<IFRO-TECA>

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** |
| <30/06/2017> | <1.0> | TESTE DE RISCOS EM VALIDACAO DE CPF, DATA, VALIDACAO DE PREEENCHIMENTO DE CAMPOS | Josileno Robeto |
|  |  |  |  |
|  |  |  |  |

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 recursoshumanos 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 realizadostestes 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 **CONTROLE DE BIBLIOTECA DO INSTITUTO TEC. FEDERAL DE RONDONIA ( IFROTECA )** os seguintes estágios de teste:

* Teste de Unidade: **Checamos se o campo nome ficou em branco, se o CPF é valido, e se a data será apresentada na tela conforme o esperado acrescimo de 30 dias.**
* 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 **IFROTECA**

* **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

Obtivemos CPF de indivíduos para testar, deixamos usuários digitarem em campos testando a possibilidade de não conseguirem avançar o cadastro com campos em branco.

* 1. Recursos Computacionais

Usamos computador com netbeans e Java 8.111 instalados

|  |  |
| --- | --- |
| **Servidor** | **Nome/Descrição** |
|  |  |
| **Estações de Trabalho** | **Nome** |
|  |  |
| **Software** | **Nome** |
|  |  |

1. Riscos e Restrições

[Listar quaisquer riscos, restrições ou contingências que possam afetar o projeto, desenvolvimento ou implementação do teste. Caso esteja em outro documento fazer a referência ao outro documento.EX: Os possíveis riscos que possam afetar o projeto estão relatados no documento/artefato Checklist dos Riscos.

1. Casos de Teste de Unidade
   1. Caso de Teste de Unidade 01 - testValida\_CPF()

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ESPECIFICAÇÃO DE CASO DE TESTE** | | | | |
| **Projeto** | | IFROTECA | | |
| **Tipo** | | Teste de unidade | | |
| **Requisito Funcional** | | VALIDACAO DE CPF – CADASTRO ALUNO | | |
| **Propósito** | | Verificar validade do CPF | | |
| **Cenários de teste** | | | | |
| Objetivo específico | | | Especificação das entradas ou ações | Saídas esperadas |
| 1 | CPF VALIDO | | 34987932253 | CPF VALIDO |
| 2 | CPF faltando um numero | | 0012554167 | CPF INVALIDO |
| 3 | CPF EM BRANCO | |  | CPF INVALIDO |
|  |  | |  |  |

* 1. Caso de Teste de Unidade 02 - testData\_Para\_ExibirNaTela()

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ESPECIFICAÇÃO DE CASO DE TESTE** | | | | |
| **Projeto** | | IFROTECA | | |
| **Tipo** | | Teste de unidade | | |
| **Requisito Funcional** | | VALIDACAO DATA | | |
| **Propósito** | | PROPOR O ACRECIMO DE 30 DIAS A DATA DO DIA | | |
| **Cenários de teste** | | | | |
| Objetivo específico | | | Especificação das entradas ou ações | Saídas esperadas |
| 1 | Tratar a data de empréstimo do item da biblioteca acrescentando um prazo para devolução do mesmo | | 30 dias | Sistema deverá tratar esta data como constante (padrão) a contar da data de retirada do item do acervo |
|  |  | |  |  |

* 1. Caso de Teste de Unidade 03

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ESPECIFICAÇÃO DE CASO DE TESTE** | | | | |
| **Projeto** | | IFROTECA | | |
| **Tipo** | | Teste de unidade | | |
| **Requisito Funcional** | | Obter a data e hora atuais do sistema operacional como ponto de partida para incluir em registros gravados no banco de dados com intuito de manter armazenadas as datas exatas de entradas ou saídas de itens do acervo. | | |
| **Propósito** | | Propósito geral do caso de teste. Conferir se a data a ser gravada em banco de dados esta correta. | | |
| **Cenários de teste** | | | | |
| Objetivo específico | | | Especificação das entradas ou ações | Saídas esperadas |
| 1 | Receber a data do sistema operacional e guardar em variável para posterior armazenamento no banco de dados | | Data do dia | 30/06/2017 |
|  |  | |  |  |

1. // -- primeiro exemplo
2. package Util;
3. import java.awt.Container;
4. import org.junit.After;
5. import org.junit.AfterClass;
6. import org.junit.Before;
7. import org.junit.BeforeClass;
8. import org.junit.Test;
9. import static org.junit.Assert.\*;
10. /\*\*
11. \*
12. \* @author Meus Documentos
13. \*/
14. public class UtilsTest {
16. public UtilsTest() {
17. }
19. @BeforeClass
20. public static void setUpClass() {
21. }
23. @AfterClass
24. public static void tearDownClass() {
25. }
27. @Before
28. public void setUp() {
29. }
31. @After
32. public void tearDown() {
33. }
34. /\*\*
35. \* Test of limpa\_campo method, of class Utils.
36. \*/
37. @Test
38. // CPF VALIDO
39. public void testValida\_CPF() throws Exception {
40. System.out.println("valida\_CPF");
41. String str = "34987932253";
42. Utils.valida\_CPF(str);
43. // TODO review the generated test code and remove the default call to fail.
44. }
45. // CPF faltando um numero 0012554167
46. public void testValida\_CPF2() throws Exception {
47. System.out.println("valida\_CPF");
48. String str = "0012554167";
49. Utils.valida\_CPF(str);
50. // TODO review the generated test code and remove the default call to fail.
51. }
52. // CPF EM BRANCO
53. public void testValida\_CPF3() throws Exception {
54. System.out.println("valida\_CPF");
55. String str = "";
56. Utils.valida\_CPF(str);
57. // TODO review the generated test code and remove the default call to fail.
58. }
60. // --- segundo exemplo
61. @Test
62. public void testData\_Para\_ExibirNaTela() {
63. System.out.println("Data\_Para\_ExibirNaTela");
64. LocalDate data = null;
65. data = LocalDate.now();
66. FormataDatas instance = new FormataDatas();
67. String expResult = "30/06/2017";
68. String result = instance.Data\_Para\_ExibirNaTela(data);
69. assertEquals(expResult, result);
70. // TODO review the generated test code and remove the default call to fail.
71. }
73. // 3 exemplo@Test
74. public void testValidarPreenchimento() {
75. System.out.println("validarPreenchimento");
76. String texto = "";
77. Utils instance = new Utils();
78. boolean expResult = false;
79. boolean result = instance.validarPreenchimento(texto);
80. assertEquals(expResult, result);
81. // TODO review the generated test code and remove the default call to fail.
82. }