Thaliane Ramos Gomes
Orientador: Francisco das Chagas Imperes Filho

Smoke test utilizando técnicas de testes funcionais em aplicações *web* desenvolvidas por discentes da UFPI - CSHNB

Picos - PI
03 de março de 2023

Thaliane Ramos Gomes
Orientador: Francisco das Chagas Imperes Filho

Smoke test utilizando técnicas de testes funcionais em aplicações *web* desenvolvidas por discentes da UFPI - CSHNB

Monografia submetida ao Curso de Bacharelado em Sistemas de Informação como requisito parcial para obtenção de grau de Bacharel em Sistemas de Informação.

Universidade Federal do Piauí
Campus Senador Helvídio Nunes de Barros
Bacharelado em Sistemas de Informação

Picos - PI
03 de março de 2023

FICHA CATALOGRÁFICA
Serviço de Processamento Técnico da Universidade Federal do Piauí
Biblioteca José Albano de Macêdo

G633s Gomes, Thaliane Ramos

Smoke test utilizando técnicas de testes funcionais em aplicações *web* desenvolvidas por discentes da UFPI – CSHNB [recurso eletrônico] / Thaliane Ramos Gomes – 2023.
38 f.

1 Arquivo em PDF
Indexado no catálogo *online* da biblioteca José Albano de Macêdo-CSHNB
Aberto a pesquisadores, com restrições da Biblioteca

Trabalho de Conclusão de Curso (Graduação) – Universidade Federal do Piauí, Bacharelado em Sistemas de Informação, Picos, 2023.
"Orientador: Me. Francisco das Chagas Imperes Filho"

1. Técnicas de testes. 2. Software. 3. Qualidade de desempenho. 4. Aplicação *web*. I. Imperes Filho, Francisco das Chagas. II. Título.

CDD 004.2

Emanuele Alves Araújo CRB 3/1290

SMOKE TEST UTILIZANDO TÉCNICAS DE TESTES FUNCIONAIS EM APLICAÇÕES WEB DESENVOLVIDAS POR DISCENTES DA UFPI - CSHNB

THALIANE RAMOS GOMES

Monografia aprovado como exigência parcial para obtenção do grau de Bacharel em Sistemas de Informação.

Data de Aprovação

Picos - PI, 20 de março de 2023

Prof. Francisco das Chagas Imperes Filho

Prof. Fredison Muniz de Sousa

Prof. Francisca Pâmela Carvalho Nunes

Agradecimentos

Portanto dEle, por Ele e para Ele são todas as coisas. Eu não poderia começar esses agradecimentos sem oferecer toda glória a Deus, pois foi por causa dEle que eu cheguei até aqui, e é por causa dEle que conheci todos que fizeram parte da minha jornada.

Em segundo lugar, agradeço eternamente aos meus pais, Ana Lúcia e Noan, que nunca mediram esforços para que eu pudesse estar aqui hoje, e sempre me apoiaram, me incentivaram, investiram, fizeram sacrifícios e acima de tudo, sempre oraram por mim. Eu reconheço que sou resultado das suas orações. Agradeço também à minha madrasta Juliana, pela preocupação e todo o cuidado comigo. Aos meus irmãos, Silas e Dayan, vocês são uma parte de mim, muito obrigada por acreditarem e estarem comigo. Eu amo vocês, muito obrigada por tanto!

Ao meu esposo, Marcos Vinício, você é parte de tudo isso, pois foi você que enxugou muitas das minhas lágrimas quando tudo estava difícil (no curso e fora dele), foi você que, com todo amor, cuidou de mim quando eu esquecia até mesmo de me alimentar para terminar algum trabalho da universidade. Obrigada por tanto esforço para cuidar do nosso lar, você é um exemplo para mim. Te amo!

Quero agradecer ao meu professor e orientador, Dr. Francisco Imperes, por acreditar, acolher e me orientar até aqui. Sua ajuda foi fundamental para o meu sucesso acadêmico. Aqui também deixo meus agradecimentos a todos os professores do curso que fizeram parte da minha jornada, obrigada por serem inspiração para tantos alunos.

Não posso deixar de agradecer aos meus professores do curso técnico em Informática, no IFPI: Jonathas Jivago, Felipe Gonçalves e Kleber Kroll. Nunca esquecerei do que fizeram para que eu conseguisse entrar no curso técnico integrado ao médio no IFPI. Ali vocês estavam me mostrando e principalmente acreditando que eu poderia ir longe na área de tecnologia. Obrigada por acreditarem em mim quando nem eu sabia que poderia chegar aqui!

Aos meu amigos que conheci e me ajudaram tanto durante essa longa jornada: Lucas Vinicius, Erick Macgregor, Carlos Vitor, Cícero, Elievelto, Gabriel Holanda, Hidelgardo, Jederilson, Rubenilson, Lucas Sousa e Samuel Lélis. Muito obrigada por serem parte disso.

Mas, em especial, eu agradeço aos mais incríveis dessa turma: Marcos paulo, Gabriell Oliveira, Lucas Bezerra e Samuel Oliveira, e quero falar um pouco sobre cada.

Marcos Paulo: Foi a pessoa que ficou uma semana sem me chamar pelo nome pois esquecia e tinha vergonha de falar comigo quando dividíamos apartamento, e hoje é um dos meus melhores amigos, e tenho muito orgulho da sua jornada e de tudo o que construiu até hoje, pois tive o privilégio de ser parte dela. Ele é o que nunca dizia um não quando o assunto era ajudar. Detalhe importante que não pode ser esquecido: ele tinha o sonho de lançar um perfume com cheiro de churrasco.

Gabriell Oliveira: É o que geralmente fica na dele, quieto, mas tem um coração gigante, e sempre tem uns conselhos um tanto quanto peculiares. Sua forma de nos fazer sorrir com suas piadas é bem única, homem de poucas palavras (kkkkkk). Ah, não posso esquecer dele dando sermão na gente (eu, Marcos Paulo e Samuel) por levar pessoas pro apartamento (quando dividimos no início do curso) e ficar até tarde jogando. Mas superamos isso e já perdoamos ele.
Lucas Bezerra: É aquele que só em pensar no que escrever já sinto vontade de sorrir. É o caçula da turma, que quando chegou no curso só falava de arduíno e do IFPI. Ele é o que se desespera muito rápido (acho que até mais do que eu) e acaba fazendo a gente sorrir do desespero dele. Jamais vou esquecer das sessões de desabafo comigo (kkkkk), e muito menos dele me chamando de azeda, na saída do RU, por não gostar muito de doces.
Samuel Oliveira: É um irmão pra mim, estudamos juntos no ensino médio, e viemos para a graduação juntos. Sempre foi minha dupla de trabalhos, e quando era para montar grupos, sempre estávamos no mesmo também. Após uns meses ficamos só nós dois dividindo apartamento, e ele viu meus desesperos; meus choros quando não conseguia fazer os trabalhos e provas; minhas crises existenciais quando eu me perguntava o que estava fazendo nesse curso (kkkkkk), enfim. Foram inúmeras noites em claro fazendo trabalhos, você me explicando conteúdos, sorrindo, se desesperando, tendo conversas aleatórias, ou as vezes só "virando bicho"mesmo. Obrigada pelas coisas bem simples e que pareciam inúteis, como por exemplo: apagar a luz do meu quarto quando eu dormia de tão cansada e esquecia a lâmpada acesa. Você é parte da minha história, e tenho orgulho de fazer parte da sua também! Por fim, só quero dizer que eu amo vocês quarto, muito obrigada por tudo, e contem sempre comigo para o que precisarem!

Deus troveja maravilhosamente com a sua voz; grandes coisas Ele faz, as quais não podemos compreender.

Jó 37:5

Resumo

Desde que os *softwares* passaram a fazer parte da nossa rotina, surgiu a necessidade de desenvolvê-los com qualidade, que atendam as necessidades dos usuários, e principalmente, que atendam aos requisitos que foram levantados desde o início de sua concepção e planejamento. Buscar melhorias para os *softwares* desenvolvidos é algo significativo, pois, quanto antes um erro for detectado, menos custo e trabalho ele trará para a empresa e equipe de desenvolvimento. Mesmo assim, na maioria dos processos de desenvolvimento de *softwares*, a qualidade passa a ser pensada e os testes começam a ser realizados apenas no final de todo o desenvolvimento. Pensar em testes apenas no final do processo faz com que essa fase do desenvolvimento seja negligenciada, realizada com pressa e sem analisar quais as melhores técnicas que poderiam ser aplicadas para verificar que todos os requisitos foram cumpridos. Desse modo, o objetivo desse trabalho foi realizar *smoke tests* nas funcionalidades de Cadastro de Usuário e *Login*, aplicando duas técnicas de testes funcionais em conjunto: Particionamento de Equivalência e Análise de Valor Limite, tendo como base os requisitos funcionais levantados durante a fase de desenvolvimento de dois *softwares* desenvolvidos por discentes da UFPI-CSHNB. Os resultados obtidos mostraram que o objetivo da etapa de testes foi bem-sucedido, pois demostraram a presença de erros nos *softwares*.

Palavras-chaves: qualidade, *software*, técnicas de testes, casos de teste, aplicação *web*.

Abstract

Since software became part of our routine, the need arose to develop it with quality, to meet the needs of users, and above all, to meet the requirements that were raised from the beginning of its conception and planning. Seeking improvements for the software developed is significant, because the sooner an error is detected, the less cost and work it will bring to the company and the development team. Even so, in most software development processes, quality starts to be thought of and tests begin to be carried out only at the end of all development. Thinking about tests only at the end of the process causes this phase of development to be neglected, carried out in a hurry and without analyzing the best techniques that could be applied to verify that all requirements were met. Thus, the objective of this work was to perform smoke tests on the User Registration and Login functionalities, applying two functional testing techniques together: Equivalence Partitioning and Limit Value Analysis, based on the functional requirements raised during the development phase. of two software developed by UFPI-CSHNB students. The results obtained showed that the objective of the testing stage was successful, as they demonstrated the presence of errors in the software.

Lista de ilustrações

Lista de tabelas

Lista de abreviaturas e siglas

BVA	*Boundary Value Analisys* (Análise de Valor Limite)
S1	Aplicação *web* 1: UBS *System* - *Software* para gerenciamento de procedimentos clínicos realizados na Unidade Básica de Saúde do CSHNB/UFPI - UBS/UFPI
S2	Aplicação *web* 2: GAMEWORK - Uma platafroma gamificada para resolução e análise estatística e atividades escolares
RF	Requisito Funcional

Sumário

1 Introdução

Os *softwares* passaram a fazer parte da rotina diária das pessoas e estão introduzidos em nossas vidas de maneira indispensável. Desde então, surgiu a necessidade de desenvolver *softwares* de qualidade, que atendam as necessidades dos usuários, e principalmente, que atendam aos requisitos que foram levantados desde o início do planejamento. A pedra fundamental que sustenta a engenharia de *software* é o foco na qualidade (PRESSMAN; MAXIM, 2016). E, de modo geral, a qualidade de um *software* está diretamente ligada a um produto que seja proveitoso/útil. Pressman e Maxim (2016) afirmam que os *softwares* devem oferecer um valor mensurável para quem utiliza e para quem desenvolve.

A garantia da qualidade no processo de desenvolvimento tem se tornado, cada vez mais, uma premissa básica para a competitividade das indústrias de *software* (VOLPI, 2001). É fundamental que o processo de qualidade seja completo, a fim de assegurar que erros, falhas e defeitos sejam identificados o quanto antes, ocasionando o menor custo e esforço possível. Produzir e manter um *software* tem um custo elevado, no entanto todo erro também tem um custo, seja ele para o usuário ou para a organização. Não há dúvida nenhuma de que a qualidade tem um preço, mas a falta de qualidade também tem um preço (PRESSMAN; MAXIM, 2016).

Visando diminuir o custo para manutenção de um *software* e encontrar erros nos estágios iniciais da sua concepção, existe uma etapa chamada de teste de *software*. Ela é uma fase dentro do processo da verificação e validação do desenvolvimento de um produto de *software*. O processo de verificação e validação (V&V) tem a intenção de mostrar que um *software* se adequa às suas especificações, ao tempo que satisfaz as especificações do cliente (SOMMERVILLE, 2011). Desse modo, a atividade de teste de *software* consiste em informar determinadas entradas, e examinar as saídas juntamente com seu comportamento para saber se ele está realizando o que se espera (PESSONI, 2011). Mas, apesar da importância, a atividade de teste costuma ser a mais negligenciada do desenvolvimento de *software* (SILVA, 2003).

Para que não haja negligência e a fase de testes seja executada com êxito, faz-se necessário o uso de técnicas para conduzir os testes. Essas técnicas de testes ajudam a pessoa que testa a aplicação (também conhecida como analista de testes ou *tester*), uma vez que o contato com o *software* se dá, na maioria dos processos de desenvolvimento, apenas na etapa de testes. O *International Software Testing Qualifications Board* fomenta que os testes são feitos de maneira diferente dependendo dos cenários (ISTQB, 2018). Assim sendo, cabe ao profissional que testa a aplicação discernir qual(ais) técnica(s) de teste deve aplicar no contexto em que ele está inserido.

Alguns trabalhos anteriores empregaram uma técnica de teste funcional (também conhecidas como caixa-preta) em aplicações *web* (KARISMA; WAHANANI; NURLAILI,

2021; SHALEH et al., 2021; AHRIZAL et al., 2020). A aplicação destas técnicas tem como base os requisitos funcionais levantados durante a fase de desenvolvimento do *software*. Desses trabalhos, nenhum empregou duas técnicas de testes funcionais em conjunto afim de garantir que mais cenários fossem mapeados e que houvesse uma maior cobertura dos requisitos funcionais.

Dessa forma, este trabalho propõe a utilização de duas técnicas de testes funcionais, complementares, em duas aplicações *web* para demostrar sua eficácia. Para isso, é necessário entender sobre cada uma das técnicas de testes funcionais, bem como a forma que elas se complementam, para que assim sejam aplicadas em conjunto e proporcione uma maior cobertura dos requisitos funcionais, bem como a correção de falhas ainda durante a fase de desenvolvimento.

1.1 Objetivos

O objetivo geral deste trabalho é demonstrar a importância e necessidade da aplicação de técnicas de testes funcionais nas aplicações *web* desenvolvidas por alunos do Curso de Sistemas de Informação do campus CSHNB/UFPI, a fim de melhorar a qualidade das aplicações, bem como demonstrar a necessidade da elaboração de casos de testes desde a fase de levantamento de requisitos até a entrega do produto aos usuários finais.

Os objetivos específicos deste trabalho são:

- Aplicar técnicas de testes funcionais nas principais funcionalidades das aplicações *web* desenvolvidas por discentes do curso de Sistemas de Informação do campus CSHNB/UFPI.
- Analisar os resultados obtidos e esperados com a aplicação das técnicas de testes funcionais utilizadas.
- Demonstrar a eficácia da combinação de técnicas de testes funcionais aplicadas em conjunto nas aplicações *web* analisadas.

1.2 Organização

O restante desta obra está estruturado da seguinte maneira: O Capítulo 2 é dedicado à apresentação de conceitos importantes para a compreensão plena do conteúdo deste trabalho. O Capítulo 3 apresenta uma revisão da literatura, mostrando trabalhos relacionados e explorando características específicas, organizadas em uma tabela claramente apresentada. O Capítulo 4 fornece uma visão geral sobre o processo de desenvolvimento deste projeto, incluindo a forma de elaboração dos casos de teste, os resultados dos testes realizados e uma discussão sobre os mesmos. Finalmente, o Capítulo 5 apresenta as

conclusões finais e direciona possíveis caminhos para trabalhos futuros. E, no Apêndice A estão dispostos os casos de testes elaborados para a funcionalidade de Cadastro de Usuário na aplicação *web* 1 (*Software* de Prontuário Eletrônico para Gerenciamento de Procedimentos Clínicos).

2 Referencial Teórico

Este capítulo descreve conceitos fundamentais que compõe a base deste projeto. Para isto, as seções incluem abordagens sobre teste de *software*; defeito, erro e falha; técnicas de testes funcionais, dentre elas, as duas que foram utilizadas nesse trabalho: Análise de valor limite e Particionamento de equivalência; *Smoke test* e por fim sobre casos de testes.

2.1 Teste de *software*

Teste de *software* é uma maneira de avaliar a qualidade do *software* e reduzir o risco de falha do *software* em operação (ISTQB, 2018). Visando diminuir o custo para manutenção de uma aplicação e encontrar erros nos estágios iniciais da sua concepção, existe uma etapa/atividade no desenvolvimento chamada de teste de *software*. Ela é uma fase/atividade dentro do processo da verificação e validação do desenvolvimento de um produto de *software*. O processo de verificação e validação (V&V) tem a intenção de mostrar que um *software* se adequa às suas especificações, ao tempo que satisfaz as especificações do cliente (SOMMERVILLE, 2011). Desse modo, a atividade de teste de *software* consiste em informar determinadas entradas, e examinar as saídas juntamente com seu comportamento para saber se ele está realizando o que se espera (PESSONI, 2011).

É crucial enfatizar que o objetivo do teste de *software* não é, como se costuma acreditar, provar que o produto é correto. Ao invés disso, o objetivo é fazer com que o produto apresente erros e encontrar seus pontos fracos, destacando assim a natureza destrutiva dos testes. Ao identificar os erros e os problemas, a equipe de desenvolvimento pode então se concentrar em corrigi-los. Adicionalmente, o teste de *software* também visa garantir que o sistema seja capaz de funcionar adequadamente quando utilizado pelos usuários. Dessa forma, a atividade de teste procura por situações em que o produto não atenda às expectativas de um cliente ou usuário em determinado aspecto.

Mas, apesar da importância, a atividade de teste costuma ser a mais negligenciada do desenvolvimento de *software* (SILVA, 2003). É comum a algumas instituições deixarem a atividade de teste somente para o final do processo de desenvolvimento (PESSONI, 2011). Deixar a fase de testes somente para o final do processo de desenvolvimento acarreta em testes mal planejados, pouco tempo para execução dos casos de testes, falta de tempo para a correção de erros, falhas e defeitos encontrados, assim, essa fase acaba sendo realizada de qualquer forma. O *software* que não funciona corretamente pode levar a muitos problemas, incluindo a perda de recursos financeiros, de tempo ou da reputação comercial, e até mesmo ferimentos ou morte (ISTQB, 2018).

2.2 Defeito, Erro e Falha

Embora mais conhecidos como *bugs*, os defeitos, erros e falhas têm significados diferentes dentro da engenharia de *software*. O defeito (*Fault*) pode ser entendido como algo que se comporta de uma forma distinta do que é esperado. Esse comportamento está relacionado com a inconformidade da especificação do *software* e ocorre nas linhas do código. O defeito é resultado de algum engano na criação do *software* e pode ocasionar um erro, isso porque muitas vezes um defeito está em uma linha de código que pode nunca ser executada.

O erro (*Error*) ocorre quando um defeito é executado. É um estado inconsistente ou inesperado que acontece quando o *software* é executado e se comporta de uma forma diferente do esperado. Nem sempre um erro é resultado de um defeito, e muitos erros só são encontrados quando um conjunto específico de entradas são realizadas para ativar o defeito. Um erro pode ou não deixar o *software* em um estado que impossibilite o usuário de utilizar ou prosseguir o fluxo de uso.

Quando um erro se propaga, gera um comportamento inesperado e pode impossibilitar o usuário de prosseguir na utilização do *software*, pode-se dizer que houve uma falha (*Failure*). Normalmente, alguns defeitos exigem entradas ou pré-condições muito específicas para desencadear uma falha, o que pode ocorrer raramente ou nunca (ISTQB, 2018). A Figura 1 representa como um defeito pode se tornar uma falha.

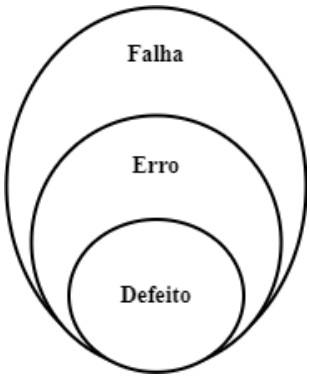

Figura 1 – Defeito, erro e falha

2.3 *Smoke Test*

O *Smoke test* (Teste de fumaça) é uma técnica de teste muito útil, pois permite que os testadores identifiquem rapidamente se as principais funcionalidades de uma aplicação estão funcionando corretamente, sem precisar gastar tempo com testes mais detalhados a princípio. Ou seja, é uma forma de avaliar se as funcionalidades básicas do *software* estão funcionando corretamente, pois segundo Bartlett (2022) os testes de fumaça são uma

maneira eficaz de localizar os principais defeitos no *software* antes de testar o restante do produto em um nível mais profundo.

Em seu livro "*Agile Testing: A Practical Guide for Testers and Agile Teams*", Crispin e Gregory (2009) reafirmam que o *smoke test* é importante para detectar problemas que possam impedir testes mais detalhados, como falhas na instalação ou configuração do *software*, erros de integração, problemas de conectividade ou instabilidade geral do sistema.

2.4 Técnicas de testes funcionais

Guarienti et al. (2014) afirma que é preciso a aplicação de técnicas de teste durante o processo de desenvolvimento de *software* para garantir que o sistema proporcione um serviço válido. Assim sendo, técnicas de teste de *software* podem ser entendidas como um apoio para a pessoa que testa aplicações. A partir da utilização de técnicas, o testador poderá ser mais exato na elaboração dos testes. O objetivo de uma técnica de teste é ajudar a identificar as condições de teste, os casos de teste e os dados de teste (ISTQB, 2018). Dentre diversas técnicas que existem, este trabalho irá focar nas técnicas de testes funcionais.

As técnicas de testes funcionais, também chamadas de Teste caixa-preta, se baseiam nas especificações do *software* levantadas durante a fase inicial de desenvolvimento e, permitem derivar séries de condições de entrada que utilizarão completamente todos os requisitos funcionais para um programa (PRESSMAN; MAXIM, 2016). Nessa técnica os detalhes de implementação não são considerados e o *software* é avaliado segundo o ponto de vista do usuário (PESSONI, 2011). As técnicas de testes funcionais mais conhecidas e utilizadas neste trabalho serão: Particionamento de equivalência e Análise de valor limite.

2.4.1 Particionamento de Equivalência

O particionamento é usado para criar partições de equivalência, geralmente chamadas de classes de equivalência, que são compostas de conjuntos de valores que são processados da mesma maneira (GILLER, 2018). Em outras palavras, o particionamento de equivalência divide o domínio de entrada de um programa em classes de dados a partir das quais podem ser criados casos de teste (PRESSMAN; MAXIM, 2016). Em vez de testarmos cada possibilidade individualmente, podemos testar apenas uma representação de cada classe, o que reduz o tempo e os recursos necessários para executar os testes.

Uma classe de equivalência representa um conjunto de estados válidos e inválidos para uma condição de entrada (NETO, 2007), que pode ser uma condição numérica, condição lógica, grupo de valores relacionados, e um intervalo de valores, assim podemos ter uma melhor compreensão de como o sistema se comporta com diferentes tipos de entradas.

Na Figura 2 temos um exemplo. Suponhamos que temos um requisito onde determina que o usuário pode listar no mínimo 3 e no máximo 8 itens por vez em um sistema. As classes de equivalências para essa regra podem ser 3:

Classe de equivalência 1: excluir 1 item (quantidade menor que o valor mínimo);

Classe de equivalência 2: excluir 10 itens (quantidade maior que o valor máximo);

Classe de equivalência 3: excluir 6 itens (quantidade válida, dentro do limite estabelecido).

Figura 2 – Exemplo do particionamento de equivalência

Dessa forma, podemos garantir que tanto os valores válidos quanto os inválidos que abrangem esse requisito foram contemplados e servirão como casos de teste. A vantagem de utilizar o particionamento de equivalência é que ele nos permite ser mais eficientes e efetivos na cobertura de testes.

2.4.2 Análise de Valor Limite

A análise de valor limite (*Boundary Value Analisys* - BVA) é uma extensão do particionamento de equivalência, mas só pode ser usada quando a partição é ordenada, consistindo em dados numéricos ou sequenciais (ISTQB, 2018). Em vez de selecionar qualquer elemento de uma classe de equivalência, a BVA conduz à seleção de casos de teste nas "bordas" da classe (PRESSMAN; MAXIM, 2016). Ou seja, a análise de valor limite testa o valor limite das classes de equivalência, os valores mínimos e máximos, bem como um valor abaixo e acima do limite, pois segundo NETO (2007) muitos erros tendem a ocorrer nos limites das classes de entrada e não no "centro".

Seguindo o exemplo do requisito onde determina que o usuário pode listar no mínimo 3 e no máximo 8 itens por vez em um sistema, na Figura 3 pode-se observar que teremos que testar 6 valores no total, para os 2 limites que temos:

Limite 1: excluir 2, 3 e 4 itens, onde 3 é o limite mínimo;

Limite 2: excluir 7, 8 e 9 itens, onde 8 é o limite máximo.

Assim, teremos seis casos de testes que abrangem os dois limites estabelecidos pelo requisito, e abrangem também um valor abaixo e acima de cada valor limite.

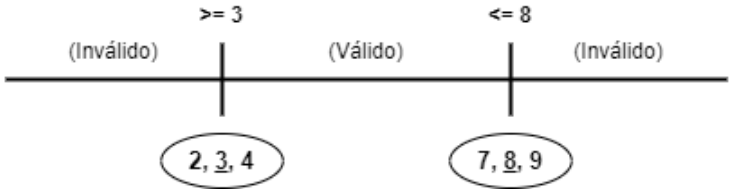

Figura 3 – Exemplo da análise do valor limite

2.5 Cenários e Casos de Teste

A atividade de teste é composta por alguns elementos essenciais que auxiliam na formalização desta atividade (NETO, 2007). Segundo Myers, Sandler e Badgett (2011), os cenários de teste são criados a partir da análise das funcionalidades do *software*, das especificações do sistema e das necessidades dos usuários, visando identificar e validar o comportamento do software em diferentes situações. O cenário de teste descreve uma situação mais abrangente e genérica. E de acordo com a ISO/IEC/IEEE-29119-1 (2022), os cenários são usados como uma base para gerar os casos de testes.

Já os casos de teste são artefatos que incluem um conjunto de entradas que precisam ser testadas, juntamente com as saídas esperadas e as saídas obtidas, descrevendo um único caso de teste que deve ser executado para verificar uma condição específica da funcionalidade. Dessa forma, a pessoa responsável por testar a aplicação pode comparar as saídas esperadas com as saídas reais e identificar erros e falhas. Para isso, é fundamental que os casos de teste sejam escritos de maneira clara e metódica, possibilitando encontrar a maior quantidade de problemas possíveis. Além disso, a estruturação dos casos de teste deve ser pensada cuidadosamente, para garantir sua repetibilidade e reutilização em caso de erros ou problemas.

Desse modo, o princípio fundamental dos casos de teste diz que esses devem ser escritos de maneira metódica e bem estruturada para que sejam repetíveis/reutilizáveis (PESSONI, 2011), para que, em caso de erros, seja possível utilizá-los novamente. A estruturação adequada de um caso de teste é fundamental para o sucesso da atividade de teste, pois permite que sejam identificados os erros e falhas do *software* de maneira eficiente. A escolha da estrutura ideal para o caso de teste dependerá de diversos fatores, tais como a natureza do *software*, a complexidade das funcionalidades a serem testadas e as expectativas em relação ao resultado final.

Além disso, é importante que a estruturação do caso de teste esteja alinhada ao contexto no qual o *software* se encontra, de modo a garantir a efetividade e a relevância dos testes realizados.

3 Trabalhos Relacionados

Este capítulo apresenta alguns trabalhos relacionados a utilização de técnicas de testes funcionais em aplicações *webs*. A Tabela 1 fornece um comparativo dos trabalhos relacionados considerando alguns critérios: qual técnica foi utilizada; em qual fase do desenvolvimento essas técnicas foram aplicadas; qual a plataforma do software; por fim, com quais tipos de usuários do sistema os testes foram realizados.

Tabela 1 – Comparação de trabalhos relacionados

Trabalho	**Técnica de teste**	**Fase do desenvolvimento**	**Plataforma**	**Usuário**
(NURUDIN et al., 2019)	Análise de valor limite	Após	*Web* - Local	Não expôs
(TRENGGINAZ et al., 2020)	Particionamento de equivalência	Após	*Web*	Não expôs
(AHRIZAL et al., 2020)	Análise de valor limite	Após	*Web* - Local	Não expôs
(SHALEH et al., 2021)	Particionamento de equivalência	Após	*Web*	Administrador
(KARISMA; WAHANANI; NURLAILI, 2021)	Particionamento de equivalência	Após	*Web*	Todos
Este Trabalho	Particionamento de equivalência e Análise de valor limite	Durante	*Web*	Todos

O primeiro critério de comparação é referente a **técnica utilizada** pelos trabalhos. Dentre eles, particionamento de equivalência foi a técnica mais utilizada, nos trabalhos de Trengginaz et al. (2020), Shaleh et al. (2021), Karisma, Wahanani e Nurlaili (2021). Isso se dá pelo fato dessa técnica ser um tanto mais abrangente quando refere-se à quantidade de casos de testes que podem ser elaborados. De modo que a análise de valor limite é um complemento do particionamento de equivalência, apenas Nurudin et al. (2019) e Ahrizal et al. (2020) utilizaram essa técnica para realizar os testes. No entanto, nenhum dos trabalhos utilizou essas duas técnicas de testes funcionais em conjunto.

Em relação a **fase de desenvolvimento** em que a elaboração e execução dos casos de testes foram realizados, todos os trabalhos realizaram os testes após os *softwares* estarem finalizados. Entretanto, Nurudin et al. (2019) e Ahrizal et al. (2020) realizaram os testes antes do *software* estar em uso pelos usuários finais. A elaboração dos casos de testes deste

trabalho será durante o desenvolvimento dos softwares, para que ao serem executados, os erros identificados sejam corrigidos o quanto antes.

O terceiro critério de comparação está relacionado à **plataforma** para qual o *software* foi disponibilizado. Assim como este, todos os trabalhos optaram por aplicar técnicas em *softwares* que utilizam a plataforma *web*. Porém, é válido ressaltar que os *softwares* utilizados por Nurudin et al. (2019) e Ahrizal et al. (2020) não estavam disponíveis na *internet*.

O último critério comparado foi em relação ao tipo de **usuário**. Para este critério, foi analisado os tipos de usuários que podem utilizar os *softwares* estudados. Nurudin et al. (2019), Trengginaz et al. (2020) e Ahrizal et al. (2020) não expuseram quais os tipos de usuários que podem ter acesso aos *softwares*. Este trabalho irá focar nos usuários administradores (caso tenha) e todos os outros tipos de usuários que podem utilizar as aplicações que serão examinadas. O único trabalho que realizou testes para mais de um usuário foi Karisma, Wahanani e Nurlaili (2021). Uma aplicação *web* que permite sua utilização para tipos de usuários com perfis de acesso distintos, deve ter seu uso testado e validado para todos os perfis de usuários que acessam o sistema.

Desta maneira, a proposta definida neste projeto é a única que aplicará as técnicas de particionamento de equivalência e análise de valor limite em conjunto, bem como garantir que as principais funcionalidades do *software* atenda aos requisitos para todos os tipos de usuários finais. Outro aspecto a destacar é que este trabalho vai elaborar e executar os casos de teste durante a fase de desenvolvimento das aplicações *web* desenvolvidas por alunos do Curso de Sistemas de Informação do campus CSHNB/UFPI.

4 Capítulo de Desenvolvimento

Neste capítulo é apresentado o processo para a realização dos testes nas principais funcionalidades de duas aplicações *web* desenvolvidas por discentes do curso de Sistemas de Informação, do campus CSHNB/UFPI.

4.1 Aplicações *Web* utilizadas

A seguir, será apresentado uma breve descrição dos sistemas escolhidos para estudo e aplicação das técnicas de testes funcionais propostas neste trabalho.

- **Aplicação *web* 1 (S1)**: UBS *System* - *Software* de prontuário eletrônico para gerenciamento de procedimentos clínicos com gerenciamento completo das consultas e dos componentes que envolvem a coleta de dados do paciente, requisição de exames, prescrição de medicamentos e gerenciamento de CID-10. Neste software há níveis de acesso à plataforma, ou seja, há três tipos de usuários(administrador, médico e técnico), bem como permissões diferentes para cada um.
- **Aplicação *web* 2 (S2)**: GAMEWORK - Plataforma gamificada para auxiliar professores no processo de correção automatizada, bem como análises estatísticas de atividades escolares direcionadas aos alunos do Ensino Médio. Nessa plataforma, os usuários cadastrados como professores podem cadastrar, visualizar, alterar e excluir atividades e/ou avaliações escolares para seus alunos, bem como o cadastro de questões e turmas. Já os usuários cadastrados como alunos, poderão ter acesso às atividades e/ou avaliações da turma em que foi cadastrado.

Após escolher as aplicações a serem utilizadas, foi realizado reuniões com os desenvolvedores de cada uma com o propósito de entender quais as funcionalidades imprescindíveis. Após essas reuniões, ficou claro que o Cadastro e o *Login* são as funcionalidades mais cruciais pois são porta de entrada do usuário no sistema. Devido a isso, é inadmissível que essas funcionalidades apresentem quaisquer problemas que possam causar transtornos aos usuários.

Nesse sentido, considerando a relevância do Cadastro e do *Login* para o funcionamento adequado das aplicações, essas funcionalidades foram definidas como os dois cenários a serem testados no presente trabalho. Dessa forma, será possível avaliar a capacidade das aplicações de gerenciarem adequadamente o processo de cadastro e autenticação dos usuários, bem como identificar e corrigir possíveis problemas que possam surgir nessas funcionalidades críticas.

4.2 Elaboração dos casos de testes

Após a definição dos cenários de teste para as funcionalidades das aplicações *web* em questão, foi necessário elaborar os casos de teste correspondentes a cada cenário. Nesse sentido, foram realizadas análises detalhadas dos dados requeridos para a realização do processo de cadastro e *login* em cada aplicação, a fim de compreender as regras de negócio associadas a cada tipo de dado.

As Figuras 4 e 5 mostram quais dados são necessários para a realização do cadastro dos usuários do tipo Médico e Técnico no S1. É importante salientar que, não foram elaborados casos de teste para cadastro de usuário do tipo Administrador, pois só é possível cadastrar esse tipo de usuário por meio do painel de administrador do sistema, que fica sob responsabilidade do desenvolvedor da aplicação.

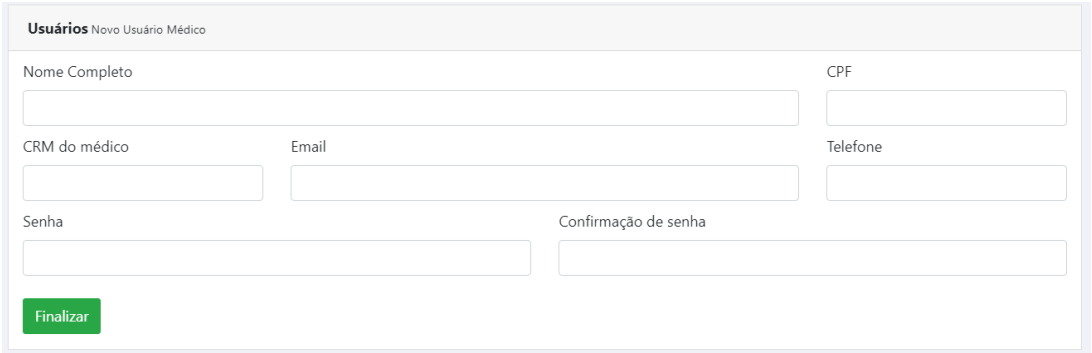

Figura 4 – Dados solicitados para cadastro de usuário Médico no S1

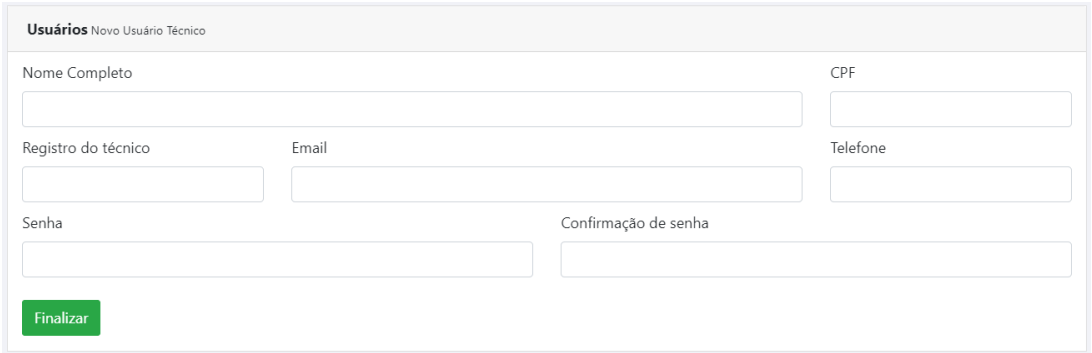

Figura 5 – Dados solicitados para cadastro de usuário Técnico no S1

Subsequente, a Figura 6 exibe os dados solicitados para realizar login, independente do tipo de usuário, na aplicação S1.

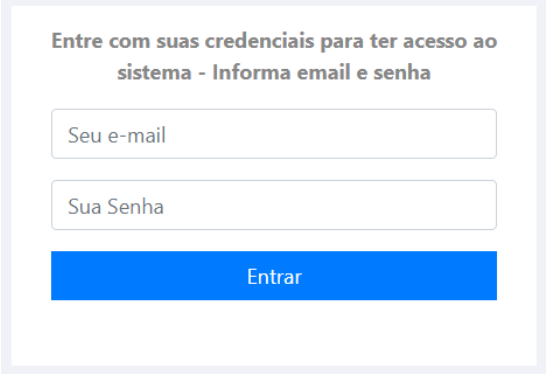

Figura 6 – Dados solicitados para efetuar login no S1

Já a Figura 7 mostra quais dados são necessários para a realização do cadastro dos usuários do tipo Professor e Aluno no S2. Vale evidenciar que os dados solicitados para cadastro de Professor e Aluno são os mesmos.

Subsequente, a Figura 8 exibe os dados solicitados para realizar login, independente do tipo de usuário, na aplicação S2.

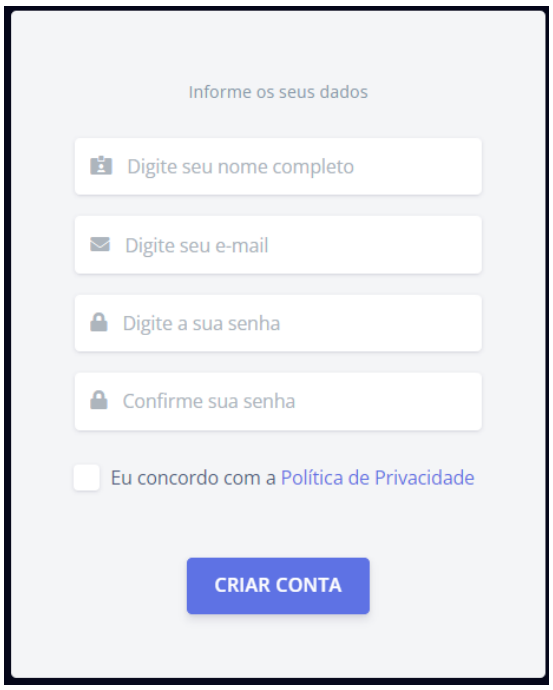

Figura 7 – Dados solicitados para cadastro de usuário no S2

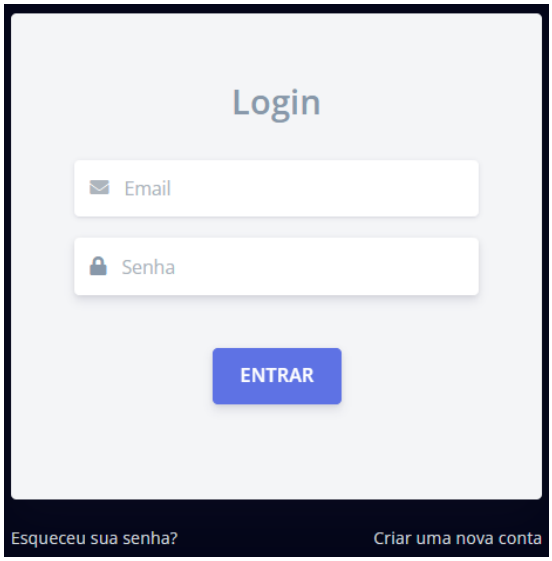

Figura 8 – Dados solicitados para efetuar login no S2

Na Tabela 2 é apresentado a quantidade de casos de testes elaborados para cada funcionalidade das duas aplicações *web* escolhidas para esse trabalho.

Tabela 2 – Quantidade de casos de testes elaborados

Funcionalidade	**Aplicação *web***	
	S1	**S2**
Cadastro	32	16
Login	3	3
Total	**35**	**19**

A estrutura em formato de tabela foi a escolhida para os casos de testes nesse trabalho. Nesses artefatos contém informações básicas e específicas que são necessárias para validação do requisito funcional. A Tabela 3 representa um exemplo de caso de teste para a funcionalidade de efetuar *Login* em um sistema qualquer. Este modelo foi adotado nesse trabalho pois contém as informações essenciais para análise dos resultados, podendo ainda ser adicionada mais uma coluna que informa se o caso de teste obteve êxito ou não.

Tabela 3 – Exemplo de casos de teste para um cenário de login

ID	**Descrição**	**Condição prévia**	**Entrada**	**Resultado esperado**	**Resultado obtido**	**RF**
001	*Login* do usuário X	Usuário cadastrado no sistema	*login*: user01 senha: teste12	Logado com sucesso	Logado com sucesso	RF02
002	*Login* do usuário X com senha inválida	Usuário cadastrado no sistema	*login*: user01 senha: teste	Aviso de senha inválida	Aviso de usuário inválido	RF02

Utilizou-se a ferramenta Google Sheets[1] para criar as tabelas com os dados necessários para elaboração dos casos de teste. O Google Sheets é um aplicativo de planilha eletrônica baseada em nuvem disponibilizada pela empresa Google. Essa ferramenta foi escolhida devido à sua praticidade e facilidade de compartilhamento, além de ser uma opção gratuita e acessível pela *internet*.

4.3 Execução dos casos de testes

Nesta seção, serão apresentados os casos de teste elaborados e os resultados obtidos a partir da execução de cada um deles. Os testes foram realizados no navegador *web* Chrome, na versão 110.0.5481.104 (Versão oficial) 64 bits, que foi escolhido devido à sua ampla utilização e compatibilidade com as aplicações *web* em questão.

4.3.1 S1

Os casos de teste para a funcionalidade de Cadastro de usuário, estão disponíveis no Apêndice A devido a quantidade de casos criados. Para elaboração, foi levado em consideração o seguinte requisito do *software*:

RF01: Cadastro de Usuário - Usuários podem ser cadastrados como médicos, técnicos ou administradores.

A Tabela 7 mostra os casos de teste para os campos de Nome completo, CPF, CRM e Email. A Tabela 8 mostra os casos de teste para os campos de Telefone e Senha, para cadastrar um usuário do tipo Médico. Subsequente, a Tabela 9 mostra os casos de teste dos campos Nome completo, CPF, Registro e *E-mail* para cadastrar um usuário do tipo Técnico. E a Tabela 10 mostra os casos de teste para os campos de Telefone e Senha, para cadastrar um usuário do tipo Técnico no S1.

Os casos de teste para a funcionalidade de *Login* estão exibidos na Tabela 4, bem como o resultado obtido com a execução de cada caso de teste. Para elaboração, foi levado em consideração o seguinte requisito do *software*:

RF02: Login - O sistema permitirá o acesso à plataforma aos usuários, autenticados com *e-mail* e senha.

Tabela 4 – Casos de testes para Login: S1

ID	Descrição	Cond. prévia	Entrada	Resultado Esperado	Resultado Obtido	Êxito
01	Realizar login com email e senha cadastrados	Usuário cadastrado no sistema	Email: medica_thaliane@gmail.com Senha: (inserir a senha deste usuário)	Acesso a plataforma	Acesso a plataforma!	S
02	Realizar login com email inválido	Usuário cadastrado no sistema	email: thaliane Senha: (informar a senha deste usuário)	Exibir um aviso que o email é inválido/incompleto	Aviso informando que o email está incompleto	S
03	Realizar login com senha inválida/incorreta	Usuário cadastrado no sistema	Email: medica_thaliane@gmail.com Senha: teste123	Exibir um aviso informando que a senha está incorreta	Aviso informando que e-mail e senha precisam estar corretos	S

[1] https://www.google.com/intl/pt-BR/sheets/about/#features.

4.3.2 S2

A Tabela 5 apresenta todos os casos de testes referentes à funcionalidade de Cadastro de Professor e Aluno do S2, bem como o resultado obtido com a execução de cada um. Para elaboração, foi levado em consideração o seguinte requisito do *software*:

RF01: Cadastro de Usuário - O usuário pode se cadastrar como professor ou estudante. Os dados a serem coletados são: nome completo, e-mail e senha.

Tabela 5 – Casos de testes para Cadastro - S2

ID	Descrição	Cond. prévia	Entrada	Resultado Esperado	Resultado Obtido	Êxito
01	Cadastrar usuário como Professor	-	Nome: Professor Silva email: profsilva@gmail.com senha: professor@01	Usuário ser cadastrado no sistema	Cadastro efetuado!	S
02	Cadastrar usuário como Aluno	-	Nome: Aluno silva email: alunosilva@gmail.com senha: aluno@01	Usuário ser cadastrado no sistema	Cadastro efetuado!	S
03	Cadastrar Professor com apenas 1 nome	-	nome: Professor email: prof02@gmail.com senha: professor@02	Usuário não deve ser cadastrado. Informar que o Nome deve ser completo.	Usuário cadastrado!	N
04	Cadastrar Professor com um E-mail inválido	-	nome: Professor Silva email: professor.com.br senha: professor@03	Exibir um aviso que o campo E-mail deve ser preenchido com email válido	Aviso que o campo E-mail deve ser preenchido com um email válido	S
05	Cadastrar um Professor com o campo Senha inválido	-	nome, Professor Silva email: prof02@gmail.com senha: 12	Exibir um aviso com o motivo da senha ser inválida	Aviso informando que a senha é inválida	S
06	Cadastrar Professor com senha de 5 dígitos	-	nome: Professor Silva email: prof03@gmail.com senha: silva	Informar que a senha é muito curta	Aviso informando que a senha é muito curta!	S
07	Cadastrar Professor com senha de 7 dígitos	-	nome: Professor Silva email: prof04@gmail.com senha: silva@p	Informar que a senha é muito curta	Aviso informando que a senha é muito curta!	S
08	Cadastrar Professor com senha de 8 dígitos	-	nome: Professor Silva email: prof05@gmail.com senha: silva@pr	Cadastro realizado.	Cadastro realizado!	S
09	Cadastrar Professor com senha apenas numérica	-	nome: Professor Silva email: prof06@gmail.com senha: 1011121314	Informar que a senha é inteiramente numérica	Aviso informando que a senha é apenas numérica!	S
010	Cadastrar Aluno com apenas 1 nome	-	nome: Aluno email: aluno02@gmail.com senha: aluno@02	Usuário não deve ser cadastrado. Informar que o Nome deve ser completo.	Usuário cadastrado!	N
011	Cadastrar Aluno com um E-mail inválido	-	nome: Aluno Silva email: aluno.com.br senha: aluno@03	Exibir um aviso que o campo E-mail deve ser preenchido com um email válido	Aviso que o campo E-mail deve ser preenchido com um email válido	S
012	Cadastrar Aluno com o campo Senha inválido	-	nome, Aluno Silva email: aluno02@gmail.com senha: 12	Exibir um aviso com o motivo da senha ser inválida	Aviso informando que a senha é inválida	S
013	Cadastrar Aluno com senha com 5 dígitos	-	nome: Aluno Silva email: aluno03@gmail.com senha: silva	Exibir um aviso com o motivo da senha ser inválida	Aviso informando que a senha é muito curta!	S
014	Cadastrar um Aluno com senha com 7 dígitos	-	nome: Aluno Silva email: aluno04@gmail.com senha: silva@a	Exibir um aviso com o motivo da senha ser inválida	Aviso informando que a senha é muito curta!	S
015	Cadastrar um Aluno com senha com 8 dígitos	-	nome: Aluno Silva email: aluno05@gmail.com senha: silva@al	Usuário cadastrado.	Cadastro realizado!	S
016	Cadastrar um Aluno com senha apenas numérica	-	nome: Aluno Silva email: aluno06@gmail.com senha: 1011121314	Exibir um aviso com o motivo da senha ser inválida	Aviso informando que a senha é apenas numérica!	S

Os casos de teste para a funcionalidade de *Login* estão exibidos na Tabela 6, bem como o resultado obtido com a execução de cada caso de teste. Para elaboração, foi levado em consideração o seguinte requisito do *software*:

RF02: Login - O sistema permitirá o acesso à plataforma aos usuários devidamente autenticados com *E-mail* e senha.

Tabela 6 – Casos de testes para Login - S2

ID	Descrição	Cond. prévia	Entrada	Resultado Esperado	Resultado Obtido	Êxito
01	Realizar login com email e senha cadastrados	Usuário cadastrado no sistema	Email: thaliane.ramos123@gmail.com Senha: (inserir a senha deste usuário)	Acesso a plataforma	Acesso a plataforma!	S
02	Realizar login com email inválido	Usuário cadastrado no sistema	email: thaliane.ramos Senha: (informar a senha deste usuário)	Exibir um aviso que o email é inválido/incompleto	Aviso informando que o email está incompleto	S
03	Realizar login com senha inválida	Usuário cadastrado no sistema	Email: thaliane.ramos123@gmail.com Senha: 123	Exibir um aviso informando que a senha está incorreta	Aviso informando que email e senha precisam estar corretos	S

4.4 Resultados e Discussão

Esta seção tem como objetivo sintetizar os resultados das análises feitas neste trabalho.

A Figura 9 apresenta os resultados obtidos durante os testes realizados no S1, com foco na funcionalidade de Cadastro de Usuário e *Login*. A partir dos resultados apresentados na Figura 9, pode-se observar que a atividade de testes foi bem-sucedida em evidenciar os erros presentes no S1. Dos casos de testes realizados, 45,7% apresentaram defeitos, indicando a presença de problemas que poderiam comprometer a experiência do usuário, como por exemplo: cadastrar um médico com CRM inválido e isso gerar problemas futuros para o médico e paciente.

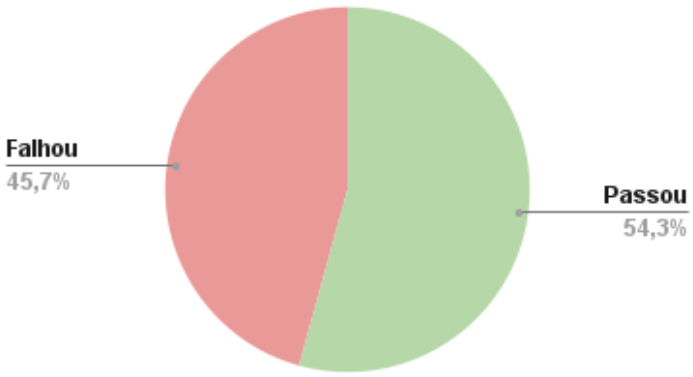

Figura 9 – Porcentagem dos casos de teste - S1

A Figura 10 apresenta os resultados obtidos durante os testes realizados no S2, com foco na funcionalidade de Cadastro de Usuário e *Login*. A partir dos resultados apresentados na Figura 10, pode-se observar que a atividade de testes também foi bem-sucedida em evidenciar erros presentes no S2. Dos casos de testes realizados, 10,5% apresentaram defeitos, indicando a presença de problemas que poderiam comprometer a experiência do usuário.

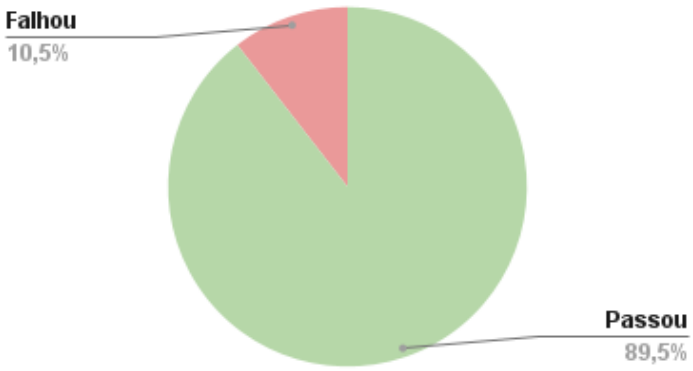

Figura 10 – Porcentagem dos casos de teste - S2

É importante ressaltar que a redução do número de erros encontrados na aplicação S2 em comparação com a S1 reforça a importância da adoção de boas práticas de desenvolvimento de *software*, pois essa redução do número de erros pode ter ocorrido pelo fato das definições dos requisitos e regras de negócios do S2 terem sido mais detalhadas e melhor planejadas. Assim, essa prática de planejar e elaborar bem os requisitos pode trazer benefícios significativos para a equipe de desenvolvimento e para o produto final.

A comunicação dos erros encontrados aos desenvolvedores é um passo crucial para a qualidade do *software*. Por isso, todos os erros encontrados foram repassados para os desenvolvedores de cada aplicação, para que sejam corrigidos antes do *software* ser disponibilizado para os clientes. Dessa forma, é possível garantir que as aplicações tenham uma qualidade adequada, credibilidade no mercado e que os usuários possam utilizá-lo de forma confiável e segura.

5 Conclusão e Direções futuras

Este trabalho teve como objetivo principal demonstrar a importância e necessidade da aplicação do Particionamento de equivalência e Análise de valor limite (técnicas de testes funcionais) nas funcionalidades de Cadastro de usuário e *Login* nas aplicações *web* desenvolvidas por alunos do Curso de Sistemas de Informação do campus CSHNB/UFPI. Para chegar ao objetivo, foram elaborados 35 casos de teste para o S1 e 19 para o S2, buscando abranger o maior número de possibilidades utilizando as técnicas escolhidas para este trabalho. Esses casos de testes abrangeram todos os campos de dados solicitados do usuário em cada plataforma.

Os casos de testes elaborados permitiram evidenciar que, apesar das duas funcionalidades serem semelhantes para ambas aplicações *web*, apresentaram resultados distintos na execução dos testes. Enquanto no S1, 45,7% dos casos de testes falharam, no S2, apenas 10,5% dos casos de testes detectaram erro. Isso indica que o processo de desenvolvimento da aplicação 2 foi mais criterioso e que as regras de negócio foram melhor definidas no início do projeto.

Apesar da eficácia das técnicas de teste utilizadas, é importante destacar que a utilização apenas dessas duas técnicas em conjunto não garante inteiramente a detecção de todos os erros de uma aplicação. Por isso, como trabalhos futuros, é necessário ampliar a utilização de outras técnicas de teste tanto Funcionais como Não Funcionais e expandir a prática para todo o restante dos sistemas. Além disso, a implementação de testes automatizados pode ser considerada para aprimorar o processo de testes, reduzindo o tempo e o custo associados à realização de testes manuais repetitivos e diminuindo o risco de erros humanos e imprecisões nos resultados. Com essas melhorias, a confiabilidade e qualidade do software como um todo serão aprimoradas.

Referências

AHRIZAL, D. et al. Pengujian perangkat lunak sistem informasi peminjaman playstation dengan teknik boundary value analysis menggunakan metode black box testing. *Jurnal Informatika Universitas Pamulang*, v. 5, n. 1, p. 73–77, 2020. Citado 3 vezes nas páginas 14, 21 e 22.

BARTLETT, J. *What is Smoke Testing?* 2022. Disponível em: <https://blog-.testlodge.com/what-is-smoke-testing/>. Acesso em: 02 nov. 2022. Citado na página 17.

CRISPIN, L.; GREGORY, J. *Agile testing: A practical guide for testers and agile teams*. [S.l.]: Pearson Education, 2009. Citado na página 18.

GILLER, J. *Using Equivalence Partitioning and Boundary Value Analysis in Black Box Testing*. 2018. Disponível em: <https://www.stickyminds.com/article/using-equivalence-partitioning-and-boundary-value-analysis-black-box-testing>. Acesso em: 02 out. 2021. Citado na página 18.

GUARIENTI, P. et al. Abordagem de análise do tempo de resposta para teste de desempenho em aplicações web. Pontifícia Universidade Católica do Rio Grande do Sul, 2014. Citado na página 18.

ISO/IEC/IEEE-29119-1. *Software and systems engineering — Software testing — Part 1: General concepts*. 2022. International Organization for Standardization. Disponível em: <https://www.iso.org/standard/77845.html>. Acesso em: 02 fev. 2023. Citado na página 20.

ISTQB. Certified tester foundation level syllabus. 2018. Acesso em: 10 set. 2021. Citado 5 vezes nas páginas 13, 16, 17, 18 e 19.

KARISMA, E. D.; WAHANANI, H. E.; NURLAILI, A. L. Pengujian menggunakan black box berbasis equivalence partitioning pada layanan aspirasi dan pengaduan online dinas kominfo jombang. *Jurnal Informatika dan Sistem Informasi (JIFoSI)*, v. 2, n. 2, p. 275–281, 2021. Citado 3 vezes nas páginas 14, 21 e 22.

MYERS, G. J.; SANDLER, C.; BADGETT, T. *The art of software testing*. [S.l.]: John Wiley & Sons, 2011. Citado na página 20.

NETO, A. C. D. Introdução a teste de software. *Revista Engenharia de Software. ano*, v. 1, n. 1, 2007. Citado 3 vezes nas páginas 18, 19 e 20.

NURUDIN, M. et al. Pengujian black box pada aplikasi penjualan berbasis web menggunakan teknik boundary value analysis. *Jurnal Informatika Universitas Pamulang*, v. 4, n. 4, p. 143–148, 2019. Citado 2 vezes nas páginas 21 e 22.

PESSONI, V. V. Implantação de processos de teste de software um estudo de caso voltado ao cercomp-ufg. 2011. Acesso em: 18 set. 2021. Citado 4 vezes nas páginas 13, 16, 18 e 20.

PRESSMAN, R.; MAXIM, B. *Engenharia de Software-8ª Edição*. [S.l.]: McGraw Hill Brasil, 2016. Citado 3 vezes nas páginas 13, 18 e 19.

SHALEH, I. A. et al. Pengujian black box pada sistem informasi penjualan buku berbasis web dengan teknik equivalent partitions. *Jurnal Teknologi Sistem Informasi dan Aplikasi*, v. 4, n. 1, p. 38–45, 2021. Citado 2 vezes nas páginas 14 e 21.

SILVA, E. J. da. Preetool: Uma ferramenta para apoiar o teste baseado em predicados. 2003. Acesso em: 18 set. 2021. Citado 2 vezes nas páginas 13 e 16.

SOMMERVILLE, I. *Engenharia de software*. 9. ed. São Paulo: Pearson Education–BR, 2011. Citado 2 vezes nas páginas 13 e 16.

TRENGGINAZ, R. B. et al. Pengujian aplikasi pemesanan tiket kereta berbasis website menggunakan metode black box dengan teknik equivalence partitioning. *Jurnal Teknologi Sistem Informasi dan Aplikasi*, v. 3, n. 3, p. 144–149, 2020. Citado 2 vezes nas páginas 21 e 22.

VOLPI, L. M. Uma estratégia de teste de software para ambiente cliente - servidor. 2001. Acesso em: 17 set. 2021. Citado na página 13.

Apêndices

APÊNDICE A – Casos de teste do Cadastro de usuário - S1

Tabela 7 – Casos de testes para Nome completo, CPF, CRM e E-mail - Usuário Médico - S1

ID	Descrição	Cond. prévia	Entrada	Resultado Esperado	Resultado Obtido	Êxito?
01	Cadastrar usuário como Médico	-	Nome completo: Medico Silva CPF: 861.197.150-70 CRM: 123456 email: medico@gmail.com telefone: (89) 996500100 senha: medico@01	Usuário cadastrado com sucesso	Usuário cadastrado com sucesso!	S
02	Cadastrar usuário como Técnico	-	Nome completo: Técnico Silva CPF: 651.346.840-06 Registro: 123456 email: tecnico@gmail.com telefone: (89) 965530001 senha: tecnico@01	Usuário cadastrado com sucesso	Usuário cadastrado com sucesso!	S
03	Cadastrar Médico com apenas 1 nome no campo Nome completo	-	Nome completo: medico CPF: 861.197.150-70 CRM: 123654 email: medico1@gmail.com telefone: (89) 996500100 senha: medico@01	Usuário não deve ser cadastrado! Informar que o Nome deve ser completo.	Usuário cadastrado.	N
04	Cadastrar Médico com 5 dígitos no CPF	-	Nome completo: Medico Silva CPF: 861.19 CRM: 123654 email: medico2@gmail.com telefone: (89) 996500100 senha: medico@02	Usuário não deve ser cadastrado. Informar que o CPF é inválido	Usuário cadastrado.	N
05	Cadastrar Médico com 10 dígitos no CPF	-	Nome completo: Medico Silva CPF: 861.197.150-7 CRM: 123654 email: medico3@gmail.com telefone: (89) 996500100 senha: medico@03	Usuário não deve ser cadastrado. Informar que o CPF é inválido	Usuário cadastrado	N
06	Cadastrar Médico com 12 dígitos no CPF	-	Nome completo: Medico Silva CPF: 861.197.150-702 CRM: 123654 email: medico4@gmail.com telefone: (89) 996500100 senha: medico@04	Não deixar inserir mais de 11 dígitos. Não cadastrar usuário.	O campo permite inserir no máximo 11 dígitos! Usuário não cadastrado	S
07	Cadastrar Médico com CRM de 1 dígito	-	Nome completo: Medico Silva CPF: 861.197.150-70 CRM: 1 email: medico4@gmail.com telefone: (89) 996500100 senha: medico@04	Usuário não deve ser cadastrado. Informar que o CRM é inválido	Usuário cadastrado	N
08	Cadastrar Médico com CRM de 5 dígitos	-	Nome completo: Medico Silva CPF: 861.197.150-70 CRM: 12345 email: medico5@gmail.com telefone: (89) 996500100 senha: medico@05	Usuário não deve ser cadastrado. Informar que o CRM é inválido	Usuário cadastrado	N
09	Cadastrar Médico com CRM de 7 dígitos	-	Nome completo: Medico Silva CPF: 861.197.150-70 CRM: 1234567 email: medico6@gmail.com telefone: (89) 996500100 senha: medico@06	Usuário não deve ser cadastrado. Informar que o CRM é inválido	Usuário cadastrado!	N
010	Cadastrar Médico com email inválido	-	Nome completo: Medico Silva CPF: 861.197.150-70 CRM: 123456 email: medico7.gmail telefone: (89) 996500100 senha: medico@07	Exibir um aviso que o campo E-mail deve ser preenchido com um email válido	Aviso informando que o campo E-mail deve ser preenchido com um email válido!	S

Tabela 8 – Casos de teste para Telefone e Senha - Usuário Médico - S1

ID	Descrição	Cond. prévia	Entrada	Resultado Esperado	Resultado Obtido	Êxito?
011	Cadastrar Médico com telefone de 4 dígitos	-	Nome completo: Medico Silva CPF: 861.197.150-70 CRM: 123456 email: medico7@gmail.com telefone: (89) 12 senha: medico@07	Usuário não deve ser cadastrado. Informar que o Telefone é inválido	Usuário cadastrado!	N
012	Cadastrar Médico com telefone de 10 dígitos	-	Nome completo: Medico Silva CPF: 861.197.150-70 CRM: 123456 email: medico8@gmail.com telefone: (89) 12345678 senha: medico@08	Usuário não deve ser cadastrado. Informar que o Telefone é inválido	Usuário cadastrado!	N
013	Cadastrar Médico com telefone de 12 dígitos	-	Nome completo: Medico Silva CPF: 861.197.150-70 CRM: 123456 email: medico9@gmail.com telefone: (89) 1234567890 senha: medico@09	Não deixar inserir mais de 11 dígitos	O campo permite inserir no máximo 11 dígitos! Usuário não cadastrado.	S
014	Cadastrar Médico com o campo Senha inválido	-	Nome completo: Medico Silva CPF: 861.197.150-70 CRM: 123456 email: medico9@gmail.com telefone: (89) 123456789 senha: @09	Usuário não deve ser cadastrado! Exibir um aviso com o motivo da senha ser inválida	Aviso informando que a senha é inválida	S
015	Cadastrar Médico com senha de 5 dígitos	-	Nome completo: Medico Silva CPF: 861.197.150-70 CRM: 123456 email: medico9@gmail.com telefone: (89) 1234567890 senha: medic	Não cadastrar usuário. Informar que a senha é muito curta	Aviso informando que a senha é muito curta! Usuário não cadastrado	S
016	Cadastrar Médico com senha de 7 dígitos	-	Nome completo: Medico Silva CPF: 861.197.150-70 CRM: 123456 email: medico9@gmail.com telefone: (89) 1234567890 senha: medico@	Informar que a senha é muito curta	Aviso informando que a senha é muito curta!	S
017	Cadastrar Médico com senha apenas numérica com mais de 8 dígitos	-	Nome completo: Medico Silva CPF: 861.197.150-70 CRM: 123456 email: medico9@gmail.com telefone: (89) 1234567890 senha: 1011121314	Informar que a senha é inteiramente numérica	Aviso informando que a senha é apenas numérica!	S

Tabela 9 – Casos de teste para Nome completo, CPF, Registro e E-mail - Usuário Técnico - S1

ID	Descrição	Cond. prévia	Entrada	Resultado Esperado	Resultado Obtido	Êxito?
018	Cadastrar Técnico com apenas 1 nome	-	Nome completo: Técnico CPF: 861.197.150-70 Registro: 123654 email: tecnico1@gmail.com telefone: (89) 996500100 senha: tecnico@01	Usuário não deve ser cadastrado. Informar que o Nome deve ser completo.	Usuário cadastrado.	N
019	Cadastrar Técnico com 5 dígitos no CPF	-	Nome completo: Técnico Silva CPF: 861.19 Registro: 123654 email: tecnico2@gmail.com telefone: (89) 996500100 senha: tecnico@02	Usuário não deve ser cadastrado. Informar que o CPF é inválido	Usuário cadastrado.	N
020	Cadastrar Técnico com 10 dígitos no CPF	-	Nome completo: Técnico Silva CPF: 861.197.150-7 Registro: 123654 email: tecnico3@gmail.com telefone: (89) 996500100 senha: tecnico@03	Usuário não deve ser cadastrado. Informar que o CPF é inválido	Usuário cadastrado	N
021	Cadastrar Técnico com 12 dígitos no CPF	-	Nome completo: Técnico Silva CPF: 861.197.150-702 Registro: 123456 email: tecnico4@gmail.com telefone: (89) 996500100 senha: tecnico@04	Não deixar inserir mais de 11 dígitos	O campo permite inserir no máximo 11 dígitos! Usuário não cadastrado	S
022	Cadastrar Técnico com Registro de 1 caractere	-	Nome completo: Técnico Silva CPF: 861.197.150-70 Registro: 1 email: tecnico4@gmail.com telefone: (89) 996500100 senha: tecnico@04	Usuário não deve ser cadastrado. Informar que o Registro é inválido	Usuário cadastrado	N
023	Cadastrar Técnico com Registro de 5 dígitos	-	Nome completo: Técnico Silva CPF: 861.197.150-70 Registro: 12345 email: tecnico5@gmail.com telefone: (89) 996500100 senha: tecnico@05	Usuário não deve ser cadastrado. Informar que o Registro é inválido	Usuário cadastrado	N
024	Cadastrar Técnico com Registro de 7 dígitos	-	Nome completo: Técnico Silva CPF: 861.197.150-70 Registro: 1234567 email: tecnico6@gmail.com telefone: (89) 996500100 senha: tecnico@06	Usuário não deve ser cadastrado. Informar que o Registro é inválido	Usuário cadastrado!	N
025	Cadastrar Técnico com email inválido	-	Nome completo: Técnico Silva CPF: 861.197.150-70 Registro: 123456 email: tecnico7.gmail telefone: (89) 996500100 senha: tecnico@07	Exibir um aviso que o campo E-mail deve ser preenchido com um email válido	Aviso informando que o campo E-mail deve ser preenchido com um email válido!	S

Tabela 10 – Casos de teste para Telefone e Senha - Usuário Técnico - S1

ID	Descrição	Cond. prévia	Entrada	Resultado Esperado	Resultado Obtido	Êxito?
026	Cadastrar Técnico com telefone de 4 dígitos	-	Nome completo: Técnico Silva CPF: 861.197.150-70 Registro: 123456 email: tecnico7@gmail.com telefone: (89) 12 senha: tecnico@07	Usuário não deve ser cadastrado. Informar que o Telefone é inválido	Usuário cadastrado!	N
027	Cadastrar Técnico com telefone de 10 dígitos	-	Nome completo: Técnico Silva CPF: 861.197.150-70 Registro: 123456 email: tecnico8@gmail.com telefone: (89) 12345678 senha: tecnico@08	Usuário não deve ser cadastrado. Informar que o Telefone é inválido	Usuário cadastrado!	N
028	Cadastrar Técnico com telefone de 12 dígitos	-	Nome completo: Técnico Silva CPF: 861.197.150-70 Registro: 123456 email: tecnico9@gmail.com telefone: (89) 1234567890 senha: tecnico@09	Não deixar inserir mais de 11 dígitos	O campo permite inserir no máximo 11 dígitos! Usuário não cadastrado.	S
029	Cadastrar Técnico com o campo Senha inválido	-	Nome completo: Técnico Silva CPF: 861.197.150-70 Registro: 123456 email: tecnico9@gmail.com telefone: (89) 123456789 senha: @09	Exibir um aviso com o motivo da senha ser inválida	Aviso informando que a senha é inválida	S
030	Cadastrar Técnico com senha de 5 dígitos	-	Nome completo: Técnico Silva CPF: 861.197.150-70 CRM: 123456 email: tecnico9@gmail.com telefone: (89) 1234567890 senha: tecni	Informar que a senha é muito curta	Aviso informando que a senha é muito curta! Usuário não cadastrado	S
031	Cadastrar Técnico com senha de 7 dígitos	-	Nome completo: Técnico Silva CPF: 861.197.150-70 Registro: 123456 email: tecnico9@gmail.com telefone: (89) 1234567890 senha: tecnic@	Informar que a senha é muito curta	Aviso informando que a senha é muito curta!	S
032	Cadastrar Técnico com senha apenas numérica com mais de 8 dígitos	-	Nome completo: Técnico Silva CPF: 861.197.150-70 Registro: 123456 email: tecnico9@gmail.com telefone: (89) 1234567890 senha: 1011121314	Informar que a senha é inteiramente numérica	Aviso informando que a senha é apenas numérica!	S

TERMO DE AUTORIZAÇÃO PARA PUBLICAÇÃO DIGITAL NA BIBLIOTECA "JOSÉ ALBANO DE MACEDO"

Identificação do Tipo de Documento

( ) Tese

( ) Dissertação

(x) Monografia

( ) Artigo

Eu, **Thaliane Ramos Gomes**, autorizo com base na Lei Federal nº 9.610 de 19 de Fevereiro de 1998 e na Lei nº 10.973 de 02 de dezembro de 2004, a biblioteca da Universidade Federal do Piauí a divulgar, gratuitamente, sem ressarcimento de direitos autorais, o texto integral da publicação "***Smoke test* utilizando técnicas de testes funcionais em aplicações *web* desenvolvidas por discentes da UFPI - CSHNB**", de minha autoria, em formato PDF, para fins de leitura e/ou impressão, pela internet a título de divulgação da produção científica gerada pela Universidade.

Picos-PI, 27 de Março de 2023.

gov.br Documento assinado digitalmente
THALIANE RAMOS GOMES DE MOURA
Data: 24/03/2023 22:38:04-0300
Verifique em https://validar.iti.gov.br

_____

Assinatura

_____

Assinatura