# Plano de Testes - Sistema de Gerenciamento de Inventário

## 1. Introdução

O Plano de Testes descreve a estratégia e os procedimentos para testar o Sistema de Gerenciamento de Inventário. O objetivo é garantir que o sistema atenda aos requisitos funcionais e não funcionais e que seja de alta qualidade e confiabilidade.

## 2. Objetivo

O objetivo deste plano de testes é definir as atividades de teste necessárias para validar e verificar o funcionamento do Sistema de Gerenciamento de Inventário, garantindo que todas as funcionalidades atendam às expectativas dos usuários.

## 3. Escopo

Este plano de testes abrange os testes de unidade, testes de integração e testes de sistema do Sistema de Gerenciamento de Inventário. Ele inclui testes funcionais para validar as principais funcionalidades do sistema.

## 4. Estratégia de Teste

* Os testes serão realizados em ambientes de desenvolvimento e teste.
* Serão utilizadas técnicas de teste automatizado para garantir a eficiência e a abrangência dos testes.
* Os casos de teste serão elaborados com base nos requisitos funcionais e não funcionais.
* Serão priorizados os testes de funcionalidades críticas e de alto impacto no sistema.
* Será utilizado um processo de revisão e aprovação para garantir a qualidade dos testes e a conformidade com os requisitos.

## 5. Casos de Teste

Os casos de teste serão elaborados para cobrir as seguintes áreas do sistema:

### 5.1 Testes Unitários

* Testes para todas as classes de serviço e repositório.
* Verificação de casos normais, de borda e de exceção.
* Cobertura de código para garantir a qualidade e a robustez do sistema.

### 5.2 Testes de Integração

* Testes para garantir a integração correta entre os componentes do sistema.
* Verificação da comunicação entre controladores, serviços e acesso ao banco de dados.
* Validação de cenários de uso com dados reais.

### 5.3 Testes de Sistema

* Testes para validar as principais funcionalidades do sistema, como cadastro, atualização e exclusão de produtos.
* Verificação de casos de uso com cenários completos do ciclo de vida do produto.
* Testes de desempenho e escalabilidade para garantir o funcionamento adequado em diferentes cargas de trabalho.

### 5.4 Testes de Serviço (ProdutoService)

1. testaddProduto: Este teste verifica se o método addProduto do serviço ProdutoService adiciona corretamente um produto. Ele cria um objeto Produto com um nome e quantidade específicos, simula a chamada ao método save do repositório, e então verifica se o produto retornado é o mesmo que foi adicionado, garantindo que a operação foi bem-sucedida.
2. testAddProdutoComNomeNulo: Este teste verifica se o método addProduto do serviço lança uma exceção quando o nome do produto é nulo. Ele cria um objeto Produto com um nome nulo e uma quantidade válida, e verifica se uma exceção IllegalArgumentException é lançada durante a execução do método.
3. testAddProdutoComQuantidadeNegativa: Este teste verifica se o método addProduto do serviço lança uma exceção quando a quantidade do produto é negativa. Ele cria um objeto Produto com um nome válido e uma quantidade negativa, e verifica se uma exceção IllegalArgumentException é lançada durante a execução do método.
4. testGetAllProdutos: Este teste verifica se o método getAllProdutos do serviço retorna corretamente uma lista de todos os produtos. Ele cria uma lista de produtos fictícios, simula a chamada ao método findAll do repositório para retornar essa lista, e então verifica se a lista retornada pelo serviço corresponde à lista original.
5. testAddProdutoNotFound: Este teste verifica o comportamento do serviço quando o método save do repositório retorna null. Ele cria um objeto Produto com um nome e quantidade específicos, simula a chamada ao método save do repositório, e verifica se o produto retornado pelo serviço é null, indicando que a operação não foi bem-sucedida.

### 5.5 Testes de Integração (ProdutoRepositoryIntegrationTest)

1. testeSalvarProduto: Este teste verifica se um produto pode ser salvo corretamente no banco de dados. Um produto é salvo no banco de dados e o ID gerado é verificado para garantir que a operação de salvamento foi bem-sucedida.
2. testeEncontrarProdutoPorNome: Este teste verifica se é possível encontrar um produto no banco de dados por seu nome. Um produto é salvo no banco de dados e em seguida é feita uma consulta pelo nome do produto. Os resultados da consulta são verificados para garantir que o produto foi encontrado corretamente.

### 5.6 Testes de Repositório (ProdutoRepositoryCRUDTest)

1. testCreateProduto: Este teste verifica se um produto pode ser criado corretamente no banco de dados. Um produto é criado e salvo no banco de dados e o ID gerado é verificado para garantir que a operação de criação foi bem-sucedida.
2. testReadProduto: Este teste verifica se um produto pode ser lido corretamente do banco de dados. Um produto é salvo no banco de dados e em seguida é feita uma consulta pelo ID do produto. Os resultados da consulta são verificados para garantir que o produto foi lido corretamente.
3. testUpdateProduto: Este teste verifica se um produto pode ser atualizado corretamente no banco de dados. Um produto é salvo no banco de dados, em seguida é atualizado e novamente salvo. Os resultados da consulta são verificados para garantir que o produto foi atualizado corretamente.
4. testDeleteProduto: Este teste verifica se um produto pode ser excluído corretamente do banco de dados. Um produto é salvo no banco de dados, em seguida é excluído e uma nova consulta é feita pelo ID do produto. Os resultados da consulta são verificados para garantir que o produto foi excluído corretamente.

## 6. Recursos de Teste

* Ambientes de desenvolvimento e teste configurados com as dependências necessárias.
* Ferramentas de automação de teste, como JUnit, Mockito e Spring Boot Test.

## 7. Cronograma de Testes

Os testes serão realizados ao longo do ciclo de desenvolvimento, com foco em cada etapa do processo de desenvolvimento e entrega do sistema.

## 8. Responsabilidades

* A equipe de desenvolvimento será responsável por elaborar os casos de teste.
* A equipe de testes será responsável por executar os testes e reportar eventuais problemas.
* A equipe de gerenciamento será responsável por acompanhar o progresso dos testes e tomar medidas corretivas, se necessário.

## 9. Riscos e Considerações

* Alterações nos requisitos podem impactar os casos de teste e exigir revisões adicionais.
* Problemas técnicos podem afetar a execução dos testes e a qualidade dos resultados.

## 10. Métricas de Teste

* Cobertura de código alcançada pelos testes.
* Número de casos de teste executados e bem-sucedidos.
* Tempo médio de execução dos testes.

## 11. Revisão e Aprovação

Este plano de testes será revisado pela equipe de desenvolvimento, testes e gerenciamento para garantir a adequação e a eficácia das atividades de teste.

Esses testes ajudarão a garantir a qualidade e a confiabilidade do seu sistema de gerenciamento de inventário. Certifique-se de executá-los regularmente durante o desenvolvimento para identificar e corrigir quaisquer problemas o mais cedo possível.