SaleSystem

Plano de Testes

Versão 1.0

[O nome do módulo deverá ser informado somente quando aplicável. Caso contrário, deverá ser excluído da capa do documento]

Histórico de Revisão

|  |  |  |  |
| --- | --- | --- | --- |
| **Data** | **Versão** | **Descrição** | **Autor** |
| 26/06/2017 | 1.0 | TESTE DE RISCOS VAI AVALIAR E ANALISAR CPF, DATAS E CAMPOS VAZIOS. | DÊANGELLES, JÉSSYCA MELO E CELSO HENRIQUE |
|  |  |  |  |
|  |  |  |  |

Sumário

[1. Introdução 4](#_Toc486269695)

[2. Estágios de Teste 4](#_Toc486269696)

[3. Tipos de Testes 4](#_Toc486269697)

[4. Recursos necessários 5](#_Toc486269698)

[4.1. Recursos Humanos 5](#_Toc486269699)

[4.2. Recursos Computacionais 5](#_Toc486269700)

[5. Riscos e Restrições 5](#_Toc486269701)

[6. Casos de Teste de Unidade 5](#_Toc486269702)

[6.1. Caso de Teste de Unidade 01 5](#_Toc486269703)

[6.2. Caso de Teste de Unidade 02 6](#_Toc486269704)

[6.3. Caso de Teste de Unidade 03 6](#_Toc486269705)

Plano de Testes

1. Introdução

Este documento relaciona os casos de uso/requisitos funcionais a serem testados, os estágios de testes, método de qualificação, detalhamento dos tipos de testes, alvos de testes, a estratégia adotada para a execução dos testes, os recursos humanos necessários, bem como os casos de teste de unidade a serem executados.

1. Estágios de Teste

Definem o momento do ciclo de vida do software em que são realizados testes por pessoas diferentes daquelas que o programaram. Entretanto, considerando a divisão das tarefas de teste em quatro níveis relacionados ao escopo do software, estão previstos para o projeto **SaleSystem** os seguintes estágios de teste:

* Teste de Unidade: **Checar, avaliar e analisar CPF, DATAS.**
* Teste de Integração: são realizados para verificar basicamente se as unidades testadas de forma individual executam corretamente quando colocadas juntas, isto é, quando integradas. Os testes são realizados pelo Analista de Testes.
* Teste de Sistema: são realizados pelo Analista de Testes, visando a execução do sistema, dentro de um ambiente operacional controlado, para validar a exatidão e perfeição na execução de suas funções.
* Teste de Aceitação ou Homologação: são os testes finais de execução do sistema, realizados pelos usuários, visando verificar se a solução atende aos objetivos do negócio e a seus requisitos, no que diz respeito à funcionalidade e usabilidade, antes da utilização no ambiente de produção.

1. Tipos de Testes

Seguem abaixo os tipos de testes a serem aplicados ao projeto **SaleSystem:**

* **Configuração:** verifica se o software está apto a rodar em diferentes versões ou configurações de ambientes (hardware e software), como, por exemplo, em diferentes browsers.
* **Funcional:** grupos de testes que avaliam se o que foi especificado foi implementado.
* **Integridade de dados:** verificar se os dados do sistema foram incluídos, alterados, excluídos e pesquisados corretamente no banco de dados. Além de validar conteúdos de campos.
* **Performance:** mede e avalia o tempo de resposta de cada transação dos requisitos sensíveis ao tempo.
* **Usabilidade:** verificam o nível de facilidade de uso do software pelos usuários.
* **Regressão:** verifica a ocorrência de novos defeitos após a resolução de defeitos.
* **Acessibilidade**: verifica se a interface do usuário fornece o acesso apropriado às funções do sistema e a navegação adequada. Além disso, estes testes garantem que os objetos dentro da interface do usuário funcionem de acordo com os padrões definidos pelo cliente.
* **Disponibilidade**: avaliam a capacidade do software em continuar operando mesmo quando algum elemento (software ou hardware) fica inoperante ou para de funcionar.

1. Recursos necessários
   1. Recursos Humanos

Utilizaremos dados de clientes para a analise e testes.

* 1. Recursos Computacionais

Um computador com sistema operacional win7 ou superior 32/64 bits, com programas Netbeans, Mysql, Xamp.

|  |  |
| --- | --- |
| **Servidor** | **Nome/Descrição** |
| Msql | Banco de Dados |
| **Estações de Trabalho** | **Nome** |
| Computador |  |
| **Software** | **Nome** |
| Sist.OP, NETBEANS, XAMP |  |

1. Riscos e Restrições
2. Casos de Teste de Unidade
   1. Caso de Teste de Unidade 01

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ESPECIFICAÇÃO DE CASO DE TESTE** | | | | |
| **Projeto** | | SaleSystem | | |
| **Tipo** | | Teste de unidade | | |
| **Requisito Funcional** | | Verificar CPF da tabela cliente | | |
| **Propósito** | | Avalidar CPF | | |
| **Cenários de teste** | | | | |
| Objetivo específico | | | Especificação das entradas ou ações | Saídas esperadas |
| 1 | CPF VALIDO | | 52971457249 | CPF OK |
| 2 | CPF INVALIDO | | 529741524 | CPF INVALIDO |

* 1. Caso de Teste de Unidade 02

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ESPECIFICAÇÃO DE CASO DE TESTE** | | | | |
| **Projeto** | | SaleSystem | | |
| **Tipo** | | Teste de unidade | | |
| **Requisito Funcional** | | Verificar data | | |
| **Propósito** | | Verificar se a data está atualizada no sistesma | | |
| **Cenários de teste** | | | | |
| Objetivo específico | | | Especificação das entradas ou ações | Saídas esperadas |
| 1 | Receber a data atual | | Data atual | Data atual e formatada |
|  |  | |  |  |

7. // -- PRIMEIRO EXEMPLO

8. PACKAGE UTIL;

9.

10. IMPORT JAVA.AWT.CONTAINER;

11. IMPORT ORG.JUNIT.AFTER;

12. IMPORT ORG.JUNIT.AFTERCLASS;

13. IMPORT ORG.JUNIT.BEFORE;

14. IMPORT ORG.JUNIT.BEFORECLASS;

15. IMPORT ORG.JUNIT.TEST;

16. IMPORT STATIC ORG.JUNIT.ASSERT.\*;

17.

18. /\*\*

19. \*

20. \* @AUTHOR MEUS DOCUMENTOS

21. \*/

22. PUBLIC CLASS UTILSTEST {

23.

24. PUBLIC UTILSTEST() {

25. }

26.

27. @BEFORECLASS

28. PUBLIC STATIC VOID SETUPCLASS() {

29. }

30.

31. @AFTERCLASS

32. PUBLIC STATIC VOID TEARDOWNCLASS() {

33. }

34.

35. @BEFORE

36. PUBLIC VOID SETUP() {

37. }

38.

39. @AFTER

40. PUBLIC VOID TEARDOWN() {

41. }

42.

43.

46. @TEST

47. // TESTAR SE O CP É VALIDO

48. PUBLIC VOID TESTECPF() THROWS EXCEPTION {

49. SYSTEM.OUT.PRINTLN("VERIFICAR\_CPF");

50. STRING STR = "52971457249";

51. UTILS.VERIFICACPF(STR);

52. // TODO REVIEW THE GENERATED TEST CODE AND REMOVE THE DEFAULT CALL TO FAIL.

53. }

54. // CPF INVALIDO

55. PUBLIC VOID TESTECPF() THROWS EXCEPTION {

49. SYSTEM.OUT.PRINTLN("VERIFICAR\_CPF");

50. STRING STR = "52145245";

51. UTILS.VERIFICACPF(STR);

59. // TODO REVIEW THE GENERATED TEST CODE AND REMOVE THE DEFAULT CALL TO FAIL.

60. }

61.

68.

69. // --- TESTAR DATA

70. @TEST

71. PUBLIC VOID VERIFICAR\_DATA() {

72. SYSTEM.OUT.PRINTLN("DATA\_ATUAL\_EXECULTADA");

73. LOCALDATE DATA = NULL;

74. DATA = LOCALDATE.NOW();

75. FORMATADATAS INSTANCE = NEW FORMATADATAS();

76. STRING EXPRESULT = "03/06/2017";

77. STRING RESULT = INSTANCE.DATA\_PARA\_EXIBIRNATELA(DATA);

}