**PLANO DE TESTE: *PlayPiece***

Gustavo Horikochi

Gustavo Leme

Henrick Adrian

Ian Saggioratto

Leonardo Noboru

Centro Universitário Senac

**Histórico de revisões**

|  |  |  |  |
| --- | --- | --- | --- |
| Versão | Data | Autor | Descrição |
|  |  |  |  |

**PLANO DE TESTE**

O objetivo deste documento é descrever o Plano de Teste do Projeto de Software PlayPiece. Ele contém a definição do escopo do Projeto de Software, a especificação dos requisitos do sistema, a estratégia de teste e o esforço necessário para sua realização. Na estratégia de teste estão definidos as técnicas, níveis e tipos de teste que serão executados na iteração e os objetivos que devem ser atingidos. O documento apresenta ainda a definição do ambiente de testes, a matriz de responsabilidade e a relação dos casos de testes. Em seguida são listados os riscos de teste, os critérios para conclusão dos testes e o cronograma do projeto de teste.

Sumário

[1](#_heading=h.gjdgxs) ESCOPO DO PROJETO DE sOFTWARE 4

[*1.1*](#_heading=h.1fob9te) *Objetivo do Projeto* 4

[*1.2*](#_heading=h.3znysh7) *Metodologia de desenvolvimento* 4

[2](#_heading=h.2et92p0) Matriz de rastreabilidade de requisitos de sistema 5

[*2.1*](#_heading=h.tyjcwt) *Requisitos Funcionais* 5

[*2.2. Requisitos Não Funcionais* 6](#_heading=h.3dy6vkm)

[3](#_heading=h.1t3h5sf) ESTRATÉGIA DE TESTE 7

[4](#_heading=h.2s8eyo1) AMBIENTE de TESTE 8

[5](#_heading=h.17dp8vu) MATRIZ DE RESPONSABILIDADE 9

[6](#_heading=h.3rdcrjn) RELAÇÃO DOS CASOS DE TESTES 10

[7](#_heading=h.26in1rg) RISCOS DO PROJETO DE TESTE DE SOFTWARE 11

[8](#_heading=h.lnxbz9) CRITÉRIOS DE CONCLUSÃO 12

[9](#_heading=h.35nkun2) CRONOGRAMA DO PROJETO DE TESTE DE SOFTWARE 13

[REFERÊNCIAS 14](#_heading=h.1ksv4uv)

# ESCOPO DO PRODUTO DE SOFTWARE

O escopo deste produto de software é desenvolver um site e-commerce especializado na venda de produtos relacionados a jogos de mesa. O site proporcionará aos usuários uma experiência de compra simplificada e envolvente, oferecendo uma ampla gama de produtos, como cartas, tabuleiros, RPG, acessórios e outros itens relacionados a jogos de mesa.

## *Objetivo do Projeto*

Desenvolver um site e-commerce para produtos relacionados a jogos de mesa, como cartas, tabuleiros, RPG, entre outros.*]*

## *Metodologia de desenvolvimento*

Metodologia Ágil adotada para permitir maior flexibilidade, colaboração e entrega contínua de valor aos clientes. A abordagem ágil valoriza a interação frequente com os clientes, a capacidade de responder a mudanças rapidamente e a entrega de incrementos funcionais do produto em intervalos regulares. Uma das práticas comuns dentro da metodologia ágil é o uso de sprints.

Sprints são iterações de trabalho com prazos definidos, durante os quais o time de desenvolvimento se concentra em criar um incremento de produto funcional. Cada sprint normalmente tem uma duração fixa, tipicamente de duas a quatro semanas, durante as quais o time se compromete a entregar determinadas funcionalidades ou melhorias específicas.

No contexto do projeto de desenvolvimento de um site e-commerce para produtos relacionados a jogos de mesa, a implementação de sprints traria vários benefícios, incluindo:

Entregas Incrementais: O projeto avança em etapas mensuráveis, permitindo a entrega de funcionalidades específicas a cada sprint.

Feedback Contínuo: Os sprints facilitam a obtenção de feedback dos stakeholders e dos usuários finais em intervalos regulares, o que permite ajustes e melhorias contínuas.

Flexibilidade para Mudanças: A abordagem de sprints permite que as equipes respondam a mudanças nos requisitos de forma oportuna, integrando ajustes necessários nos ciclos de desenvolvimento subsequentes.

Maior Envolvimento das Partes Interessadas: As entregas frequentes e visíveis durante os sprints promovem maior envolvimento e transparência com os clientes e outros stakeholders.

Melhoria Contínua: A cada sprint, as equipes podem refletir sobre o trabalho concluído e identificar oportunidades de melhorias nos processos de desenvolvimento e na qualidade do produto.

# Matriz de rastreabilidade de requisitos de sistema

A rastreabilidade de requisitos de sistema descreve a capacidade de documentar e acompanhar a vida útil dos requisitos ao longo de todo o ciclo de vida do desenvolvimento de software, desde a concepção até a entrega final do produto. Esta prática é essencial para garantir a consistência, a completude e a conformidade entre os requisitos do sistema e as funcionalidades implementadas no software. A importância da rastreabilidade dos requisitos do sistema para o planejamento de testes de software pode ser descrita em vários aspectos fundamentais:

Verificação da Cobertura de Testes;

Rastreamento de Mudanças e Impactos;

Garantia da Qualidade do Software;

Identificação de Defeitos e Problemas Precoces;

Avaliação da Conclusão do Projeto;

Em resumo, a rastreabilidade dos requisitos de sistema é uma prática fundamental para garantir a qualidade e a conformidade do software com os requisitos do cliente. Ela ajuda a facilitar o planejamento, a execução e o gerenciamento eficaz de testes de software, garantindo que o produto final atenda às necessidades e expectativas dos usuários.

## *Requisitos Funcionais*

Descreve as características do produto, serviço ou resultado para cuja criação o projeto será realizado. Essas características terão normalmente menos detalhes nas fases iniciais e mais detalhes nas fases posteriores, conforme as características do produto forem progressivamente elaboradas. Embora a forma e o conteúdo das características variem, as descrições dos requisitos funcionais devem sempre fornecer detalhes suficientes para dar suporte ao planejamento do projeto de software e teste.

|  |  |  |
| --- | --- | --- |
| **REQUISITOS FUNCIONAIS (RF)** | | |
| **ID** | **Nome do RF** | **Descrição** |
| RF01 | Cadastro de Usuário e Login | Permitir que os usuários se cadastrem no site com informações básicas. |
| RF02 | Navegação e Pesquisa | Facilitar a navegação pelos produtos através de categorias como jogos de tabuleiro, cartas colecionáveis, jogos de RPG, entre outros.  Implementar um sistema de pesquisa eficiente para encontrar produtos específicos. |
| RF03 | Visualização de Produtos | Exibir imagens de alta qualidade e informações detalhadas de cada produto.  Permitir que os usuários visualizem as opções de compra, preço, disponibilidade e avaliações dos produtos. |
| RF04 | Carrinho de Compras | Permitir que os usuários adicionem produtos ao carrinho de compras.  Possibilitar a visualização do carrinho de compras, incluindo o subtotal e o total de compra. |
| RF05 | Processo de Compra | Oferecer um processo de compra seguro e intuitivo.  Integrar opções de pagamento confiáveis, como cartões de crédito, PayPal, entre outros. |
| RF06 | Gerenciamento de Conta do Usuário | Permitir que os usuários visualizem seu histórico de compras e acompanhem o status dos pedidos.  Oferecer a opção de editar informações pessoais e preferências de conta. |
| RF07 | Avaliações | Permitir que os usuários deixem avaliações dos produtos adquiridos.  Exibir avaliações para outros usuários. |
| RF08 | Gerenciamento de Estoque | Manter atualizações em tempo real sobre a disponibilidade de produtos.  Notificar os usuários sobre a disponibilidade de itens fora de estoque e os prazos estimados de reabastecimento. |

## *2.2. Requisitos Não Funcionais*

Segundo Pressman (2016), requisitos não-funcionais são os requisitos relacionados ao uso da aplicação em termos de desempenho, usabilidade, confiabilidade, segurança, disponibilidade, manutenção e tecnologias envolvidas. Estes requisitos dizem respeito a como as funcionalidades serão entregues ao usuário do software.

|  |  |  |  |
| --- | --- | --- | --- |
| **REQUISITOS NÃO-FUNCIONAIS** | | | |
| **RNF** | **Característica** | **Subcaracterística** | **Descrição** |
| 01 | Desempenho | Sistema | O sistema deve ser responsivo e capaz de lidar com uma carga significativa de tráfego sem queda de desempenho. |
| Carregamento das páginas | O tempo de carregamento das páginas deve ser rápido para garantir uma experiência de usuário eficiente. |
| 02 | Usabilidade | Interface de usuário | Garantir uma interface de usuário intuitiva e fácil de usar para facilitar a navegação e a compra de produtos. |
| Experiência | Oferecer uma experiência consistente em diferentes dispositivos e tamanhos de tela. |
| 03 | Segurança | Protocolos de segurança | Implementar protocolos de segurança robustos para proteger informações confidenciais dos clientes, como dados pessoais e informações de pagamento. |
| Normas de segurança | Garantir a conformidade com as normas de segurança estabelecidas para transações online e armazenamento de dados do cliente. |
| 04 | Portabilidade | Compatibilidade | Garantir que o site seja compatível com uma ampla variedade de navegadores da web, sistemas operacionais e dispositivos.  Adaptar o site para oferecer uma experiência consistente em diferentes plataformas. |
| 05 | Confiabilidade |  | Garantir a disponibilidade contínua do site, minimizando o tempo de inatividade não programado.  Implementar backups regulares e planos de contingência para lidar com possíveis falhas no sistema. |
| 06 | Manutenibilidade |  | Estruturar o código-fonte de forma clara e modular para facilitar a manutenção e as futuras atualizações.  Documentar adequadamente o código e os processos para permitir que outros desenvolvedores compreendam e modifiquem o sistema conforme necessário. |
| 07 | Escalabilidade |  | Projetar a arquitetura do sistema de forma a permitir o fácil escalonamento para lidar com o crescimento futuro do tráfego e do volume de transações.  Garantir que o sistema possa lidar com um aumento repentino no número de usuários sem comprometer o desempenho. |
| 08 | Legal e Regulamentar |  | Assegurar que o site e-commerce esteja em conformidade com todas as leis e regulamentações relacionadas a transações online, proteção de dados e direitos do consumidor.  Garantir a transparência nas políticas de privacidade e termos de uso do site. |

|  |  |
| --- | --- |
|  |  |

# ESTRATÉGIA DE TESTE

A estratégia de teste define quais técnicas, níveis e tipos de testes serão utilizados no projeto.

<Descreva aqui de forma detalhada quais técnicas, níveis e tipos de testes serão utilizados no projeto, relacionando com os requisitos de sistema.>

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ESTRATÉGIA DE TESTE** | | | | |
| **NÍVEIS DE TESTE** | | | | |
| **ID** | **Nome do Teste** | **Técnicas** | **Descrição** | **Requisitos de Sistemas** |
| 01 | Teste Unitário |  |  | RF01, RF02... |
| 02 | Teste de Integração |  |  |  |
| 03 | Teste de Sistema |  |  |  |
| 04 | Teste de Aceitação |  |  |  |
| **TIPOS DE TESTES** | | | | |
| **ID** | **Tipo do Teste** | **Técnicas**  **Descrição** | | **Requisitos de Sistemas** |
| 01 | Desempenho | Realizar análises de desempenho utilizando ferramentas como JProfiler e VisualVM para identificar e resolver possíveis problemas de desempenho | |  |
| 02 | Usabilidade | Realizar testes de usabilidade com usuários reais para avaliar a experiência do usuário e identificar possíveis melhorias na interface do usuário.  Implementar um design responsivo e realizar testes em diferentes dispositivos para garantir uma experiência consistente. | |  |
| 03 | Portabilidade | Testar o aplicativo em diferentes ambientes, como sistemas operacionais, para garantir que ele seja facilmente implantado e executado em diferentes configurações. | |  |
| 04 | Acessibilidade | Realizar testes de acessibilidade para garantir que o aplicativo atenda aos padrões de acessibilidade, como o WCAG.  Implementar recursos de acessibilidade, como suporte para leitores de tela, para tornar o aplicativo acessível para todos os usuários. | |  |
| 05 | Segurança | Utilizar ferramentas de verificação de segurança, como OWASP Dependency-Check e SonarQube, para identificar e corrigir vulnerabilidades no código-fonte. Utilizar proteções contra SQL-Injection. | |  |
| 06 | Análise Estática | Configurar e executar ferramentas de análise estática, como Checkstyle, PMD e FindBugs, para identificar possíveis problemas no código-fonte.  Corrigir e refatorar o código conforme as diretrizes recomendadas pelas ferramentas de análise estática. | |  |

# AMBIENTE de TESTE

O ambiente de testes mostra as condições em que serão executados os testes.

<Descreva aqui a importância da definição do ambiente de teste no planejamento de teste com base nos autores da área de engenharia e teste de software.>

| **AMBIENTE DE TESTE** | |
| --- | --- |
| **Recurso** | **Descrição** |
| **Computador cliente** |  |
| **Mobile** |  |
| **Servidor de aplicação** |  |
| **Servidor de banco de dados** |  |
| **Criação de relatórios** |  |
| **Browser cliente** |  |
| **SO cliente** |  |
| **<< Outro recurso>>** | <Informe a descrição completa. Apague esta linha da tabela se não existir outro recurso.> |

# MATRIZ DE RESPONSABILIDADE

<Descreva aqui a importância da definição da matriz de responsabilidade no planejamento de teste com base nos autores da área de engenharia e teste de software.>

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **MATRIZ DE RESPONSABILIDADE** | | | | |
| **Fase do Projeto** | **Atribuição Técnica** | **Responsabilidade** | **Nome** | **Descrição da Atividade** |
| Planejamento | Líder de Projeto de Teste | Técnico responsável pela liderança de um projeto de teste específico, geralmente relacionado a um sistema de desenvolvimento, seja um projeto novo ou uma manutenção. |  |  |
| Arquiteto de teste |  |  |  |
| Especificação | Analista de Teste |  |  |  |
| Execução | Testador |  |  |  |
| Resultados |  |  |  |  |

# RELAÇÃO e descrição DOS CASOS DE TESTES

Em [engenharia de software](https://pt.wikipedia.org/wiki/Engenharia_de_software), o caso de teste é um conjunto de condições usadas para [teste de software](https://pt.wikipedia.org/wiki/Teste_de_software). Neste plano de teste consta a relação de todos os cenários e seus respectivos casos de testes a serem especificados pelos analistas de teste.

|  |  |  |  |
| --- | --- | --- | --- |
| **CASOS DE TESTES** | | | |
| **ID** | **Cenário de Teste** | **Caso de Teste** | **Descrição** |
| 01 | CT\_LOGIN | CT01\_Novo Usuário | Caso de teste para validar o cadastramento de um novo usuário. |
| CT02\_Usuário Cadastrado | Caso de teste para validar o acesso de um novo usuário já cadastrado. |
| CT03\_Esqueci a senha | Caso de teste para validar a reinicialização de senha para o usuário. |
| 02 |  |  |  |
|  |  |
|  |  |
| 03 |  |  |  |
|  |  |
|  |  |
| 04 |  |  |  |
|  |  |
|  |  |
| 05 |  |  |  |
|  |  |
|  |  |

## *Descrição dos principais cenários de testes*

<Utilizando as técnicas de caixa preta descreva os principais cenários de testes do projeto (mínimo 3).>

# RISCOS DO PROJETO DE TESTE DE SOFTWARE

Este capítulo contém um sumário dos riscos chaves do projeto de teste, que se acontecerem, poderão afetar os resultados esperados e o projeto de software.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **RISCOS DO PROJETO DE TESTE** | | | | |
| **ID** | **Risco** | **Descrição** | **Grau do Risco** | **Resposta** |
| 01 | Hardware | Risco do equipamento (hardware) utilizado para execução da homologação do teste falhar. | Baixo | Mitigar – equipamento de reserva |
| 03 |  |  |  |  |
| 04 |  |  |  |  |
| 05 |  |  |  |  |

# CRITÉRIOS DE CONCLUSÃO

Os critérios de conclusão/continuidade dos testes determinam qual a tolerância a falhas, ou seja, partir deste momento se os testes serão suspensos. Detalhando quais critérios serão utilizados para suspensão de um teste e retomada do desenvolvimento, ou mesmo continuidade apesar dos cenários de falha, ou seja, se após executado um determinado número de cenários de testes obtiver-se um número significativo de falhas que os testes são suspensos e a entrega deverá passar por uma revisão completa, ou mesmo se determinada falha ocorrer os testes não prosseguirão até que a mesma esteja corrigida. Aqui também se define as condições as quais o teste prosseguirá ou não se os dados de teste não estiverem em conformidade com a massa especificada anteriormente.

<Defina quais os critérios para aceitação, parada e retomada dos testes.>

| **Critério** | **Descrição** |
| --- | --- |
| **Cobertura** | * Todos os requisitos definidos no escopo deste projeto deverão ser testados. * Testar fluxos básico, alternativo e de exceção de cada requisito. * Todas as regras de negócio serão testadas. |
| **Quando parar de testar** | * Quando nenhuma falha for encontrada após executar todos os casos de teste pelo menos uma vez. * Quando for encontrado um número excessivo de falhas logo no início dos testes. * Quando for encontrada alguma falha que inviabilize a continuidade dos testes. * Quando atingir no mínimo os seguintes critérios:   + Casos de Teste     - 100% dos casos de teste executados   + Defeitos     - 100% dos defeitos de severidade 1 corrigidos e verificados     - 90% dos defeitos de severidade 2 corrigidos e verificados     - 75% dos defeitos de severidade 3 corrigidos e verificados |
| **Em que momento reportar as falhas** | * Reportar todas as falhas imediatamente após encontrá-las. |
| **Teste de regressão** | * O teste de regressão será feito sobre a falha encontrada, sendo um fluxo básico e um fluxo alternativo do caso de uso. |

# CRONOGRAMA DO PROJETO DE TESTE DE SOFTWARE

<Descreva aqui a importância do cronograma do projeto no planejamento de teste com base nos autores da área de engenharia e teste de software, e insira o cronograma das atividades a serem executadas no projeto de teste.>

# REFERÊNCIAS

PRESSMAN, Roger S; MAXIM, Bruce R. **Engenharia de Software:** Uma Abordagem Profissional. 8ª ed. Editora McGraw-Hill,2016.

<Todos os autores citados no texto devem ser incluídos na lista de referências, que seguirão a formatação da NBR 6023:2002. A referência é composta pelos seguintes elementos: autor, título, edição, local, editora e data de publicação. Devem ser alinhadas à margem esquerda do texto com espaço simples e separadas umas das outras por um espaço simples. Usar o mesmo tipo e tamanho de fonte do texto do artigo, *Arial*, tamanho 12. Não deve ser numerada.>

EXEMPLOS E SUGESTÕES:

DELAMARO, Márcio Eduardo; MALDONADO, José Carlos; JINO, Mario. **Introdução ao Teste de Software.** São Paulo: Elsevier, 2007.

MOLINARI, Leonardo. **Inovação e Automação de Testes de Software**. Rio de Janeiro: Érica, 2010.

RIOS, Emerson. **Análise de riscos**: em projetos de teste de software. Rio de Janeiro: Alta Books, 2005.

RIOS, Emerson; MOREIRA FILHO, Trayahú. **Teste de Software.** Rio de Janeiro:Alta Books, 2006.

SOMMERVILLE, Ian. **Engenharia de software**. 9. ed. São Paulo: Pearson, 2011.