Revista Eletrônica de Iniciação Científica em Computação, 2025, 23:1
ISSN: 3085-8461 • doi: 10.5753/reic.2025.5676

Artigo de Pesquisa/Research Paper

Um Sistema Web para Gestão Eficiente de Testes em Projetos de Desenvolvimento de Software

A Web System for Efficient Test Management in Software Development Projects

Erick Silva Kokubum [Universidade Federal da Bahia | *erick.kokubum@ufba.br*]
Guilherme Souza Brandão [Universidade Federal da Bahia |*guilhermebrandao@ufba.br*]
Vítor Hugo Barbosa dos Santos [Universidade Federal da Bahia |*vitorbarbosa@ufba.br*]
Frederico Araújo Durão [Universidade Federal da Bahia |*fdurao@ufba.br*]

Instituto de Computação, Universidade Federal da Bahia, Avenida Milton Santos, s/n, Ondina, Salvador, BA, 40170-110, Brasil.

Resumo. O desenvolvimento de software em ambientes com múltiplas equipes apresenta desafios significativos, especialmente no que diz respeito à gestão eficiente dos testes. A garantia da qualidade do software é crucial para o sucesso do projeto, e um sistema de gerenciamento de testes eficaz desempenha um papel fundamental nesse processo. A complexidade de projetos de grande escala intensifica-se com a participação de múltiplas equipes, exigindo coordenação precisa e comunicação transparente para garantir a integração adequada e identificar, de forma precoce, falhas ou incompatibilidades entre os componentes desenvolvidos. É nesse contexto que esta pesquisa se insere, propondo o ICTEST: uma resposta específica aos desafios enfrentados por equipes distribuídas. O sistema fornece uma plataforma integrada e adaptável, que visa simplificar e aprimorar os processos de teste nos mais diversos ambientes.

Abstract. Software development in multi-team environments presents significant challenges, especially with regard to efficient test management. Ensuring software quality is crucial to the success of the project, and an effective test management system plays a key role in this process. The complexity of large-scale projects intensifies with the participation of multiple teams, requiring precise coordination and transparent communication to ensure proper integration and identify faults or incompatibilities between the components developed at an early stage. This is the context of this research, which proposes ICTEST: a specific response to the challenges faced by distributed teams. The system provides an integrated and adaptable platform that aims to simplify and improve testing processes in the most diverse environments.

Palavras-chave: Testes, Qualidade, Gestão, Monitoramento, Falhas.

Keywords: Testing, Quality, Management, Monitoring, Failures.

Recebido/Received: 12 March 2025 • **Aceito/Accepted:** 01 August 2025 • **Publicado/Published:** 08 August 2025

1 Introdução

No cenário contemporâneo de desenvolvimento de software, o gerenciamento eficaz de testes é a base para garantir a qualidade e a confiabilidade de uma aplicação. Os testes desempenham um papel fundamental no ciclo de vida de desenvolvimento, servindo como um mecanismo crítico para assegurar que as funcionalidades implementadas estejam alinhadas com os requisitos especificados e funcionem conforme o esperado.

Conforme descrito nas práticas padrão do setor, como as definidas por Black *et al*. [2007], o gerenciamento de testes abrange um espectro de atividades, incluindo planejamento e controle, análise e design, implementação e execução, até o fluxo final de avaliação e completude dos testes. Essa abordagem holística visa alcançar uma cobertura abrangente e facilitar a detecção oportuna de falhas, mitigando riscos e melhorando a qualidade do produto.

É válido ressaltar que o gerenciamento de testes não é apenas um exercício processual, mas um imperativo estratégico para organizações de software modernas. Ele tem como objetivo aumentar a eficiência dos processos e garantir que os requisitos de usabilidade das aplicações sejam atendidos, diferenciando as utilizações genéricas das específicas e direcionadas ao contexto organizacional. De acordo com o Institute of Electrical and Electronics Engineers [1990], o gerenciamento eficaz de processos, como os testes, traduz-se na necessidade de coordenação, direção e controle de ferramentas para otimizar a eficiência e a qualidade dos serviços executados. Ao adotar metodologias estabelecidas, como Agile e DevOps, as organizações podem integrar os testes perfeitamente aos seus fluxos de trabalho de desenvolvimento, promovendo uma cultura de melhoria contínua e colaboração.

Apesar de sua importância, muitas empresas enfrentam desafios significativos na gestão eficaz de testes, especialmente em ambientes caracterizados por múltiplos projetos e equipes dispersas. Nesses cenários, é comum a ausência de equipes dedicadas à Garantia de Qualidade (QA), transferindo a responsabilidade pela execução dos testes diretamente para os desenvolvedores. Essa falta de recursos especializados, aliada à complexidade de coordenar os esforços dos testes em diversos projetos, frequentemente resulta em ineficiências e lacunas na cobertura dos testes.

De acordo com a pesquisa conduzida por Bensten *et al*. [2020], apenas 40% dos representantes possuem um time especializado em qualidade, responsável pelos fluxos de teste, e cerca de 19% afirmam realizar poucos ou nenhum teste em suas aplicações, destacando uma lacuna preocupante no direcionamento adequado dos esforços e na presença de conhe-

cimentos especializados em muitas organizações.

Além disso, Walgude and Natarajan [2019], outra pesquisa realizada pelo mesmo grupo, revela que cerca de 56% dos representantes relatam a falta de um ambiente de teste e dados apropriados, um aumento de 13% desde 2016. Adicionalmente, 43% enfrentam dificuldades em aplicar testes em equipes que seguem a metodologia Agile. Essa disparidade crescente apenas ressalta a necessidade crítica de investimento das organizações em infraestruturas e recursos robustos para apoiar o processo de desenvolvimento e manutenção dos testes.

Em resposta a esses desafios, há uma demanda crescente por soluções inovadoras de gerenciamento de testes que simplifiquem os processos e capacitem as equipes a fornecer software de alta qualidade. Dessa forma, este artigo, que constitui um trabalho de conclusão de curso para a obtenção do título de Bacharel em Ciência da Computação pela Universidade Federal da Bahia, visa apresentar uma ferramenta web que oferece uma solução direcionada aos desafios enfrentados no gerenciamento de testes em múltiplos projetos. A ferramenta fornece uma interface amigável e recursos colaborativos, auxiliando as equipes na melhoria da eficiência dos testes enquanto garante a qualidade do produto.

2 Referencial Teórico

Esta seção explora conceitos fundamentais em testes de software, bem como os fundamentos teóricos de sistemas web para gerenciamento de testes, que servem como base para o desenvolvimento do ICTEST.

2.1 Teste de Software e Garantia de Qualidade

O teste de software é uma tarefa essencial para garantir a qualidade e a confiabilidade de sistemas. Pezzè and Young [2008] definem qualidade de software como a realização da correção, confiabilidade e robustez, alcançadas por meio de atividades integradas de verificação e validação. Essas atividades abrangem todo o processo de desenvolvimento, desde o levantamento de requisitos até a manutenção, e são fundamentais para identificar e corrigir falhas de forma antecipada. Princípios como sensibilidade, redundância e restrição, também propostos por Pezzè and Young [2008], reforçam a importância de projetar sistemas em que falhas sejam facilmente detectáveis, suposições sejam explicitamente mantidas e a complexidade seja reduzida por meio de restrições.

Os pesquisadores da Silva Lima and Barreiros [2014] destacam a necessidade de abordagens sistemáticas para técnicas de teste, como o particionamento em classes de equivalência e a análise de limites. Essas técnicas são fundamentais para o design eficiente de casos de teste, otimizando a identificação de erros e minimizando o esforço de execução.

Além disso, conforme definido na norma ISO/IEC/IEEE 29119 [2021], o processo de testes de software é composto por atividades padronizadas, organizadas nas seguintes etapas: (1) planejamento dos testes, onde são definidos os objetivos, estratégias, critérios e recursos necessários; (2) projeto e especificação dos casos de teste, baseando-se em requisitos funcionais e não funcionais; (3) implementação e execução dos testes, incluindo a preparação do ambiente e a execução dos casos planejados; (4) avaliação dos resultados e análise de cobertura; e (5) encerramento dos testes, com lições aprendidas e documentação formal do ciclo de testes.

O ICTEST incorpora esses princípios ao fornecer suporte para a fase de planejamento, execução e acompanhamento dos testes. Dessa forma, promove práticas estruturadas que favorecem a rastreabilidade e aumentam a eficiência nos diversos contextos de desenvolvimento de software.

2.2 Sistemas Web para Gerenciamento de Testes

Prasad and Gopal [2008] destacam que sistemas web para gerenciamento de testes aprimoram a colaboração, simplificam a execução de testes e melhoram o rastreamento de defeitos por meio de recursos como coordenação de atividades, monitoramento de progresso e relatórios em tempo real. O ICTEST adota essa abordagem para criar um ambiente único de gerenciamento do ciclo de vida dos testes, promovendo a colaboração em equipes e permitindo a tomada de decisões baseadas em dados e *insights* gerados por um *dashboard* detalhado.

Sistemas de gerenciamento de testes têm demonstrado benefícios significativos nos níveis técnico, gerencial e organizacional no contexto do desenvolvimento de software Prasad and Gopal [2008]. No nível técnico, sistemas eficientes produzem informações de alta qualidade, aumentando a eficácia dos testes. Em termos gerenciais, a qualidade dessas informações auxilia na tomada de decisões por parte dos gestores. Por fim, no nível organizacional, esses sistemas suportam a padronização de processos, resultando em melhores resultados de projeto. O ICTEST se alinha a essas ideias ao oferecer ferramentas que aumentam a eficácia e a confiabilidade do fluxo de testes de software, contribuindo para a melhoria contínua nos três níveis mencionados.

3 ICTEST

Nesta seção serão descritos os métodos utilizados para o desenvolvimento do trabalho juntamente com os modelos gráficos e textuais que serviram de base para a criação do sistema de gerenciamento de testes ICTEST.

3.1 Arquitetura

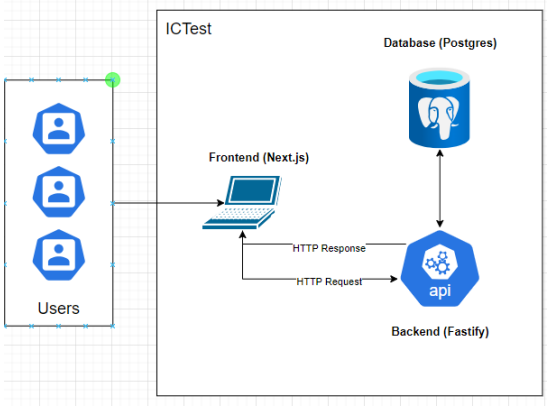

Figura 1. Arquitetura do ICTEST.

O ICTEST adota uma arquitetura cliente-servidor, composta por três camadas principais: *frontend*, *backend* e banco de dados. A camada de *frontend*, desenvolvida com TypeScript e o framework Vercel and Next.js [2021], fornece uma interface interativa para o usuário. A API *backend*, construída com Fastify [2021], atua como intermediária entre a interface e o banco de dados, processando requisições e aplicando regras de negócio. O armazenamento dos dados é feito com o PostgreSQL [2021], acessado via o ORM Prisma [2021], garantindo confiabilidade, escalabilidade e segurança na persistência das informações.

O fluxo de dados ocorre por meio de requisições HTTP: ações do usuário no *frontend* geram chamadas à API, que por sua vez realiza operações no banco de dados e retorna as respostas necessárias, atualizando a interface em tempo real.

3.2 Requisitos

A ferramenta ICTEST foi projetada com o objetivo de oferecer uma plataforma acessível e colaborativa para o gerenciamento de testes em projetos de desenvolvimento de software. Seus requisitos foram organizados em dois grupos principais: funcionais e não funcionais.

Dentre os requisitos funcionais, destaca-se a capacidade do sistema de permitir o cadastro e autenticação de usuários, bem como a criação, visualização e gerenciamento de projetos. A plataforma oferece uma listagem paginada de todos os projetos existentes, com funcionalidades para filtragem, além da possibilidade de visualizar o *board* de testes de um projeto específico. O sistema permite a criação, edição e exclusão de casos de teste, com suporte ao *upload* de imagens para situações de erro. Também oferece suporte a interações por meio de *drag-and-drop*, permitindo ao usuário alterar o estado de um caso de teste diretamente na interface. Outro requisito funcional importante é o gerenciamento de membros de projeto, possibilitando que administradores adicionem ou removam usuários conforme necessário. Por fim, um painel de visualização de dados (dashboard) está disponível para administradores, permitindo acompanhar o progresso do projeto de forma centralizada.

Em relação aos requisitos não funcionais, o sistema foi desenvolvido com foco em acessibilidade e usabilidade. A interface permite a atualização dinâmica do progresso dos casos de teste e a análise geral do andamento do projeto. A navegação foi estruturada de forma clara, visando proporcionar uma experiência fluida para o usuário. Além disso, o ICTEST foi desenvolvido para ser compatível com diferentes navegadores e dispositivos, garantindo a execução adequada de suas funcionalidades em diversas plataformas e contextos operacionais.

3.3 Detalhes das Funcionalidades

3.3.1 Tela Inicial

Após a autenticação do usuário, a página será redirecionada para a seção “Meus Projetos”, onde será listado em formato de “cards”, todos os projetos onde o usuário em questão criou ou foi adicionado.

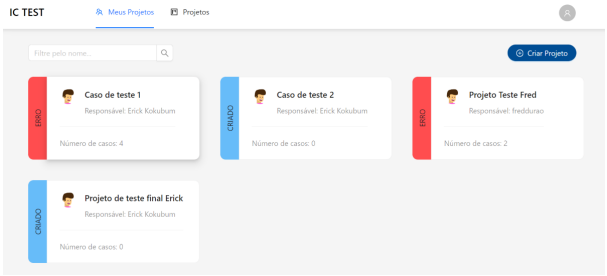

Figura 2. Tela inicial na seção “Meus Projetos”.

O usuário também pode selecionar a segunda aba existente chamada “Projetos”, onde ele será redirecionado para uma nova tela que apresentará uma listagem de todos os projetos existentes na plataforma, em um formato de tabela paginada.

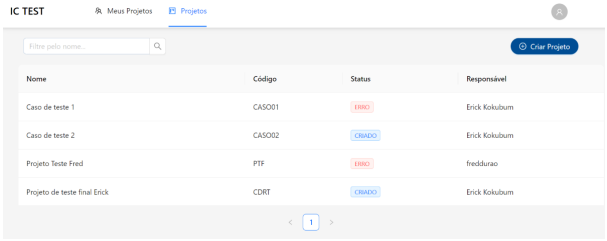

Figura 3. Tela inicial na seção “Projetos”.

3.3.2 Criar Projeto

A criação de projetos se insere na fase de planejamento da gestão de testes. Essa funcionalidade é crucial para estruturar e delimitar o escopo dos testes a serem conduzidos, definindo um espaço lógico para organização de casos, membros e resultados. O benefício principal é permitir a segmentação e rastreamento de atividades específicas por projeto, o que melhora a rastreabilidade, controle e organização das ações de teste dentro de ambientes distintos.

Para realizar a criação de um novo projeto, o usuário deverá clicar em “Criar Projeto”, na diagonal superior direita da tela inicial. Nesse momento irá aparecer um modal do lado direito da tela, com o formulário para a criação do projeto.

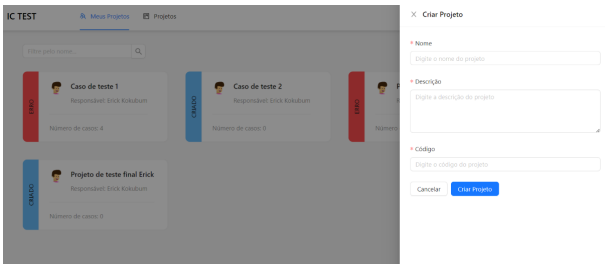

Figura 4. Modal de criação de projeto.

O formulário precisará ser preenchido com todas informações pedidas, visto que são obrigatórias. Após realizar o preenchimento, o usuário deverá clicar em “Criar Projeto” presente na parte final do modal, submetendo assim as informações. A tela irá carregar e assim que o projeto for criado, o usuário será redirecionado de volta à tela inicial em que se encontrava.

3.3.3 Casos de Teste

Para visualizar os casos de teste de um projeto específico, basta o usuário clicar diretamente no card da seção "Meus Projetos" ou no item da lista paginada da seção "Projetos", dessa forma, ele será redirecionado imediatamente para a tela dos casos de teste em sua seção principal, denominada de "*Board*".

Esta funcionalidade está relacionada às fases de projeto dos testes e execução, previstas no ciclo de vida definido pela ISO/IEC/IEEE 29119. O "*Board*" permite visualizar, gerenciar e acompanhar a evolução de cada caso de teste em tempo real, auxiliando na identificação rápida do progresso e dos gargalos do processo, promovendo transparência e agilidade na gestão do ciclo de vida dos testes.

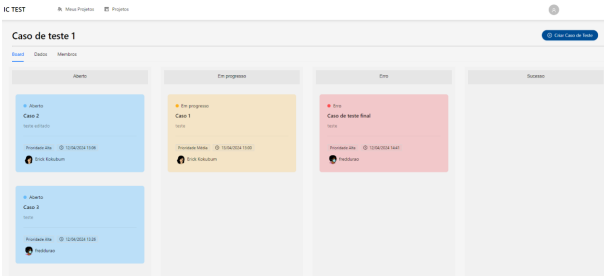

Figura 5. Tela dos Casos de Teste na seção do *Board*.

Essa parte é caracterizada pela presença de quatro campos que definem em que estado o caso de teste se encontra, podendo ser os seguintes:

- **Aberto**: O caso de teste foi criado, porém sua execução ainda não foi iniciada. Os *cards* nesse estado são representados pela cor azul;
- **Em Progresso**: O caso de teste já está sendo executado. Os *cards* nesse estado são representados pela cor amarela;
- **Erro**: O caso de teste começou a ser executado, porém foi encontrado algum erro durante esse processo. Os *cards* nesse estado são representados pela cor vermelha.
- **Sucesso**: O caso de teste foi finalizado com sucesso. Os *cards* nesse estado são representados pela cor verde.

Se o usuário desejar atualizar o estado de um caso de teste de maneira automática, basta clicar no "*card*" correspondente e arrastá-lo para o estado desejado. Por exemplo, para atualizar um caso de teste de "Aberto" para "Em Progresso", basta executar esse procedimento entre as listas dos dois estados, conforme ilustrado na imagem abaixo.

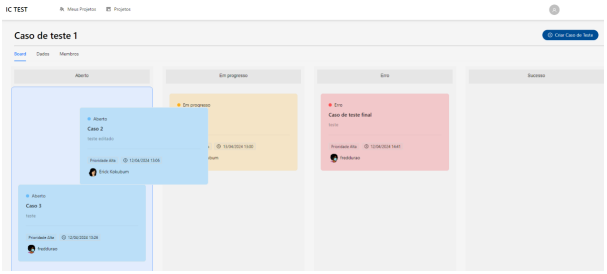

Figura 6. *Card* sendo arrastado do estado "Aberto" para "Em Progresso".

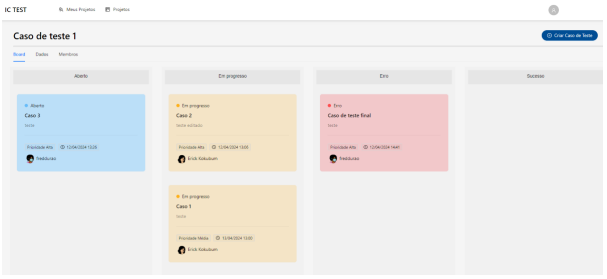

Figura 7. *Card* atualizado com seu novo estado.

Na tela dos casos de teste, além do "*Board*", é possível encontrar também a seção "Membros" e a seção "Dados", sendo essa última disponível apenas para o administrador do projeto em questão. O botão de redirecionamento dessas seções é encontrado abaixo do nome do projeto.

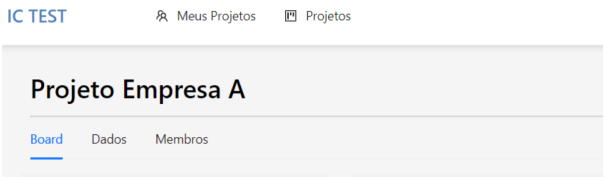

Figura 8. Seções da tela de casos de teste.

Quando o usuário selecionar "Membros", ele será direcionado para a visualização dos membros do projeto.

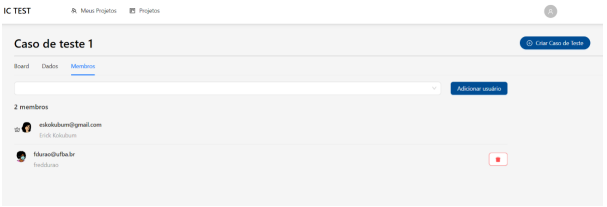

Figura 9. Tela dos Casos de Teste na seção de Membros.

Se o usuário for o administrador, terá permissão para adicionar ou remover membros, caso contrário, essas opções estarão indisponíveis. Para adicionar um membro, é necessário selecionar o usuário no menu suspenso e clicar no botão "Adicionar Usuário", presente do lado direito do menu. Para remover um membro, basta clicar no ícone vermelho da lixeira ao lado direito do membro em questão.

Essa seção de membros está vinculada à fase de planejamento, permitindo a alocação de responsabilidades aos usuários envolvidos, fortalecendo assim o nível colaborativo do time.

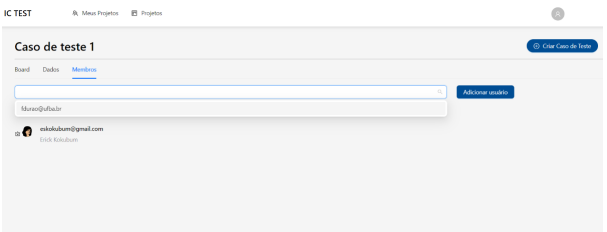

Figura 10. Tela dos Casos de Teste na seção de Membros para adição de usuário.

Quando o usuário selecionar "Dados", ele será direcionado para um *dashboard* que apresenta uma representação

gráfica do progresso e do estado do projeto como um todo. Essa seção é uma funcionalidade voltada à etapa de monitoramento e controle. Ela apresenta indicadores visuais que auxiliam o administrador a identificar o desempenho da equipe e a efetividade do processo de testes, promovendo decisões mais rápidas e baseadas em dados.

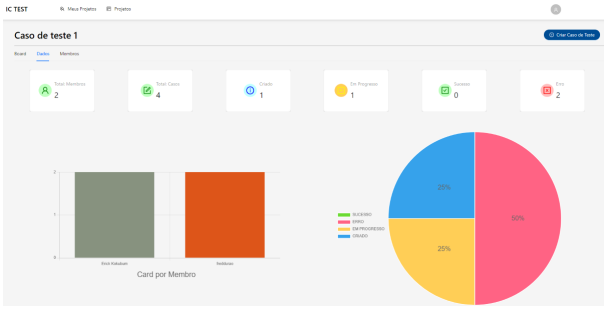

Figura 11. Tela dos Casos de Teste na seção de Dados.

Esse fluxo disponibiliza as seguintes informações:

- Número total de membros do projeto;
- Número total de casos de teste do projeo;
- Quantidade de casos de teste no estado “Aberto”;
- Quantidade de casos de teste no estado “Em Progresso”;
- Quantidade de casos de teste no estado “Erro”;
- Quantidade de casos de teste no estado “Sucesso”;
- Gráfico de barra da quantidade de casos de teste designados para cada membro do projeto;
- Gráfico de setores que representa a porcentagem dos casos de teste do projeto que se encontram em cada um dos estados disponíveis.

3.3.4 Criar caso de teste

Para criar um caso de teste, o usuário deverá clicar no botão “Criar Caso de Teste” presente na diagonal direita superior da tela inicial dos casos de teste, independente da seção.

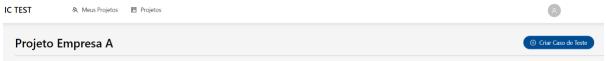

Figura 12. Botão: Criar caso de teste.

Nesse momento irá aparecer um modal do lado direito da tela, com o formulário para a criação do caso de teste.

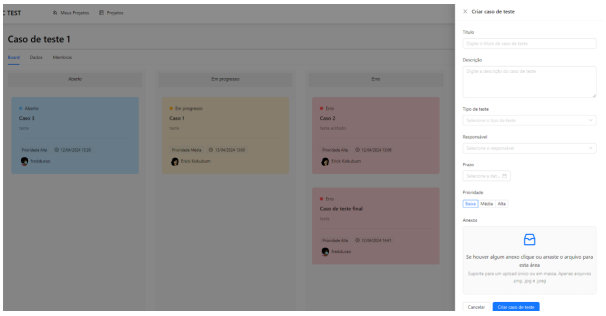

Figura 13. Modal de criação de caso de teste.

Entre as informações solicitadas, estará uma lista dos “Tipos de teste”, exibida em formato de menu contendo uma breve descrição para cada uma das opções disponíveis.

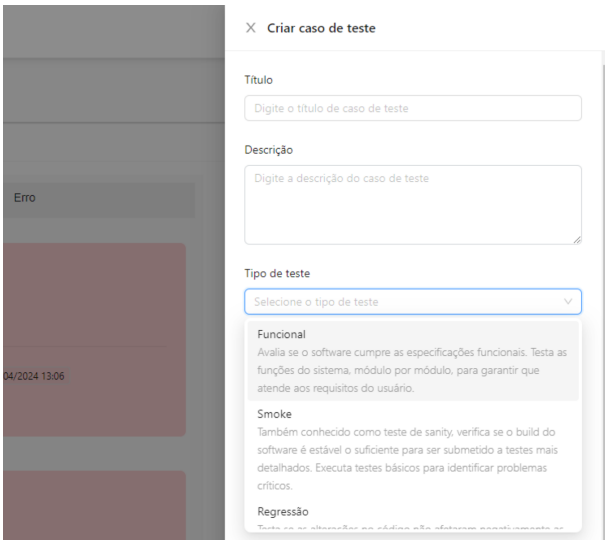

Figura 14. Menu: Tipos de teste.

Após realizar o preenchimento, o usuário deverá clicar no botão “Criar Caso de Teste” presente na parte final do modal, submetendo assim as informações. A tela irá carregar e assim que o caso de teste for criado, o usuário será redirecionado de volta a tela em que se encontrava.

3.3.5 Atualizar Caso de Teste

Conforme mencionado anteriormente, o usuário pode atualizar o estado de um caso de teste arrastando o seu cartão de uma lista de estado para outra. No entanto, além dessa forma de atualização, também é possível modificar o contexto completo do caso de teste. Para isso, o usuário deve clicar no caso de teste desejado, abrindo assim um modal contendo todas as informações associadas a ele, permitindo que o usuário faça as alterações necessárias.

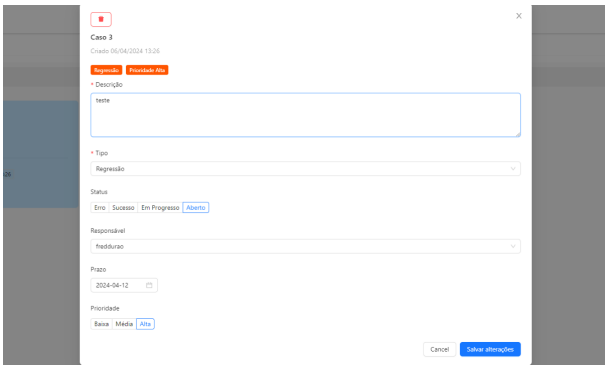

Figura 15. Modal de atualização de casos de teste.

No modal, também é disponibilizada a opção de exclusão do caso de teste. Para isso, o usuário deve clicar no ícone vermelho da lixeira, localizado acima do nome do caso de teste. Dessa maneira, o caso de teste será removido e não aparecerá mais na listagem. Para casos de teste em estado de erro, é disponibilizado para o usuário a possibilidade de definir uma descrição, juntamente com o *upload* da imagem de representação do erro. Esses campos aparecerão no momento que o usuário clicar no estado “Erro”.

Figura 16. Modal: Campos para o estado de "Erro".

Assim que a imagem for carregada, o usuário terá a opção de baixar o recurso. Para isso, basta clicar no botão "Download: Imagem do Erro", que estará disponível após o carregamento bem-sucedido.

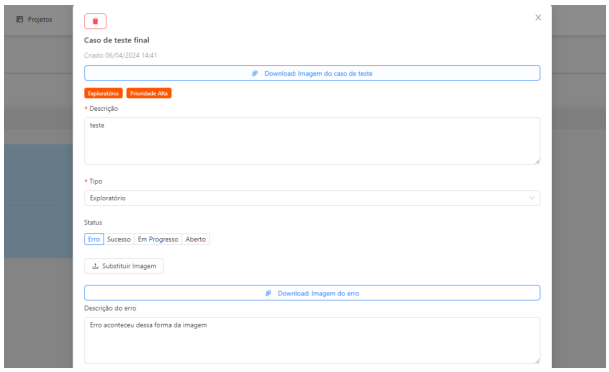

Figura 17. Modal: Download imagem do erro.

Além disso, é esperado que se uma imagem também for carregada durante a criação do caso de teste, ela estará acessível ao usuário nesse mesmo modal. Sendo assim, para fazer o download, o usuário deve simplesmente clicar no botão "Download: Imagem do caso de teste", posicionado abaixo do nome do caso de teste.

4 Avaliação Experimental

Esta seção apresenta a avaliacão experimental realizada com o objetivo de levantar aspectos a respeito da usabilidade e efetividade dos resultados apresentados pela ferramenta do ICTEST.

4.1 Metodologia

O processo de avaliação ocorreu durante o mês de agosto, onde foram convidados cerca de 38 participantes, entre eles estudantes e profissionais da área de ciência da computação, para utilizar e avaliar voluntariamente o sistema trabalhado. O experimento como um todo teve uma duração média entre 10 a 15 minutos, tendo como principal perfil de participação, profissionais da área de programação com mais de 1 ano de experiência, sendo os mesmos em sua maioria da área de *backend*, *fullstack* ou qualidade. No início do experimento todos os participantes assinaram um Termo de Consentimento Livre e Esclarecido, através do qual, apenas participantes que aceitassem os termos participaram do experimento. Não houve rejeição, porém. Em relação a análise demográfica, 68.4% dos participantes responderam que trabalhavam com programação especificamente, enquanto 31.6% trabalhavam em outras área da Tecnologia da Informação. Dentre os programadores, 43% dos participantes possuíam uma experiência acima de 3 anos no mercado.

Para facilitar a avaliação foi criado um documento abrangente o qual incluía um guia passo a passo sobre como utilizar o sistema, garantindo assim que os participantes pudessem navegar por todos os recursos sem problemas. Adicionalmente, foi disponibilizado também um vídeo informativo, com uma duração média de 4 minutos, para demonstrar visualmente as funcionalidades da aplicação, desde a criação dos casos de teste como os fluxos de gerenciamento do estado das tarefas, melhorando assim a compreensão e facilidade da utilização do mesmo.

O documento continha também um link para um formulário de avaliação (Google Forms), que foi elaborado utilizando a metodologia TAM (Technology Acceptance Model). Esse formulário era composto por cerca de dezesseis questões, sendo as iniciais para parametrizar os participantes (verificação da área de trabalho, tempo de experiência e etc), e as demais destinadas a avaliar a utilidade e a facilidade percebida do uso da aplicação, além da presença de uma pergunta discursiva para comentários adicionais de forma opcional. As perguntas foram elaboradas de forma a coletar dados quantitativos e qualitativos, fornecendo uma avaliação completa do desempenho do sistema.

Os dados recolhidos nas respostas foram analisados para identificar temas comuns e áreas de melhoria. Esta análise ajudou a compreender os pontos fortes e fracos da aplicação a partir das perspectivas dos utilizadores, fornecendo informações valiosas para o desenvolvimento e melhorias futuras.

É importante destacar algumas limitações do estudo realizado. Em primeiro lugar, o perfil dos participantes, majoritariamente composto por estudantes e profissionais em início de carreira, pode não refletir completamente o comportamento de usuários mais experientes ou de ambientes corporativos mais exigentes. Em segundo lugar, a escala de avaliação utilizada foi do tipo Likert, a qual, por ser ordinal, limita a realização de análises estatísticas mais robustas, como o cálculo do grau de confiabilidade das respostas. Por fim, o tamanho da amostra, restrito a 38 participantes, reduz a generalização dos resultados obtidos. Dessa forma, estudos complementares com amostras maiores e mais diversificadas são recomendados para fortalecer as evidências e aprofundar a avaliação da ferramenta proposta.

4.2 Resultados

Abaixo estão os resultados obtidos a partir da avaliação experimental da ferramenta do ICTEST.

4.2.1 Produtividade

Foi perguntado aos usuários se a utilização da plataforma aumentaria a produtividade no gerenciamento de testes do mesmo.

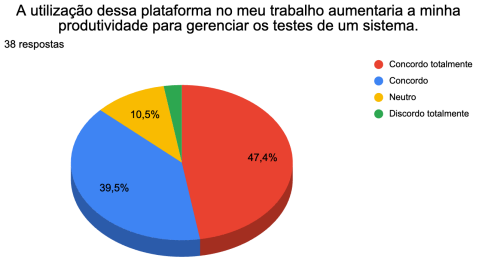

Figura 18. Produtividade.

O gráfico evidencia um resultado favorável no que diz respeito ao aumento da produtividade, com mais de 85% de aprovação dos usuários. Esse alto índice de concordância reflete o impacto positivo da plataforma em maximizar a eficiência dos processos, otimizando o tempo e os recursos disponíveis. A plataforma se destaca por impulsionar o desempenho dos usuários, possibilitando maior agilidade e qualidade no fluxo de trabalho, o que solidifica seu papel como uma ferramenta essencial para aprimorar a produtividade.

4.2.2 Velocidade de Realização dos Testes

Foi perguntado aos usuários se a utilização da plataforma ajudaria na realização dos testes de uma forma mais rápida.

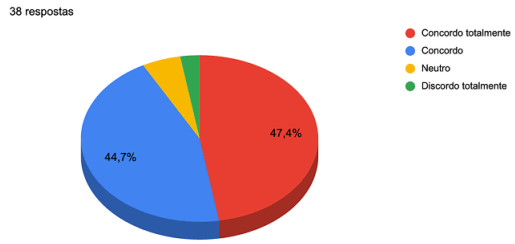

Figura 19. Velocidade de Resolução dos problemas identificados.

Observa-se um índice de mais de 90% de concordância quanto ao aumento na velocidade de resolução dos problemas identificados pelos testes, o que indica que a plataforma impacta positivamente na obtenção de soluções de forma mais rápida durante os processos de execução dos testes.

4.2.3 Performance

Foi perguntado aos usuários se a utilização da plataforma aumentaria a performance para o gerenciamento e execução dos testes de um determinado sistema.

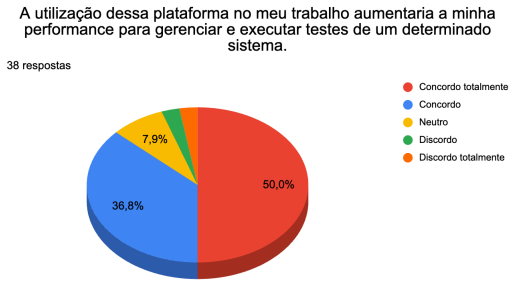

Figura 20. Performance.

Os resultados demonstraram que cerca de 85% dos usuários concordam ou concordam totalmente que a plataforma eleva a performance no gerenciamento e execução dos testes. Este alto índice de concordância reflete a percepção positiva dos usuários quanto ao impacto da plataforma em melhorar significativamente a eficiência e agilidade dos processos de teste, reforçando seu papel como uma ferramenta de otimização de desempenho.

4.2.4 Facilidade de acompanhamento.

Foi perguntado aos usuários se a utilização da plataforma tornaria mais facil o acompanhamento e execução dos testes.

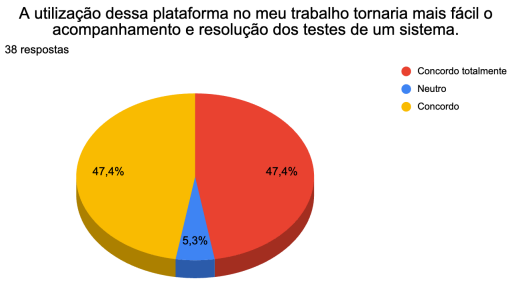

Figura 21. Facilidade de Acompanhamento.

De acordo com o gráfico, observa-se claramente uma aprovação superior a 90% em relação à facilidade de gerenciamento na utilização do sistema, sem registros de respostas negativas e com apenas 5% de feedbacks neutros. Esses resultados permitem concluir que a plataforma é altamente intuitiva no que diz respeito aos seus fluxos de gerenciamento, proporcionando uma experiência de uso eficaz e amigável.

4.2.5 Efetividade

Foi perguntado aos usuários se a utilização da plataforma aumentaria a efetividade no gerenciamento dos testes no sistema.

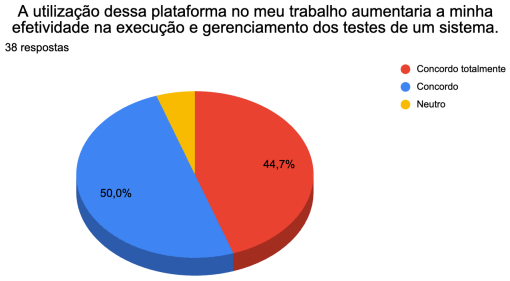

Figura 22. Efetividade.

Os resultados mostram que 50% dos usuários concordam e 44,7% concordam totalmente que a utilização da plataforma aumenta a efetividade no gerenciamento dos testes no sistema, resultando em uma aprovação global de 94,7%. A ausência de respostas de discordância reforça a percepção de que a plataforma contribui de forma substancial para melhorar a efetividade no controle e execução dos testes. Com apenas uma pequena proporção de respostas neutras, a plataforma se consolida como um recurso valioso para otimizar os processos e elevar a precisão e eficácia do gerenciamento.

4.2.6 Utilidade

Foi perguntado aos usuários se a plataforma seria útil no trabalho para gerenciar os testes.

Figura 23. Utilidade.

Pela análise gráfica, é possível perceber uma aprovação de 94,8%. Este elevado nível de aceitação evidencia que a plataforma é amplamente reconhecida como uma ferramenta valiosa e de grande utilidade para a gestão e execução dos testes no ambiente de trabalho. Com apenas uma pequena parcela de respostas neutras e discordante, fica claro que a plataforma desempenha um papel significativo em tornar os processos de gerenciamento e execução de testes mais eficientes, eficazes e organizados, contribuindo para melhores resultados no contexto profissional.

4.2.7 Facilidade de Aprendizado

Foi perguntado aos usuários se seria fácil aprender a utilizar a plataforma.

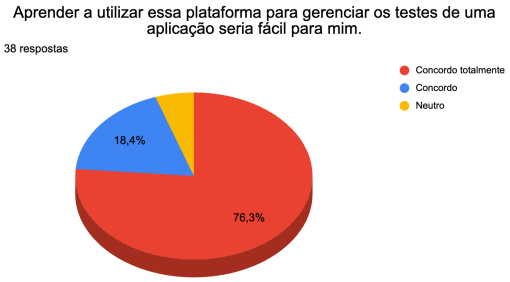

Figura 24. Facilidade de Aprendizado.

Os resultados mostram que 76,3% dos usuários concordam totalmente e 18,4% concordam que seria fácil aprender a utilizar a plataforma, resultando em uma aprovação total de 94,7%. Esse índice elevado de concordância reflete que a plataforma apresenta uma curva de aprendizado intuitiva e acessível, facilitando seu uso por novos e atuais usuários. A proporção mínima de respostas neutras reforça ainda mais a percepção de que a plataforma foi desenvolvida com foco na simplicidade e facilidade de uso, promovendo uma experiência de aprendizado prática e eficiente para gerenciar suas funcionalidades.

4.2.8 Facilidade de Uso

Foi perguntado aos usuários se eles acham fácil fazer a plataforma realizar as ações que eles buscam em relação ao gerenciamento dos testes.

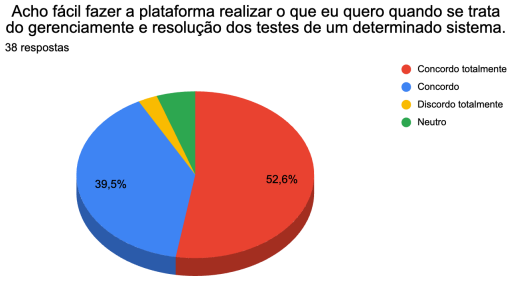

Figura 25. Facilidade de uso.

Esse alto índice de concordância, com mais de 99%, evidencia que a plataforma atende de maneira eficiente às necessidades dos usuários, permitindo que eles executem suas tarefas de forma prática e sem obstáculos. Com uma proporção muito pequena de respostas neutras e discordâncias, fica claro que a plataforma foi projetada com foco na usabilidade, garantindo que os usuários consigam alcançar seus objetivos com agilidade e precisão no gerenciamento de testes.

4.2.9 Interatividade

Foi perguntado aos usuários se a interação deles com a plataforma seria clara e tranquila.

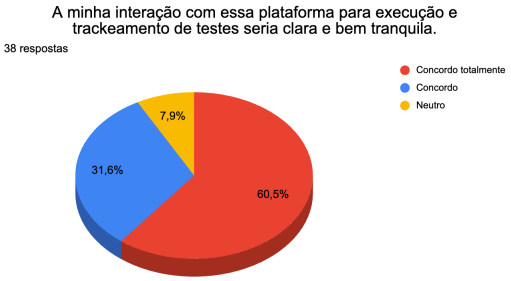

Figura 26. Interatividade.

Os resultados apontam que 60,5% dos usuários concordam totalmente e 31,6% concordam que a interação com a plataforma é clara e tranquila, totalizando 92,1% de aprovação. Esse alto nível de concordância evidencia que a experiência do uso da plataforma é marcada por clareza e facilidade, permitindo aos usuários navegar e interagir de forma serena e eficiente. A predominância de respostas positivas reforça que os fluxos interativos foram projetados para proporcionar uma experiência livre de atritos, onde os usuários podem realizar suas tarefas de maneira descomplicada e com confiança.

4.2.10 Flexibilidade

Foi perguntado aos usuários se a plataforma seria flexível para realizar as tarefas de gerenciamento e execução dos testes em questão.

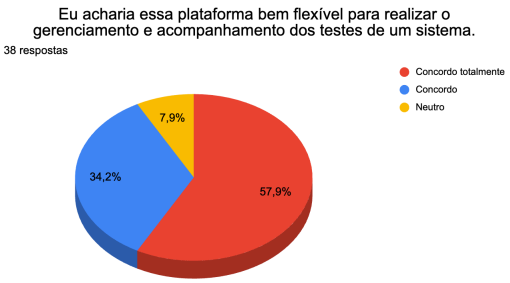

Figura 27. Flexibilidade.

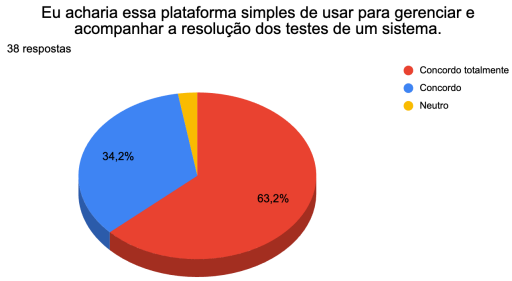

Figura 29. Simplicidade.

A avaliação dos usuários mostra uma percepção clara de que a plataforma oferece flexibilidade para a realização de tarefas de gerenciamento e execução de testes. Com uma forte predominância de respostas positivas, refletida pelos 57,9% que concordam totalmente e 34,2% que concordam, fica evidente que a plataforma consegue se adaptar às necessidades específicas dos usuários. Essa adaptabilidade permite que ela seja moldada para diferentes cenários e fluxos de trabalho, reforçando seu papel como uma solução prática e eficiente. O alto nível de aprovação destaca a capacidade da plataforma de responder às demandas com agilidade e oferecer suporte flexível no ambiente de trabalho.

4.2.11 Proficiência

Foi perguntado aos usuários se seria fácil se tornar habilidoso na utilização dessa plataforma para executar tarefas de gerenciamento de testes.

Figura 28. Proficiência.

Os usuários demonstraram uma percepção altamente positiva quanto à facilidade em se tornarem habilidosos no uso da plataforma para gerenciar e executar testes, com cerca de mais de 95% de concordância (“concordo” e “concordo totalmente”). Essa facilidade de adquirir domínio reforça a proposta da plataforma em ser uma ferramenta intuitiva, capaz de capacitar seus usuários rapidamente. Tal nível de confiança e experiência fluida torna o aprendizado mais natural, permitindo que os usuários atinjam níveis elevados de proficiência sem grandes barreiras ou dificuldades.

4.2.12 Simplicidade

Foi perguntado aos usuários se a plataforma seria simples de usar para gerenciar e acompanhar testes de um determinado sistema.

Os usuários destacaram que a plataforma é percebida como simples através de um resultado com 97,4% de aprovação. Tal fator confirma que a plataforma oferece uma experiência prática e acessível, permitindo que os usuários interajam de maneira clara e direta, sem complicações. A proposta de simplicidade contribui para uma experiência de uso clara e eficiente, concentrando os esforços nas atividades essenciais sem enfrentar barreiras desnecessárias.

4.2.13 Comentários Adicionais

Foi disponibilizado um campo para que os usuários pudessem compartilhar comentários adicionais sobre a plataforma. As respostas obtidas destacam diferentes perspectivas e experiências com o uso da ferramenta. No geral, a plataforma foi elogiada por sua facilidade de uso, interatividade intuitiva e aplicabilidade em diferentes contextos, mesmo por aqueles que não atuam diretamente na área de testes. Muitos destacaram a simplicidade e objetividade da interface, apontando que a ferramenta pode ser útil em áreas como gestão de processos e equipes.

Alguns usuários forneceram sugestões específicas, como a necessidade de manter os dados preenchidos em formulários ao clicar fora, melhorias na segurança de acesso aos projetos e opções para arquivar testes já concluídos. Além disso, foi mencionada a importância de maior integração com outras ferramentas de gestão de projetos, como o Jira, considerando que muitas empresas já possuem soluções robustas e integradas para controle de suas operações.

Outros comentários ressaltaram o potencial da plataforma para otimizar fluxos de trabalho, mesmo que, em alguns cenários, ela precise ser utilizada paralelamente a outras ferramentas já adotadas nas empresas. De forma geral, os *feedbacks* foram positivos, destacando a utilidade e a flexibilidade da plataforma, com sugestões que apontam caminhos para aprimorar ainda mais a sua eficácia e alcance no futuro.

4.2.14 Limitações

Com relação a metodologia, considero importante destacar as limitações do estudo conduzido em relação (i) ao perfil dos participantes -estudantes; (2) a escala de avaliação utilizada para coleta das respostas (likert -ordinal), que impossibilita uma análise estatística precisa dos resultados, por exemplo, identificar o grau de confiabilidade das respostas, (3) o tamanho da amostra (38 pessoas), portanto deve ser mencionado que mais estudos devem ser conduzidos no futuro.

5 Trabalhos Relacionados

Esta seção discute trabalhos relacionados no contexto de sistemas de informação voltados para o gerenciamento de testes, com o objetivo de demonstrar a eficiência prática do software implementado.

Vasconcelos *et al*. [2013] apresentaram a ferramenta Test-Module, desenvolvida para auxiliar no gerenciamento de testes em projetos que utilizam metodologias ágeis, como o Scrum. Integrada ao FireScrum, a ferramenta permite criar e organizar casos de teste, gerenciar planos e ciclos de execução, além de rastrear defeitos detectados. Destaca-se por sua interface rica, pela organização em bibliotecas reutilizáveis de testes e pela integração com ferramentas de gerenciamento de defeitos, como Bugzilla e Mantis. Em comparação, o ICTEST é uma solução mais genérica, não vinculada a metodologias específicas como o Scrum. Enquanto o Test-Module aproveita a integração com ferramentas ágeis para aumentar a eficiência dos testes, o ICTEST oferece um ambiente flexível e adaptável, permitindo gerenciar tanto testes manuais quanto automatizados. Contudo, o ICTEST não inclui suporte direto para rastreamento de defeitos ou integração com ferramentas externas.

Reis *et al*. [2012] apresentaram o VVTeste, um ambiente que integra ferramentas livres como TestLink, Condado e Mantis para a geração, execução e gerenciamento de testes de software, alinhado aos processos de verificação e validação definidos pelo MPS.Br (Melhoria do Processo de Software Brasileiro). A proposta centraliza informações em uma base de conhecimento única, permitindo a análise histórica e o aprendizado organizacional, além de oferecer suporte à geração automática de casos de teste e ao gerenciamento de defeitos.

Em comparação, o ICTEST também busca centralizar informações, mas se diferencia por ser uma solução voltada para a integração e eficiência em ambientes diversos, sem estar restrito a um conjunto específico de ferramentas ou normas, como o MPS.Br. Enquanto o VVTeste é projetado para processos de verificação e validação em conformidade com um modelo de referência, o ICTEST adota uma abordagem mais genérica e adaptável, abrangendo cenários com múltiplas equipes e demandas variadas.

Gonçalves and Silva [2020] propuseram o 4Testers, uma ferramenta destinada à documentação e gerenciamento de cenários e roteiros de testes manuais, com o objetivo de melhorar a qualidade e organização dos testes em pequenas empresas. A aplicação permite registrar cenários, associá-los a roteiros e projetos, controlar o tempo de execução e monitorar aprovações e reprovações de cenários, sendo uma solução focada exclusivamente no processo manual de testes.

Por outro lado, o ICTEST busca atender a um espectro mais amplo de necessidades, incluindo tanto testes manuais quanto automatizados, além de suportar equipes maiores e processos mais complexos. Enquanto o 4Testers oferece uma solução prática e simples voltada para empresas menores, o ICTEST propõe uma abordagem integrada e adaptável, contemplando diferentes metodologias e tamanhos de projetos, com ênfase na colaboração e centralização de informações.

Hattori *et al*. [2013] apresentaram o SPLOT-TM, uma extensão da ferramenta SPLOT projetada para gerenciar testes funcionais no contexto de Linhas de Produto de Software (LPS). A ferramenta utiliza Modelos de Características (MCs) para estruturar e associar casos de teste (CsT) às funcionalidades do sistema, oferecendo suporte tanto na Engenharia de Domínio quanto na Engenharia de Aplicação. Seu objetivo é alinhar os testes às variabilidades e características inerentes à abordagem da LPS, promovendo maior flexibilidade no gerenciamento de testes. Em comparação, o ICTEST se posiciona como uma solução mais geral, que não se limita ao contexto de Linhas de Produto de Software. Enquanto o SPLOT-TM é altamente especializado em gerenciar variabilidades em LPS usando MCs, o ICTEST prioriza a centralização e a eficiência dos processos de teste, abrangendo diferentes metodologias e contextos organizacionais. Essa distinção destaca o SPLOT-TM como uma ferramenta ideal para cenários específicos de LPS, enquanto o ICTEST busca atender a um público mais amplo, com demandas diversificadas.

Liu [1992] propôs o SEMST, um ambiente de suporte para o gerenciamento de testes de software baseado em técnicas de Gerenciamento de Configuração de Software (SCM). A ferramenta permite gerenciar versões de especificações, casos de teste e programas, além de controlar os relacionamentos entre esses componentes. Desenvolvido como um protótipo no ambiente Unix com suporte ao sistema RCS, o SEMST tem como objetivo integrar o controle de versões e a rastreabilidade no processo de testes. Em comparação, o ICTEST compartilha o objetivo de centralizar informações e otimizar o processo de testes, mas adota uma abordagem mais moderna e flexível. Ele foi projetado para operar em uma plataforma web, atendendo a diferentes metodologias e contextos. Enquanto o SEMST é focado em técnicas de SCM aplicadas ao ambiente Unix, o ICTEST é direcionado para equipes distribuídas e promove a colaboração em larga escala.

Pitschinetz and Wegener [1995] apresentaram o TESSY, uma ferramenta que automatiza testes dinâmicos de unidade e gerencia todo o ciclo de vida dos testes. A ferramenta é especialmente voltada para testes em código C e inclui funcionalidades como a execução de testes em lote, geração automatizada de drivers de teste, monitoramento de cobertura de código (branch coverage) e suporte para integração com ambientes de desenvolvimento. Além disso, o TESSY facilita a documentação automática dos testes, oferecendo uma interface gráfica amigável e orientada a objetos para gerenciar atividades específicas, como a criação de casos de teste e a configuração de ambientes de teste. Em comparação, o ICTEST não oferece automação para geração de drivers de teste ou monitoramento de cobertura de código, concentrando-se nos testes e no suporte a diferentes metodologias de desenvolvimento. Enquanto o TESSY é projetado para aumentar a eficiência em testes de unidade e integração, o ICTEST prioriza a integração de equipes e a centralização de informações em uma plataforma web. Além disso, o ICTEST permite gerenciar múltiplos projetos simultaneamente e oferece um fluxo de trabalho voltado para colaboração e rastreabilidade, embora não inclua recursos específicos para cobertura de código ou automação de testes em nível de unidade.

Generosi *et al*. [2022] apresentaram o Miora, uma pla-

taforma para avaliação remota de usabilidade de aplicações web. O sistema combina análise de emoções e métricas de interação, utilizando algoritmos de aprendizado profundo para capturar dados sobre o comportamento e as emoções dos usuários durante as interações. A ferramenta fornece gráficos e estatísticas detalhadas, permitindo que moderadores analisem a usabilidade de forma eficiente. O Miora foi validado em estudos de caso que demonstraram sua eficácia em testes formativos e somativos. Em comparação com o ICTEST, o Miora oferece um foco maior na avaliação de usabilidade baseada em métricas comportamentais e emocionais, enquanto o ICTEST concentra-se no gerenciamento geral do processo de testes, abrangendo planejamento, execução e análise de resultados. Embora o Miora seja mais voltado para a avaliação da experiência do usuário, o ICTEST pode integrar aspectos semelhantes, utilizando extensões ou ferramentas adicionais para análises mais aprofundadas de usabilidade.

Pereira and Vilela [2019] apresentaram o BDDManager, uma ferramenta voltada para o gerenciamento de testes comportamentais automatizados, baseada na metodologia Behavior Driven Development (BDD). O BDDManager permite que não-programadores gerenciem cenários de testes escritos em linguagem natural, simplificando a colaboração entre as equipes de desenvolvimento e negócios. A ferramenta automatiza a transformação de user stories em casos de teste executáveis, deixando a implementação dos steps necessários para os programadores. Em comparação, o ICTEST é projetado para atender a um público mais amplo, indo além do foco em equipes técnicas ou projetos baseados em BDD. Ele foi desenvolvido para ser acessível a qualquer pessoa responsável pelo gerenciamento de testes, independentemente de habilidades técnicas. Além disso, enquanto o BDDManager é restrito ao contexto específico de testes comportamentais, o ICTEST suporta uma gama mais ampla de metodologias e tipos de testes, como testes manuais, funcionais e automatizados, oferecendo maior flexibilidade para diferentes cenários de desenvolvimento.

Lönnfält *et al*. [2025] introduziram o ITMOS, um sistema de gerenciamento de testes inteligente projetado para otimizar a tomada de decisões no processo de teste de software. O sistema utiliza processamento de linguagem natural e aprendizado de máquina para sugerir se os casos de teste devem ser automatizados ou executados manualmente. O ITMOS foi avaliado em um projeto industrial na Ericsson, obtendo alta precisão na classificação da configuração CI e na tomada de decisões sobre a automatização de casos, reduzindo a necessidade de especialistas para essas escolhas. Comparado ao ICTEST, o ITMOS foca em decisões automatizadas para melhorar a eficiência na configuração de pipelines (CI). Enquanto o ITMOS depende de processos avançados de aprendizado de máquina e é implementado com foco na integração contínua, o ICTEST prioriza acessibilidade e simplicidade com uma plataforma web que permite que qualquer membro da equipe, mesmo sem conhecimento técnico, gerencie e organize os testes de um projeto. Por fim, enquanto o ITMOS é especializado em otimizar decisões técnicas para grandes corporações, como a Ericsson, o ICTEST foi projetado para atender a diversos contextos de projetos e equipes.

Raksawat and Charoenporn [2021] propuseram um sistema de gerenciamento de testes de software baseado no padrão internacional ISO 29119. Os autores desenvolveram uma ferramenta composta por módulos para gerenciamento de planos de teste, casos de teste e ambiente de teste. Além disso, de acordo com os padrões ISO 29119-2 e ISO 29119-3, sua solução permite criar documentos de teste, como planos organizacionais, estratégias de teste e especificações de casos de teste. O sistema também possui alertas automáticos para entradas de dados incorretas, bem como recursos para comparar os resultados esperados com os resultados reais, promovendo uma abordagem de qualidade padronizada. Em comparação, o ICTEST também visa uma gestão de testes, mas dá ênfase à flexibilidade e acessibilidade para diferentes contextos de projeto e usuários com variados níveis de experiência técnica. Enquanto a solução dos autores procura aderir estritamente ao padrão ISO 29119, o ICTEST opera em maior escala, permitindo a personalização de fluxos de trabalho e suporte a múltiplas metodologias de teste. Além disso, embora o ICTEST não contenha nativamente funcionalidades como validação de entradas de dados e documentação padronizada, esses recursos poderiam ser integrados para complementar suas funções existentes.

Kumpin and Jureczko [2010] apresentaram o AuTester, um framework de código aberto projetado para automatizar testes e gerenciar documentações de testes seguindo o padrão IEEE 829 e regulamentações específicas da Polônia. O sistema permite a criação de cenários de teste, execução automatizada de testes e gerenciamento eficiente de artefatos de teste. O AuTester utiliza a linguagem AutoIt para automação de tarefas em ambientes Windows, com suporte à geração de relatórios personalizados e integração com sistemas de integração contínua. Em comparação, o AuTester é altamente dependente de um ambiente específico e de uma linguagem de automação específica, resultando em requisitos elevados para conhecimento técnico. Por outro lado, o ICTEST oferece maior usabilidade e acessibilidade, possibilitando que usuários com diferentes níveis de aptidão técnica colaborem no gerenciamento de testes sem exigir conhecimento especializado. Uma diferença significativa é a abordagem à documentação: enquanto o AuTester segue fielmente o padrão IEEE 829, o ICTEST adota uma abordagem flexível, permitindo que as equipes personalizem os fluxos e modelos de documentação conforme suas necessidades.

6 Conclusão

Neste artigo, é apresentado o ICTEST, um sistema web desenvolvido com o objetivo de enfrentar os desafios de um mercado de software cada vez mais competitivo, especialmente no que diz respeito aos processos de gerenciamento de testes associados a projetos de desenvolvimento. Dado o cenário atual, em que a demanda por qualidade está em constante crescimento, o ICTEST surge como uma solução acessível, com funcionalidades colaborativas que centralizam e facilitam o gerenciamento de testes de forma eficaz em diferentes tipos de organização.

Como identificado na seção de artigos relacionados, é possível observar, através das comparações apresentadas, as restrições impostas por outros sistemas devido à aplicação

de metodologias específicas, à ausência de ferramentas colaborativas ou à presença de complexidades que dificultam sua operação e usabilidade. O ICTEST, por outro lado, destaca-se por sua simplicidade e pela experiência do usuário, tornando-se viável até mesmo para equipes que não possuem expertise técnica em administração e gerenciamento de testes. Ao oferecer um sistema com funcionalidades robustas, a plataforma agrega valor por meio de sua estabilidade e acessibilidade para os times.

Os resultados da avaliação prática do ICTEST revelaram que a ferramenta é altamente aceitável para os usuários, recebendo escores elevados em produtividade, eficiência, facilidade de uso e flexibilidade. Com esses achados, há evidências suficientes de que a plataforma tem um impacto significativo na gestão de testes, além de validar sua proposta de valor como uma ferramenta essencial para equipes de desenvolvimento modernas.

Como trabalhos futuros, destaca-se a possibilidade de expansão das funcionalidades do ICTEST por meio da integração com ferramentas externas populares no mercado, como Jira e Jenkins. Além disso, planeja-se implementar melhorias que facilitem o uso da plataforma e recursos que suportem metodologias de desenvolvimento de software, como Behavior Driven Development (BDD). Essas mudanças podem não apenas aumentar a relevância e a aplicabilidade da ferramenta em diferentes organizações, mas também abrir novas oportunidades para a usabilidade da aplicação.

Por fim, o ICTEST representa uma solução promissora para a otimização dos processos de gerenciamento de testes, com o potencial de contribuir para a elevação da qualidade de software e para a melhoria das práticas e da eficiência nos projetos de desenvolvimento.

Declarações complementares

Contribuições dos autores

Erick Silva Kokubum desenvolveu a solução e realizou os testes, Guilherme Souza Brandão e Vítor Hugo Barbosa dos Santos participaram da avaliação da solução. Frederico Araújo Durão contribuiu com a arquitetura do sistema, desenvolvimento e avaliação. Todos os autores foram responsaveis igualmente pela escrita e revisão de manuscrito científico.

Conflitos de interesse

Os autores declaram que não têm nenhum conflito de interesses

Disponibilidade de dados e materiais

Os conjuntos de dados (e/ou softwares) gerados e/ou analisados durante o estudo atual serão feitos mediante solicitação.

Repositório Github: https://github.com/ictestufba

Referências

Bensten, C., Mamnani, D., and Aymer, A. (2020). Continuous testing report. Disponível em: `https://www.capgemini.com/de-de/wp-content/uploads/sites/5/2020/03/Report_Continous_Testing_2020_Sogeti_Capgemini.pdf`.

Black, R., van Veenendaal, E., and Graham, D. (2007). *Foundations of Software Testing: ISTQB Certification*. Cengage Learning EMEA.

da Silva Lima, R. and Barreiros, A. (2014). *Introdução ao Teste de Software*. Novatec, São Paulo, Brazil.

Fastify (2021). Fastify: A fast and low overhead web framework for node.js. Disponível em: `https://www.fastify.io/`. Acesso em: 01 out. 2022.

Generosi, A., Villafan, J. Y., Giraldi, L., Ceccacci, S., and Mengoni, M. (2022). A Test Management System to Support Remote Usability Assessment of Web Applications. *Information*, 13(10). DOI: 10.3390/info13100505.

Gonçalves, J. and Silva, S. R. (2020). 4 testers: documentação e gerenciamento de testes de software. Sp.gov.br. Disponível em: `https://ric.cps.sp.gov.br/handle/123456789/5758`.

Hattori, R., Fantinato, M., Faria, M., and Chaim, M. (2013). SPLOT-TM: Apoio ao Gerenciamento de Testes Funcionais em Linha de Produto de Software. In *Anais do IX Simpósio Brasileiro de Sistemas de Informação*, pages 379–390, Porto Alegre, RS, Brasil. SBC. DOI: 10.5753/sbsi.2013.5705.

Institute of Electrical and Electronics Engineers (1990). Ieee standard glossary of software engineering terminology. *IEEE Std 610.12-1990*, pages 1–84. DOI: 10.1109/IEEESTD.1990.101064.

ISO/IEC/IEEE 29119 (2021). Iso/iec/ieee international standard - software and systems engineering - software testing – part 2: Test processes - redline. *ISO/IEC/IEEE 29119-2:2021(E) - Redline*, pages 1–129. Disponível em: `https://ieeexplore.ieee.org/document/9687474`.

Kumpin, A. and Jureczko, M. (2010). *AuTester a framework for automated testing and test management*. Disponível em: `https://www.researchgate.net/publication/234137861_AuTester_a_framework_for_automated_teesting_and_test_management`.

Liu, L. (1992). *A Supportive Environment for the Management of Software Testing*. PhD thesis, Durham University. Disponível em: `http://etheses.dur.ac.uk/5726/`.

Lönnfält, A., Tu, V., Gay, G., Singh, A., and Tahvili, S. (2025). An intelligent test management system for optimizing decision making during software testing. *Journal of Systems and Software*, 219:112202. DOI: 10.1016/j.jss.2024.112202.

Pereira, V. A. and Vilela, P. R. S. (2019). Bddmanager: Uma ferramenta de gerenciamento de testes comportamentais. *Revista dos Trabalhos de Iniciação Científica da UNICAMP*, 27:7557. DOI: 10.20396/revpibic2720192274.

Pezzè, M. and Young, M. (2008). *Software Testing and Analysis: Process, Principles, and Techniques*. John Wiley & Sons, Hoboken, NJ.

Pitschinetz, R. and Wegener, J. (1995). TESSY: Management of Software Tests. In *Experience with the Management of Software Projects*, Karlsruhe, Germany. Daimler-Benz AG. DOI: 10.1016/S1474-6670(17)43770-0.

PostgreSQL (2021). What is postgresql. Disponível em: `https://www.postgresql.org/docs/current/intro-whatis.html`. Acesso em: 01 out. 2022.

Prasad, M. N. V. N. K. and Gopal, M. (2008). *Software Testing and Quality Assurance: Theory and Practice*. John Wiley & Sons, Hoboken, NJ.

Prisma (2021). Prisma: Next-generation node.js and types-

cript orm. Disponível em: https://www.prisma.io/. Acesso em: 01 out. 2022.

Raksawat, C. and Charoenporn, P. (2021). Software testing system development based on iso 29119. *Journal of Advances in Information Technology*, 12(2):128–134. DOI: 10.12720/jait.12.2.128-134.

Reis, M. F. S., Ambrósio, A. M., and Ferreira, M. G. V. (2012). Vvteste: Ambiente de geração e gerenciamento de testes e de defeitos como apoio aos processos de verificação e validação do mps.br. In *Anais do Workshop Anual do MPS (WAMPS)*, pages 152–159, São José dos Campos, Brasil. Instituto Nacional de Pesquisas Espaciais (INPE). Disponível em: http://urlib.net/8JMKD3MGP7W/3BESFE5.

Vasconcelos, A. B., Souza, I. S., da Silva, I. F., and Alves, K. (2013). Test-module: Uma ferramenta para gerenciamento de testes de software integrada ao firescrum. In *Anais do Simpósio Brasileiro de Qualidade de Software (SBQS)*, pages 150–160, Recife, Brasil. SBC. Disponível em: http://www.facol.com/si/downloads/Revista_SI_2009/Artigo07.pdf.

Vercel and Next.js (2021). Next.js: The react framework for production. Disponível em: https://nextjs.org/. Acesso em: 01 out. 2022.

Walgude, A. and Natarajan, S. (2019). World quality report 2019-20. Disponível em: https://www.capgemini.com/es-es/wp-content/uploads/sites/16/2022/12/World-Quality-Report-2019-20.pdf.