 <b>MINERVA</b> SOLUTIONS		<b>Relatório de Resultado de Teste e Inspeção: Iteração 4</b>
<b>Data:</b> 09/12/2018	<b>Objetivo Estratégico:</b> Gerenciamento dos dados administrativos do Instituto Ada Lovelace.	
<b>ID:</b> 1011	<b>Nome do Projeto:</b> sigAda - Sistema de Gerenciamento do Ada	
<b>CC:</b> Paulo Pires	<b>Cliente:</b> Instituto Ada Lovelace	
<b>Patrocinador:</b> UFRJ	<b>Gerente de Projeto:</b> Rafael Cardoso	

## 1. Histórico de Versão

---

Data	Versão	Descrição	Autor(es)
09/12/2018	1.0.0	Abertura do Documento, adição dos itens 2 e 3.	Rafael, Eduardo, Tainá
10/12/2018	2.0.0	Adição do resultado de novos testes.	Eduardo, Rafael

## 2. Introdução

---

### 2.1 Finalidade

Este documento tem como finalidade apresentar:

- Os resultados dos testes e das inspeções, bem como os relatórios gerados pelas ferramentas utilizadas.
- Será apresentado possíveis soluções para os erros e falhas

### 2.2 Escopo

Este documento diz respeito aos resultados dos testes e das inspeções referentes a 4º iteração, bem como possíveis soluções para cada um dos erros e falhas detectadas. Os testes e as inspeções, que geraram o resultados presente nesse documento, foram

realizados seguindo o planejamento do plano de teste e inspeção da 4ª iteração, cujo o link está disponível no tópico de referências.

## 2.3 Visão Geral

Este documento apresenta uma estrutura inicial parecida com os outros documentos já gerados, ou seja, inicia-se com o histórico de versões e em seguida com uma breve introdução sobre o objetivo da construção de tal documento, bem como qual é seu escopo e referências para outros documentos, estes posteriormente mencionados ao longo do documento.

Após a introdução, segue um resumo contendo os resultados de todos os testes e inspeções para a 4ª iteração e por fim é apresentado possíveis soluções para cada um dos problemas encontrado.

## 2.4 Referências

[Plano de Testes e Inspeção: Iteração 4](#)

# 3. Resumo de Resultados e Ações Sugeridas do Teste e da Inspeção

---

## 3.1 Verificação dinâmica

Tipo:	Teste de Unidade
Teste:	Verificar se está sendo aceito apenas CPFs válidos.
Unidade:	Método verifyCPF(String cpf) da classe ManterAlunoForm.
Entradas e saídas esperadas:	Foram realizadas 10 entradas do tipo String. Entradas com saída esperada igual a "true": "47763581077", "49870235123", "05093936081", "13372061055". Entradas com saída esperada igual a false: "11111111111", "87415447545", "65485475422", "89754124586", "34651238569", "abcdefrthyu".
Estratégia de teste:	White box
Resultados:	Apenas o teste correspondente a entrada "abcdefrthyu" falhou, não retornando a saída esperada. No mesmo, ocorreu uma

	exceção, uma vez que o método não tratou Strings que continham caracteres alfabéticos.
Ações Sugeridas:	Fazer o tratamento da exceção quando a string recebida não contém somente dígitos.

Tipo:	Teste de Unidade
Teste:	Verificar se as solicitações ao banco de dados estão sendo feitas corretamente.
Unidade:	salvar() e consultaAlunos() da classe AlunoDAO.
Entradas e saídas esperadas:	Foi realizada a inserção de um suposto aluno e a saída esperada é que uma consulta, feita posteriormente à essa inserção, deve possuir o aluno inserido. Deste modo, foi verificado tanto o método salvar quanto o consultaAlunos ao mesmo tempo.
Estratégia de teste:	White box
Resultados:	Passou no teste com sucesso.
Ações Sugeridas:	-

Tipo:	Teste de Unidade
Teste:	Verificar se as solicitações ao banco de dados estão sendo feitas corretamente.
Unidade:	remover() da classe AlunoDAO.
Entradas e saídas esperadas:	Foi dado como entrada o mesmo aluno inserido no teste anterior (salvar() e consultaAlunos()), duas vezes seguidas, onde o resultado do primeiro pedido de remoção deveria ser true e do segundo false.
Estratégia de teste:	White box

Resultados:	O segundo pedido de remoção falhou, pois deveria retornar false ao invés de true, indicando que não havia tal aluno para ser removido.
Ações Sugeridas:	Verificar se o objeto a ser removido se encontra, de fato, no banco de dados e retornar falso caso não esteja.

Tipo:	Teste de Integridade
Teste:	Verificar se o nome de usuário digitado no form de login será mostrado no form de menu.
Integração:	Integração da view LoginForm com a view MenuRHForm
Estratégia de teste:	Black box
Resultados:	Passou com sucesso
Ações Sugeridas:	-

Tipo:	Teste de Integridade
Teste:	Verificar se ao clicar para fazer uma ação referente à um aluno no form de menu, a janela referente a essa ação será exibida.
Integração:	Integração da view MenuRHForm com a view ManterAlunoForm.
Estratégia de teste:	Black box
Resultados:	Passou com sucesso
Ações Sugeridas:	-

Tipo:	Teste de Integridade
Teste:	Verificar se ao clicar para fazer uma ação referente à um funcionário no form de menu, a janela referente a essa ação será exibida.
Integração:	Integração da view MenuRHForm com a view ManterFuncionarioForm.
Estratégia de teste:	Black box
Resultados:	Falhou. Não ocorreu ação nenhuma.
Ações Sugeridas:	-

Tipo:	Teste de Integridade
Teste:	Verificar se ação referente aos botões salvar, limpar, ou deletar estão funcionando de acordo.
Integração:	Integração da view ManterAlunoForm com o controller AlunoController e o model AlunoDAO.
Estratégia de teste:	Black box
Resultados:	Passou com sucesso.
Ações Sugeridas:	-

Tipo:	Teste de Integridade
Teste:	Verificar se ao logar-se como funcionário do R.H., o menu direcionado para o mesmo será exibido.

Integração:	Integração da view LoginForm com a view MenuRHForm.
Estratégia de teste:	Black box
Resultados:	Passou com sucesso.
Ações Sugeridas:	-

Tipo:	Teste de Sistema
Teste:	Verificar se a autenticação do login está ocorrendo de maneira correta, ou seja, quem está cadastrado, é logado e quem não está, não é.
Conformidade:	Conformidade com o caso de uso CS22
Estratégia de teste:	Black box
Resultados:	Passou com sucesso.
Ações Sugeridas:	-

Tipo:	Teste de Sistema
Teste:	Verifica se o processo de manter aluno, ou seja, cadastrar um novo aluno, alterar suas informações e, em seguida, buscá-lo e deletá-lo está funcionando da forma como esperada.
Conformidade:	Conformidade com o caso de uso CS01
Estratégia de teste:	Black box
Resultados:	Passou com sucesso.
Ações Sugeridas:	-

Tipo:	Teste de Release
-------	------------------

Teste:	Verifica se as funcionalidades desenvolvidas e testadas nas interações anteriores (teste de unidade, teste de integração e teste de sistema) estão funcionando como o esperado, assim como o desempenho do software, conforme a especificação. Como para essa iteração não temos testes anteriores foram só feitos testes de desempenho.
Estratégia de teste:	White/Black box
Resultados:	Passou com sucesso.
Ações Sugeridas:	-

### 3.2 Verificação estática

Após o uso da ferramenta Findbug, foi gerado o relatório apresentado na figura 1, onde é apresentado os possíveis problemas encontrados no código.

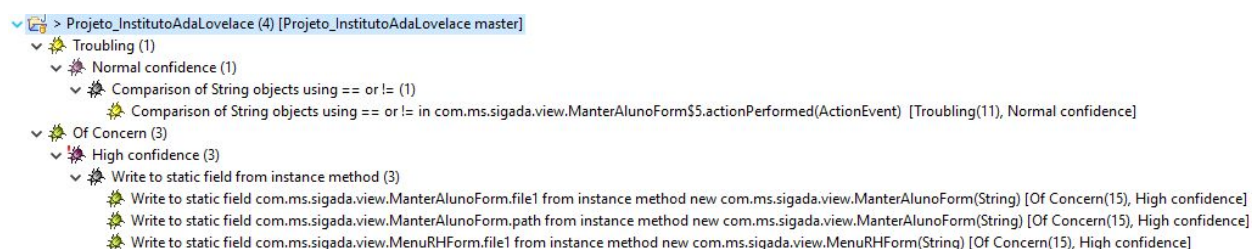


Figura 1: Resumo dos testes com o Findbugs. A imagem ampliada pode ser visualizada [aqui](#).

Tipo:	Análise estática automatizada
Categoria:	Má prática
Resultado:	<a href="#">Link</a>
Descrição:	Esse bug foi encontrado na classe ManterAlunoForm. Os operadores utilizados para fazer comparações, como os “==” e “!=”, entre “Strings” se caracteriza como um bug de má prática.
Ações Sugeridas:	Uma alternativa para resolver esse problema, seria usar o método “equals(Object)” para fazer comparações entre “Strings”.

Tipo:	Análise estática automatizada
Categoria:	Corretude
Resultado:	Passou com sucesso
Descrição:	-
Ações Sugeridas:	-

Tipo:	Análise estática automatizada
Categoria:	Performance
Resultado:	Passou com sucesso
Descrição:	-
Ações Sugeridas:	-

Tipo:	Análise estática automatizada
Categoria:	Evasivo
Resultado:	<a href="#">Link</a>
Descrição:	Esse bug foi encontrado na classe ManterAlunoForm. A declaração da variável “file1” como uma variável estática e reescritura da mesma ao longo do código, se caracteriza como um bug evasivo (dodgy), pois a forma em que foi feita pode gerar erros futuros.
Ações Sugeridas:	Uma alternativa para resolver esse bug seria declarar a variável “file1” sem o “static”.

Tipo:	Análise estática automatizada
-------	-------------------------------



Categoria:	Evasivo
Resultado:	<a href="#">Link</a>
Descrição:	Esse bug foi encontrado na classe ManterAlunoForm. A declaração da variável “path” como uma variável estática e reescritura da mesma ao longo do código, se caracteriza como um bug evasivo (dodgy), pois a forma em que foi feita pode gerar erros futuros.
Ações Sugeridas:	Uma alternativa para resolver esse bug seria declarar a variável “path” sem o “static”.

Tipo:	Análise estática automatizada
Categoria:	Evasivo
Resultado:	<a href="#">Link</a>
Descrição:	Esse bug foi encontrado na classe MenuRHForm. A declaração da variável “file1” como uma variável estática e reescritura da mesma ao longo do código, se caracteriza como um bug evasivo (dodgy), pois a forma em que foi feita pode gerar erros futuros.
Ações Sugeridas:	Uma alternativa para resolver esse bug seria declarar a variável “file1” sem o “static”.

Tipo:	Análise estática automatizada
Categoria:	Corretude Multithreading
Resultado:	Passou com sucesso
Descrição:	-
Ações Sugeridas:	-

## 4. Diagramas

---

Gráfico de Sucessos e Falhas

