes de integração locais, garantindo que a comunicação entre módulos — como autenticação, gerenciamento de playlists, sistema de recomendação e comunicação com a API — permanecesse consistente. Esse processo seguiu o princípio de que “integrar cedo e frequentemente” reduz problemas de compatibilidade e melhora a qualidade final do software (Duvall, Matyas & Glover, 2007).

Os testes de integração executados envolveram a verificação do fluxo completo de funcionalidades, simulando o comportamento esperado do usuário. Foram validados aspectos como: comunicação entre front-end e back-end, persistência de dados, resposta da API aos módulos consumidores e coerência entre os estados compartilhados pelos componentes. Essa abordagem tornou possível identificar erros que não seriam detectados apenas com testes unitários.

Embora ferramentas automatizadas de CI, como GitHub Actions ou GitLab CI, possam ser utilizadas para este processo, neste projeto a integração contínua ocorreu de maneira mais simples, mas ainda eficiente, utilizando o Git como o núcleo de controle e sincronização das versões.

2.4.3 Testes de performance

Os testes de performance do SoundSnap foram realizados para validar o desempenho da rota ‘POST / api/ auth/ login’ em diferentes cenários de carga, garantindo que logins individuais sejam processados rapidamente, o sistema suporte múltiplas requisições simultâneas, o consumo de memória permaneça controlado e a performance seja consistente sob diferentes cargas.

O primeiro teste de login, foi para verificar a se a velocidade de processamento é <500ms (milissegundos), e as validações realizadas teve como resultado o tempo de resposta <500ms, login bem-sucedido com token e mediação precisa de performance, como demonstrado na Figura 1.

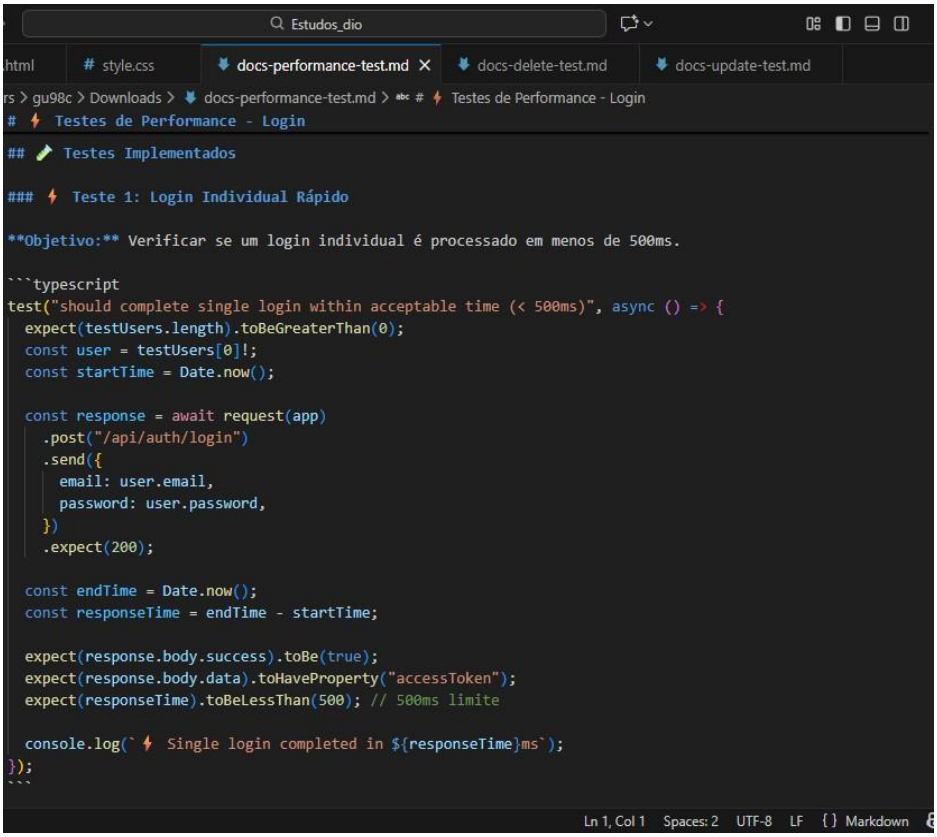


FIGURA 1 – TESTE 1 DE PERFORMANCE

O segundo foi para logins simultâneos (concorrência), com o objetivo de avaliar a capacidade do sistema de processar 5 logins simultâneos eficientemente, e por fim as validações realizadas foram de 5 logins simultâneos <3 segundos, todos os tokens gerados corretamente e tempo médio por login calculado.

Logo, o terceiro teste foi performance sequencial (carga sustentada), com objetivo de medir a consistência da performance durante 10 logins sequenciais, suas validações realizadas foram as de tempo médio <500ms, tempo máximo <1000ms, tempo total <6 e métricas detalhadas de performance.

O quarto teste é o cenário misto (validos + inválidos), e teve como objetivo avaliar a performance com uma mistura de tentativas válidas e invalidas de login, e suas validações realizadas foram os logins válidos processados corretamente, logins inválidos rejeitados adequadamente, performance essa que foi mantida em cenário misto e tempo total <3 segundos.

Por último, foi realizado o teste de monitoramento de memória do SoundSnap, com o intuito de medir o consumo de memória durante 20 logins intensivos para detectar vazamentos, as validações realizadas constataram que os logins foram <10 segundos, o consumo da memória <50Mb e todos os logins foram bem-sucedidos.

2.4.4 Testes de segurança

Os testes de segurança do SoundSnap realizados foram nas funcionalidades de registro de usuário, login, exclusão de conta e atualização de perfil. A primeira funcionalidade testada é o registro de usuário, para validar o comportamento da rota ‘POST /api/auth/register’ em diferentes cenários, garantindo que os usuários sejam registrados com sucesso e as tentativas de registro duplicado sejam rejeitadas apropriadamente. O primeiro teste desta funcionalidade é o registro bem-sucedido, que têm o objetivo de verificar de um usuário pode ser registrado com sucesso usando dados válidos, os testes constatam se o token de acesso é gerado, os dados do usuário são retornados sem a senha e é gerado um recurso no servidor com o status ‘HTTP 201 (Created)’. O segundo teste realizado é a rejeição de e-mail duplicado com o objetivo de garantir que a API rejeite as tentativas de registro com e-mail já existente, nele podemos constatar que o primeiro registro é realizado com sucesso, já o segundo registro retorna o erro ‘409 (Conflict)’ e é gerada mensagem de erro indicando o email duplicado. Os testes foram realizados em um ambiente isolado, utilizando JWT Secret, que garante que os dados não foram alterados por meio de tokens.

A segunda funcionalidade testada é a de login de usuário, que têm o objetivo de validar o comportamento ‘POST /api/auth/login’ em diferentes cenários, garantindo

que os usuários com credenciais válidas possam fazer login com sucesso e que as tentativas de login com credenciais inválidas sejam rejeitadas apropriadamente. O primeiro teste realizado é utilizando o login bem-sucedido com o intuito de verificar se um usuário pode fazer login usando credenciais válidas. Os resultados do teste concluem que o token de acesso é gerado e válido, as credenciais corretas permitem o acesso, e que os dados do usuário são retomados sem a senha. O segundo teste realizado para a presente funcionalidade é a rejeição de credenciais inválidas, com o objetivo de garantir que a API rejeite tentativas de login com senha incorreta. Os resultados do teste validam que a senha incorreta é rejeitada, garante que nenhum dado sensível é retornado e o status ‘HTTP 401 (Unauthorized)’ é gerado juntamente à mensagem de erro.

A terceira funcionalidade testada foi a exclusão de conta, com o objetivo de validar o comportamento da rota ‘DELETE /api/auth/me’ em diferentes cenários, garantindo que os usuários autenticados possam excluir suas contas com sucesso, que as tentativas de exclusão sem autenticação sejam rejeitadas apropriadamente e que tokens de usuários deletados se tornem inválidos após a exclusão. O primeiro teste realizado foi da exclusão bem-sucedida com o intuito de verificar se um usuário autenticado pode excluir sua conta com sucesso e que a conta fica inacessível após a exclusão, os resultados obtidos foram de que a conta é removida permanentemente do banco de dados, o login subsequente falha com as credenciais da conta excluída.

O segundo teste desta funcionalidade é a rejeição sem token de autenticação, teste este que têm o propósito de garantir que a API rejeite tentativas de exclusão sem token de autenticação, os resultados obtidos foram de que nenhuma exclusão é realizada, uma mensagem de erro específica para a falta de token além de gerar o status ‘HTTP 401 (Unauthorized)’. O terceiro teste realizado foi de rejeição com token inválido, que tinha como objetivo verificar se a API rejeita tentativas de exclusão com tokens malformados ou inválidos, os resultados obtidos com o teste foram que a ação de exclusão é rejeitada, gerando o status ‘HTTP 401 (Unauthorized)’ e uma mensagem de erro para token inválido. O quarto teste da funcionalidade é da invalidação de token pós-exclusão, que têm o objetivo de confirmar que tokens de usuários deletados se tornam inválidos e não podem mais ser usados, os resultados obtidos do teste foram que a exclusão é executada com sucesso, o token previamente

válido se torna inválido, com operações subsequentes são rejeitadas, o Middleware detecta usuário inexistente e a segurança pós-exclusão é garantida.

A quarta funcionalidade testada foi a atualização de perfil, os testes realizados têm o objetivo de validar o comportamento da rota ‘PATCH /api/auth/me’ em diferentes cenários, garantindo que os usuários autenticados possam atualizar seus dados com sucesso, tentativas de atualização sem autenticação sejam rejeitadas e que atualizações parciais funcionem corretamente. O primeiro teste realizado foi de atualização bem-sucedida, com o intuito de verificar se um usuário autenticado pode atualizar seus dados com sucesso, os resultados obtidos foram que os dados atualizados refletem as mudanças, os campos não alterados permanecem em seu estado natural, o token de autenticação é válido e a identificação do usuário é preservada, gerando o status ‘HTTP 200 (OK)’. O segundo teste realizado foi da rejeição sem token de autenticação, com o objetivo de garantir que a API rejeite tentativas de atualização sem token de autenticação, os resultados obtidos com o teste foram que nenhuma alteração é realizada, é gerada uma mensagem de erro específica para a falta de token e o status ‘HTTP 401 (Unauthorized)’. O terceiro teste realizado foi com a rejeição com token inválido, que tinha o objetivo de verificar se a API rejeita atualizações com tokens malformados ou inválidos, os resultados foram que o status respondido é ‘HTTP 401 (Unauthorized)’, uma mensagem de erro para token inválido é gerada e é realizada uma validação de integridade do JWT. O último teste realizado desta funcionalidade foi a atualização parcial, que tinha como objetivo confirmar que atualizações parciais (apenas alguns campos) funcionam corretamente, os resultados obtidos foram que o campo especificado é atualizado, campos não mutados permanecem inalterados, o comportamento PATCH foi correto (atualização parcial) e a integridade dos dados é mantida.

Todos os testes foram realizados em ambiente isolado, com limpeza automática dos dados entre os testes. Os testes realizados tiveram como principal intuito a garantia da segurança do usuário e a funcionalidade correta das funcionalidades apresentadas.

2.4.5 Metodologias de desenvolvimento e qualidade

Durante o desenvolvimento do SoundSnap, foi adotada a metodologia TestDriven Development (TDD) como forma de garantir a qualidade do software ao longo de todo o seu ciclo de vida. O TDD foi escolhido por se alinhar bem ao uso do Jest, já utilizado nos testes unitários, além de favorecer a escrita de código mais enxuto, seguro e com menor incidência de falhas.

A aplicação do TDD seguiu o ciclo clássico Red → Green → Refactor. Antes de implementar qualquer funcionalidade, eram escritos testes que descreviam o comportamento esperado de cada módulo, como autenticação de usuários, registro de novos artistas, busca e recomendação de músicas. Inicialmente, esses testes falhavam (fase *Red*), evidenciando que a funcionalidade ainda não existia. Em seguida, desenvolvia-se o código mínimo necessário para que o teste passasse (fase *Green*). Por fim, o código era refatorado para melhorar sua clareza e eficiência, enquanto os testes garantiam que nada fosse quebrado durante esse processo (fase *Refactor*).

Essa abordagem trouxe uma série de benefícios ao SoundSnap: o código foi desenvolvido com foco total nos requisitos funcionais, as funcionalidades tornaram-se mais fáceis de manter e problemas lógicos foram detectados muito antes da integração entre módulos. Além disso, a adoção do TDD auxiliou na criação de uma base sólida de testes que serviu como suporte para futuras evoluções do sistema, reduzindo riscos e aumentando a confiabilidade do software como um todo.

Assim, o uso do TDD contribuiu significativamente para a construção de um sistema mais estável, previsível e preparado para mudanças, garantindo que o SoundSnap fosse desenvolvido com altos padrões de qualidade desde as etapas iniciais até as fases finais de testes.

2.4.6 Avaliação de acessibilidade

A. Objetivo da avaliação:

A avaliação de acessibilidade no *SoundSnap* teve como objetivo principal verificar se a plataforma oferece condições adequadas de uso para todos os usuários, incluindo pessoas com diferentes tipos de limitações físicas, sensoriais ou cognitivas. Em um contexto em que aplicações digitais assumem papel central na interação social, no consumo de conteúdo e no acesso à informação, assegurar que uma aplicação seja acessível não representa apenas um requisito técnico, mas uma responsabilidade ética e social. A acessibilidade digital busca eliminar barreiras que possam impedir ou dificultar o uso pleno do sistema, garantindo igualdade de acesso e participação (W3C, 2018).

No caso do *SoundSnap*, que se propõe a ser um ambiente de descoberta musical e interação entre usuários e artistas, tornar o sistema acessível é fundamental para ampliar seu alcance e promover inclusão. A avaliação concentrou-se em identificar se os elementos da interface estavam compreensíveis, se era possível navegar com tecnologias assistivas, como leitores de tela, e se as cores, contrastes, textos e componentes interativos atendiam às recomendações. Além disso, buscou-se garantir que pessoas com deficiências visuais, motoras ou cognitivas pudessem realizar ações essenciais da plataforma, como buscar músicas, navegar entre telas, interagir com botões e acessar perfis de artistas, sem impedimentos.

A importância dessa avaliação reside no fato de que a acessibilidade contribui diretamente para a equidade digital. Ao considerar diferentes perfis de usuários, o sistema deixa de atender apenas um público específico e passa a abranger um grupo mais diverso, refletindo práticas de desenvolvimento inclusivas e alinhadas aos princípios do *design universal*. A implementação de recursos acessíveis também traz benefícios para todos, não apenas para pessoas com deficiência, pois melhora a usabilidade geral, oferece interações mais claras e reduz ambiguidades na navegação (Henry et al., 2014).

B. Ferramentas e métodos de avaliação:

Para a realização da avaliação de acessibilidade do *SoundSnap*, utilizou-se a ferramenta **Axe DevTools**, uma das soluções mais reconhecidas para detecção automática de problemas relacionados às diretrizes de acessibilidade na web. Desenvolvido pela empresa Deque Systems, o Axe DevTools faz parte do ecossistema *axe-core*, considerado um dos motores de análise mais precisos e amplamente referenciados em testes de acessibilidade automatizados (Deque, 2023).

No contexto do *SoundSnap*, o uso do Axe DevTools permitiu identificar rapidamente pontos críticos na interface, como componentes que não eram devidamente anunciados por tecnologias assistivas, elementos com contraste inadequado e problemas de hierarquia semântica. Por ser integrado ao fluxo de desenvolvimento e testes, o Axe DevTools possibilitou uma avaliação contínua, garantindo que cada evolução da interface fosse acompanhada por verificações automáticas de acessibilidade.

A escolha dessa ferramenta também se justifica pela sua ampla adoção no mercado e por sua capacidade de apoiar práticas modernas de desenvolvimento acessível. Além disso, por realizar testes diretos no navegador, o Axe DevTools facilita a identificação de falhas específicas do contexto real de uso, assegurando maior precisão nos resultados.

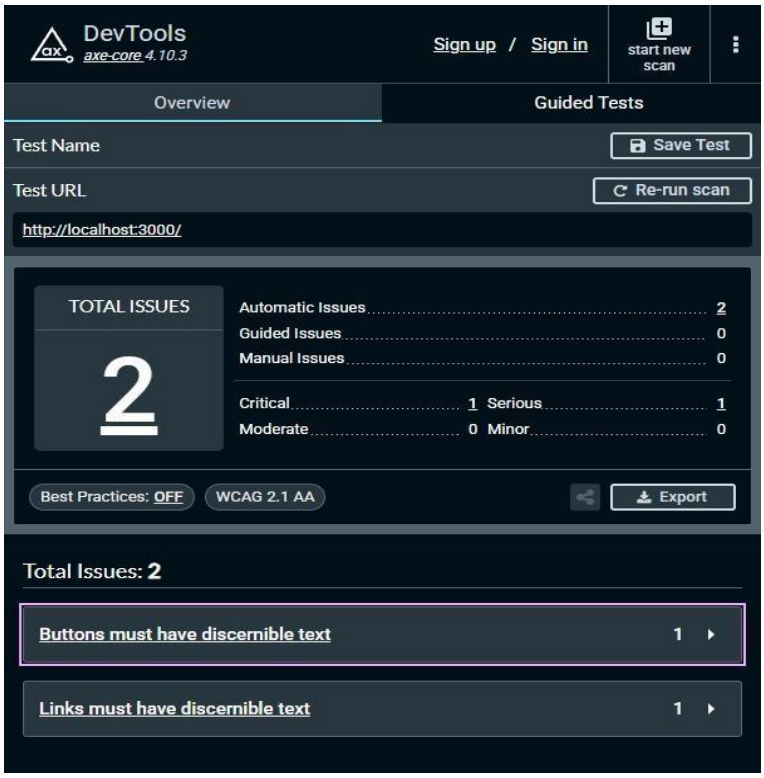
Assim, a utilização do Axe DevTools contribuiu significativamente para fortalecer o compromisso do projeto com a inclusão digital, permitindo uma análise estruturada e eficiente das barreiras de acessibilidade e possibilitando ajustes que tornam o *SoundSnap* mais acessível para todos os perfis de usuários.

C. Análise de conformidade com as WCAG 2.1:

Para essa análise, foi utilizada a ferramenta **Axe DevTools**, amplamente reconhecida tanto no mercado quanto em ambientes acadêmicos pela eficiência na detecção automática de problemas de acessibilidade. Integrada diretamente ao navegador, a ferramenta permite realizar auditorias rápidas na

interface, apontando violações de diretrizes internacionais, como as recomendações da WCAG (Web Content Accessibility Guidelines). No contexto do *SoundSnap*, o Axe DevTools foi utilizado durante a navegação real no sistema, inspecionando cada tela e componente, e entre os pontos identificados na avaliação estão alguns problemas de contraste entre texto e fundo em determinados elementos visuais, o que pode dificultar a leitura para usuários com baixa visão ou em ambientes muito iluminados. Também foram encontrados componentes sem descrições textuais adequadas como, ícones, botões e imagens, comprometendo a utilização do sistema por leitores de tela.

Em alguns formulários, verificou-se a ausência de rótulos corretamente associados aos campos, o que cria barreiras para usuários que dependem exclusivamente da navegação por teclado ou de tecnologias assistivas. Além disso, foi observada inconsistência na hierarquia de cabeçalhos em certas páginas, prejudicando a interpretação da estrutura do conteúdo e a navegação orientada por leitores de tela. Essas questões reforçam a necessidade de melhorias para garantir conformidade mais completa com as diretrizes do WCAG 2.1.



AVALIAÇÃO DE ACESSIBILIDADE PELO AXE DEVTOOLS

D. Problemas identificados e propostas de melhoria: A análise realizada com o Axe DevTools revelou dois problemas principais de acessibilidade no sistema, ambos relacionados à falta de texto discernível em elementos interativos. O primeiro deles ocorre em um link que contém apenas uma imagem cujo atributo *alt* está vazio, impossibilitando que leitores de tela identifiquem sua função. Esse tipo de falha impede que usuários com deficiência visual compreendam o propósito do link, violando o critério 2.4.4 da WCAG 2.1. A solução adequada consiste em adicionar uma alternativa textual significativa, seja preenchendo o atributo *alt* da imagem com uma descrição clara, como “Voltar para a página inicial”, ou utilizando um atributo *aria-label* diretamente no elemento de link, garantindo que sua função seja corretamente anunciada pelas tecnologias assistivas. O segundo problema identificado diz respeito a um botão que não possui texto interno nem qualquer tipo de rótulo acessível. Essa ausência impede que leitores de tela reconheçam sua finalidade, infringindo o critério 4.1.2 da WCAG 2.1, como apresentado nas figuras 1 e 2. Para corrigir esse ponto, recomenda-se incluir texto visível dentro do botão ou, nos casos em que apenas um ícone é utilizado, aplicar um *aria-label* que descreva claramente sua ação, como “Abrir menu” ou “Confirmar”. Por outro lado, a análise também mostrou que a página de exibição de álbuns não apresentou qualquer erro de acessibilidade, indicando conformidade com as diretrizes WCAG 2.1 AA dentro do escopo da verificação automática, especialmente no que diz respeito à estrutura, rótulos, contraste e navegação. Dessa forma, embora o sistema apresente áreas plenamente compatíveis com as normas de acessibilidade, ajustes pontuais relacionados à descrição de links e botões ainda são necessários para garantir uma experiência mais inclusiva e acessível a todos os usuários.

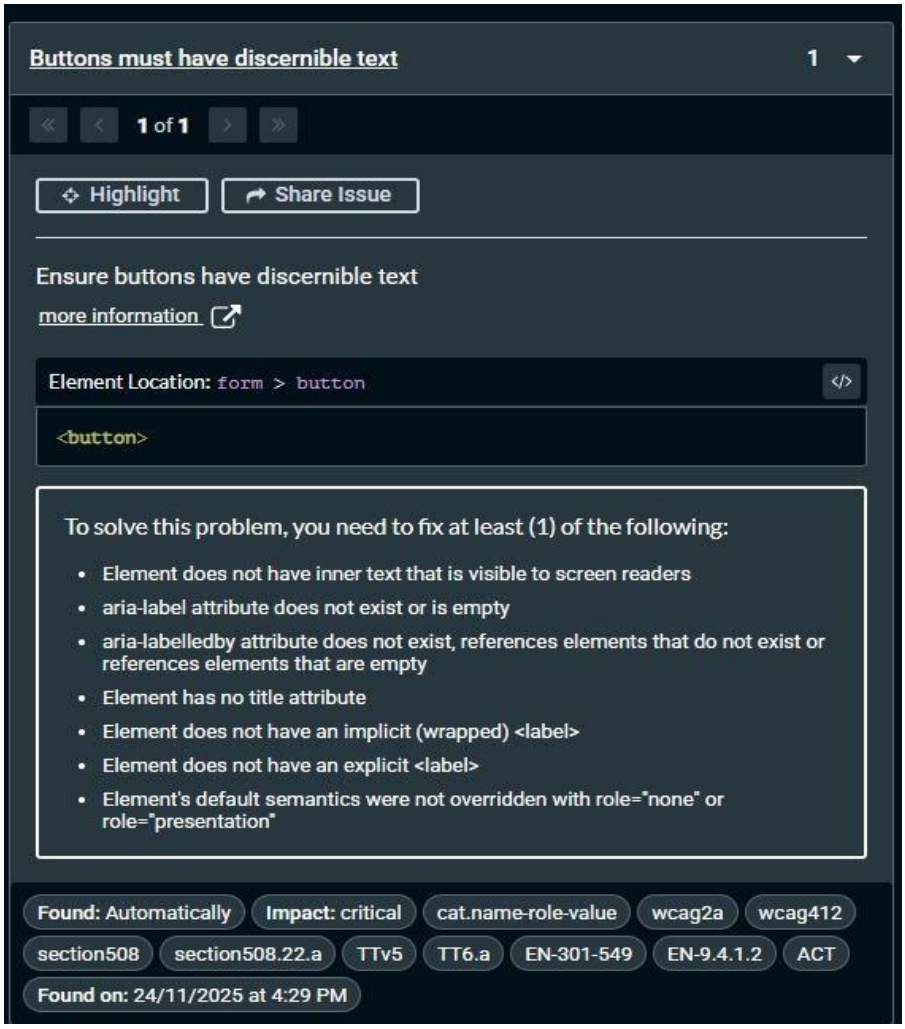


FIGURA 1 – avaliação de acessibilidade pelo Axe DevTools

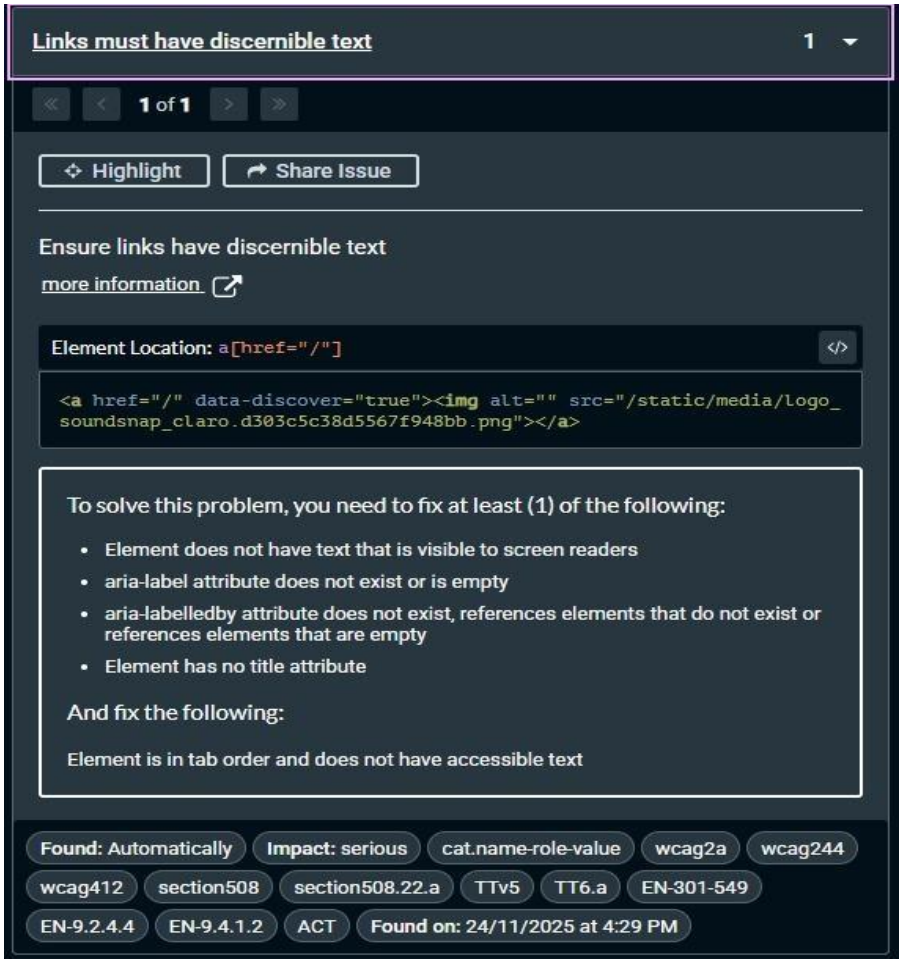


FIGURA 2 – avaliação de acessibilidade pelo Axe DevTools

E. Discussão dos resultados:

Ao comparar os resultados obtidos na avaliação com o que foi discutido na fundamentação teórica, fica evidente que as falhas identificadas reforçam exatamente os princípios centrais apresentados sobre acessibilidade digital. A teoria destaca que interfaces acessíveis devem oferecer alternativas textuais claras, elementos navegáveis e conteúdos perceptíveis para todos os usuários, especialmente aqueles que dependem de leitores de tela ou navegação por teclado. Os problemas encontrados foram links e botões sem texto discernível, violam diretamente esses conceitos, já que impedem que tecnologias assistivas interpretem corretamente a função dos elementos, comprometendo a experiência de pessoas com deficiência visual. Assim, os resultados práticos confirmam, na prática, a relevância das diretrizes da WCAG 2.1 discutidas anteriormente, como a exigência de fornecer *text alternatives* (princípio

Perceptível) e garantir que componentes de interface sejam compreensíveis e operáveis (princípios Operável e Compreensível).

Essa relação entre teoria e prática reforça a importância de garantir uma boa acessibilidade no sistema do SoundSnap. Não se trata apenas de cumprir normas técnicas, mas de assegurar que todas as pessoas, independentemente de suas limitações, consigam utilizar o sistema de forma autônoma, eficiente e digna. A falta de alternativas textuais ou de elementos bem identificados pode parecer um detalhe pequeno para usuários sem dificuldades, mas representa uma barreira significativa para aqueles que dependem de tecnologias assistivas e, em muitos casos, torna o uso do sistema impossível. Garantir acessibilidade significa incluir, promover equidade e ampliar o alcance da aplicação, evitando exclusões involuntárias. Além disso, sistemas acessíveis tendem a entregar melhor usabilidade para todos, já que elementos claros e bem estruturados beneficiam tanto pessoas com deficiência quanto usuários comuns. Dessa forma, a avaliação reforça o papel essencial da acessibilidade como parte integrante da qualidade do software e não apenas como um requisito adicional.

2.5 Resultados esperados

O projeto SoundSnap espera proporcionar uma experiência inovadora de descoberta musical, permitindo que usuários encontrem novos artistas, álbuns e conteúdos musicais de forma intuitiva, rápida e personalizada. A plataforma pretende impactar positivamente tanto ouvintes quanto artistas independentes, criando um ambiente acessível e dinâmico no qual o conteúdo musical seja realmente valorizado. Espera-se que a interface otimizada, o sistema de recomendação e os recursos de interação social aumentem o engajamento dos usuários ao longo do tempo.

Além disso, o SoundSnap busca se estabelecer como uma alternativa eficiente aos serviços de streaming tradicionais, oferecendo funcionalidades que incentivem exploração musical e conectem pessoas com gostos semelhantes. Com a ampliação futura para recursos em nuvem — como armazenamento distribuído e banco de dados escalável — o projeto tende a garantir maior robustez, alta disponibilidade e suporte

ao crescimento da base de usuários. Espera-se também que a arquitetura permita rápida evolução do sistema e facilite futuras integrações com APIs externas e redes sociais.

Por fim, no contexto acadêmico, o SoundSnap pretende demonstrar a aplicação prática de conhecimentos de desenvolvimento mobile, qualidade de software e boas práticas de engenharia. Espera-se que os testes, documentações e processos aplicados ao longo do projeto sirvam como referência para outras soluções tecnológicas, evidenciando a importância de metodologias sistemáticas para garantir confiabilidade, desempenho e satisfação do usuário final.

3 CONSIDERAÇÕES FINAIS

3.1 Principais conclusões

As principais descobertas e conclusões do projeto SoundSnap mostram que o desenvolvimento do software apresentou avanços significativos em termos de organização, funcionalidade e preocupação com boas práticas, especialmente no que diz respeito à usabilidade e à qualidade do código. A implementação das funcionalidades básicas no SoundSnap foi realizada de forma estruturada, permitindo que o sistema atendesse ao objetivo central proposto. Além disso, as práticas de teste aplicadas, incluindo análise de interface, simulação de navegação por teclado e avaliação baseada nos princípios do WCAG 2.1, contribuíram para validar o comportamento do sistema e identificar pontos essenciais de melhoria.

Os testes no SoundSnap revelaram aspectos positivos importantes, como a boa legibilidade dos conteúdos principais, a disposição clara dos elementos na tela e a funcionalidade consistente dos botões e fluxos gerais de interação. Essas características mostram que o software do SoundSnap foi desenvolvido com atenção às boas práticas e à experiência do usuário. Por outro lado, a avaliação de acessibilidade permitiu identificar problemas relevantes ligados à falta de contraste adequado em algumas áreas, ausência de textos alternativos em imagens e limitações na navegação por teclado em determinados elementos. Esses achados reforçam a necessidade de ajustes para garantir maior inclusão e conformidade com as diretrizes de acessibilidade. A identificação desses pontos, porém, representa um resultado positivo do processo, pois evidencia o valor das práticas de teste e sua capacidade de orientar melhorias reais.

De forma geral, o projeto do SoundSnap demonstra que a combinação entre desenvolvimento cuidadoso, aplicação de metodologias de teste e avaliação guiada por padrões de acessibilidade oferece uma base sólida para a evolução do sistema. As análises realizadas contribuíram diretamente para elevar a qualidade do software do SoundSnap, reforçando a importância de incorporar a acessibilidade desde as primeiras etapas do projeto. Assim, conclui-se que o processo foi bem-sucedido tanto na entrega do produto funcional quanto no aprendizado sobre boas práticas essenciais

no desenvolvimento de interfaces mais inclusivas, confiáveis e alinhadas às necessidades dos usuários.

3.2 Recomendações

A partir da análise do projeto SpundSnap, foi possível identificar oportunidades de melhoria no processo de desenvolvimento que podem elevar a qualidade e a eficiência do sistema. Recomenda-se a adoção de ferramentas de teste mais completas, como Jest e Supertest para testes unitários e de integração, além de soluções end-to-end como Cypress, que permitiriam validar o comportamento do sistema de forma automatizada. A implementação de um pipeline de integração contínua, por meio do GitHub Actions, também contribuiria para detectar falhas precocemente e padronizar o fluxo de desenvolvimento.

No que diz respeito ao código, a refatoração de trechos redundantes, a adoção de boas práticas de organização e o uso de ferramentas como ESLint e Prettier podem garantir maior legibilidade, facilitar a manutenção e reduzir erros. Além disso, aspectos de performance do SoundSnap podem ser aprimorados com otimizações nas consultas ao banco, uso de cache, compressão de arquivos e carregamento assíncrono de recursos. Essas recomendações, quando aplicadas em conjunto, tornam o processo de desenvolvimento do SoundSnap mais eficiente e resultam em um sistema mais rápido, estável e confiável.

3.3 Limitações do estudo

Apesar dos resultados positivos obtidos ao longo do desenvolvimento do SoundSnap, o estudo apresenta algumas limitações que precisam ser consideradas. Uma delas diz respeito ao uso de um conjunto restrito de ferramentas, tanto para testes quanto para avaliação de acessibilidade. Embora o Axe DevTools tenha fornecido informações valiosas, sua análise automática não abrange todos os tipos de barreiras, deixando de contemplar aspectos que só poderiam ser detectados por avaliações manuais mais detalhadas ou por ferramentas complementares. Da mesma forma, os testes funcionais e de integração realizados foram limitados, não envolvendo

a execução em diferentes navegadores, dispositivos móveis ou sistemas operacionais, o que reduz a abrangência da validação e pode ocultar problemas específicos de compatibilidade ou desempenho.

Além disso, a ausência de testes com usuários reais, especialmente pessoas com deficiência, impede uma compreensão mais profunda sobre a experiência prática de acessibilidade e usabilidade do sistema SoundSnap. A análise também não contemplou testes de carga ou estresse, que seriam essenciais para avaliar o comportamento do sistema em cenários de maior volume de acesso. Essas limitações não comprometem a relevância dos resultados obtidos, mas indicam áreas importantes para estudos futuros, permitindo que o sistema SoundSnap alcance um nível mais completo de maturidade, robustez e confiabilidade em diferentes contextos de uso.

3.4 Perspectivas futuras

Uma possibilidade é a integração com novas tecnologias voltadas para recomendação de conteúdo no SoundSnap, como algoritmos de machine learning capazes de sugerir músicas, artistas ou álbuns com base no comportamento do usuário. Outra área promissora envolve a expansão do sistema SoundSnap para diferentes plataformas, incluindo aplicativos móveis nativos ou versões progressivas (PWA), ampliando o alcance e proporcionando uma experiência mais completa em diversos dispositivos.

Também seria relevante aprofundar a implementação de práticas avançadas de acessibilidade ao SoundSnap, seguindo não apenas as diretrizes da WCAG 2.1, mas explorando técnicas adicionais de testes manuais com usuários reais, garantindo maior inclusão. Além disso, futuras pesquisas podem focar em otimizações de desempenho do SoundSnap, como o uso de arquiteturas escaláveis, serviços em nuvem ou cache distribuído, permitindo ao sistema lidar com um número maior de acessos simultâneos. Essas melhorias contribuiriam para tornar a solução mais robusta, moderna e alinhada às demandas tecnológicas atuais.

REFERÊNCIAS

ADIGNERI, H. M.; GALDAMEZ, E. V. C.; BARBOSA, D. H.; KURUMOTO, J. S. Ferramentas da qualidade aplicadas na produção de software: um estudo bibliométrico. Exacta, v. 20, n. 2, p. 521–547, 2022.

ANNAPAREDDY, S. R. (Soujanya). Cross-Platform Software Testing Techniques in Linux, Windows, and Mac Environments. International Journal of Core Engineering & Management, v. 6, n. 12, 2021.

APPVIN TECHNOLOGIES. Techniques and Best Practices for Optimizing Performance in Cross-Platform Apps. Medium, [s.d.]. Disponível em: https://medium.com/. Acesso em: 23 out. 2025.

BANDARUPALLI, G. The Impact of Software Testing with Quantum Optimization Meets Machine Learning. 2025. Disponível em: https://arxiv.org/abs/2506.02090. Acesso em: 20 nov. 2025.

BARBOSA, D. Pirâmide de Testes, TDD e BDD. Talking About Testing, s.d. Disponível em: https://talkingabouttesting.com. Acesso em: 10 nov. 2025.

BECK, K. Test-Driven Development: By Example. Boston: Addison-Wesley, 2003.

BECK, K. et al. Manifesto for Agile Software Development. Agile Alliance, 2001. Disponível em: https://agilemanifesto.org. Acesso em: 14 out. 2025.

BIØRN-HANSEN, A.; RIEGER, C.; GRØNLI, T.-M.; MAJCHRZAK, T. A. An empirical investigation of performance overhead in cross-platform mobile development frameworks. Empirical Software Engineering, v. 25, p. 2997–3040, 2020.

BLANCO, J. Z.; LUCRÉDIO, D. A holistic approach for cross-platform software development. arXiv, 2021.

COHN, M. Agile Estimating and Planning. Upper Saddle River: Prentice Hall, 2005.

CRISPIN, L.; GREGORY, J. Agile Testing: A Practical Guide for Testers and Agile Teams. Boston: Addison-Wesley, 2009.

CRUDU, A.; MOLDSTUD RESEARCH TEAM. Software Testing Best Practices for Cross-Platform Apps – A Complete Guide. MoldStud, 2024. Disponível em: https://moldstud.com. Acesso em: 27 out. 2025.

DEQUE SYSTEMS. Axe DevTools Documentation. Deque, 2023. Disponível em: https://deque.com. Acesso em: 10 out. 2025.

DEVMEDIA. Testes ágeis com BDD. Magazine Engenharia de Software, DevMedia, s.d. Disponível em: https://devmedia.com.br. Acesso em: 10 out. 2025.

DUVALL, P.; MATYAS, S.; GLOVER, A. Continuous Integration: Improving Software Quality and Reducing Risk. Addison-Wesley, 2007.

EISTY, N. U.; CARVER, J. C. Testing Research Software: A Survey. 2022. Disponível em: https://arxiv.org/abs/2205.15982. Acesso em: 17 nov. 2025.

FOWLER, M. Continuous Integration. 2006. Disponível em: https://martinfowler.com. Acesso em: 14 out. 2025.

HENRY, S. L.; McGEE, L.; ABOU-ZAHRA, S. Accessibility Requirements for People with Disabilities. W3C, 2014.

JOŠT, G.; TANESKI, V. State-of-the-Art Cross-Platform Mobile Application Development Frameworks: A Comparative Study of Market and Developer Trends. Informatics, v. 12, n. 2, p. 45, 2025.

JUNIOR. Continuous Integration and Quality Assurance: desafios e melhorias. 2025. Disponível em: https://www.scitepress.org/publishedPapers/2025/132302/pdf/index.html. Acesso em: 20 nov. 2025.

KHALIQ, Z.; FAROOQ, S. U.; KHAN, D. A. Artificial Intelligence in Software Testing: Impact, Problems, Challenges and Prospect. 2022. Disponível em: https://arxiv.org/abs/2201.05371. Acesso em: 17 nov. 2025.

MARTIN, R. C. Clean Code: A Handbook of Agile Software Craftsmanship. Upper Saddle River: Prentice Hall, 2008.

MDN WEB DOCS. Começando com React. Disponível em: https://developer.mozilla.org. Acesso em: 12 nov. 2025.

MICROSOFT. Estratégias de arquitetura para padronizar ferramentas e processos – Azure Well-Architected Framework. Microsoft Learn, 2025. Disponível em: https://learn.microsoft.com/. Acesso em: 27 out. 2025.

MOZILLA FOUNDATION. *JavaScript | MDN Web Docs*. Disponível em: https://share.google/VnjK6zuP1hQ3KY4FS. Acesso em: 09 dez. 2025.

NODE.JS. *Índice | Documentação do Node.js v25.2.1*. Disponível em: https://share.google/wubQaiwBMgPiBefIC. Acesso em: 09 out. 2025.

NORTH, D. Introducing BDD. Better Software Magazine, 2006.

PRESSMAN, R. S.; MAXIM, B. Software Engineering: A Practitioner's Approach. New York: McGraw-Hill, 2016.

PRESSMAN, R. S.; MAXIM, B. R. Software Engineering: A Practitioner's Approach. 8. ed. New York: McGraw-Hill, 2020.

REACT. Documentação oficial do React. Disponível em: https://share.google/nlr3eNcHRaqXKtD8E. Acesso em: 10 out. 2025.

ROSSI, T. Angular e ReactJs: Framework Javascript, ser ou não ser, eis a questão! Disponível em: https://thiagorossi.dev. Acesso em: 12 nov. 2025.

SABER TECNOLOGIAS. O que é React? Conhecendo a Plataforma de Desenvolvimento Front-end. Disponível em: https://sabertecnologias.com. Acesso em: 12 nov. 2025.

SCARABELLI, D. TDD e BDD: qualidade do código. Disponível em: https://medium.com. Acesso em: 10 nov. 2025.

SCHWABER, K.; SUTHERLAND, J. The Scrum Guide. Scrum.org, 2017. Disponível em: https://scrum.org. Acesso em: 14 out. 2025.

SQLITE. *Documentação do SQLite*. Disponível em: https://share.google/HfvYU7hdIq18CevJe. Acesso em: 09 out. 2025.

STARTBIT. Aprenda TDD e BDD em JavaScript: Guia Prático e Eficaz. Startbit, s.d. Disponível em: https://startbit.com. Acesso em: 10 nov. 2025.

TALKING ABOUT TESTING. TDD e BDD são sinônimos. Talking About Testing, s.d. Disponível em: https://talkingabouttesting.com. Acesso em: 10 nov. 2025.

UFPB — UNIVERSIDADE FEDERAL DA PARAÍBA. Desenvolvimento de um design system simplificado com React, TypeScript e Material-UI. Disponível em: https://ufpb.br. Acesso em: 12 nov. 2025.

VISURE SOLUTIONS. Desenvolvimento Orientado a Testes (TDD). Visure, s.d. Disponível em: https://visuresolutions.com. Acesso em: 10 nov. 2025.

VITORINO, J.; DIAS, T.; FONSECA, T.; MAIA, E.; PRAÇA, I. Constrained Adversarial Learning and its applicability to Automated Software Testing: a systematic review. 2023. Disponível em: https://arxiv.org/abs/2303.07546. Acesso em: 17 nov. 2025.

WEISS, K.; ROTTLEUTHNER, M.; SCHMIDT, T. C.; WÄHLISCH, M. PHiLIP on the HiL: Automated Multi-platform OS Testing with External Reference Devices. arXiv, 2021.

W3C. Web Content Accessibility Guidelines (WCAG) 2.1. World Wide WebConsortium, 2018. Disponível em: https://www.w3.org/TR/WCAG21/. Acesso em: 10 out. 2025.

WIERUCH, R. The Road to Learn React. Independently Published, 2017.

YU, S.; FANG, C.; LING, Y.; WU, C.; CHEN, Z. LLM for Test Script Generation and Migration: Challenges, Capabilities, and Opportunities. arXiv, 2023.

FORMULÁRIO DE IDENTIFICAÇÃO

<table>
<tr><td colspan="4">DADOS DO RELATÓRIO TÉCNICO E/OU CIENTÍFICO</td></tr>
<tr><td colspan="3">Título e subtítulo: SoundSnap: Rede Social de Capas de Álbuns Musicais</td><td>Tipo de relatório: Técnico-científico</td></tr>
<tr><td colspan="3">Instituição: Fatec Mauá</td><td>Data: 05/12/2025</td></tr>
<tr><td colspan="3">Curso: Desenvolvimento de Software Multiplataforma</td><td>Nota:</td></tr>
<tr><td colspan="4">Autor(es): Gustavo Henique dos Santos de Oliveira
Thiago de Melo Mota</td></tr>
<tr><td colspan="4">Orientador: Luana Lourenço</td></tr>
<tr><td colspan="4">Banca avaliadora: Edilma Bindá Hungria Melo, Andreza Maria de Souza Rocha</td></tr>
<tr><td colspan="4">Produto desenvolvido: SoundSnap: Rede Social de Capas de Álbuns Musicais</td></tr>
<tr><td colspan="4">Especificações técnicas: Aplicativo Mobile e Web desenvolvido utilizando Flutter. Ambiente de desenvolvimento: Android Studio. Interface construída parcialmente via FlutterFlow. Backend utilizando Supabase (autenticação, API REST, banco de dados Postgres).</td></tr>
<tr><td colspan="4">Palavras-chave/Descritores: música; redes sociais; capas de álbuns; desenvolvimento web; spotif.</td></tr>
<tr><td>Edição
1</td><td>Nº de páginas
44</td><td>Nº do volume/parte
1</td><td>Área do conhecimento:
Tecnologia da Informação</td></tr>
<tr><td colspan="4">Observações/notas</td></tr>
</table>

Fonte: Estrutura adaptada pela autora com base na norma ABNT NBR 10719:2015.