



UNIVERSIDADE FEDERAL DO AMAZONAS  
BACHARELADO EM ENGENHARIA DA COMPUTAÇÃO

# IsenSys - Processador de Solicitações de Isenção de Taxa de Inscrição em Concursos Públicos

Felipe André Souza da Silva

Manaus - AM  
Novembro de 2023

Felipe André Souza da Silva

# IsenSys - Processador de Solicitações de Isenção de Taxa de Inscrição em Concursos Públicos

Monografia de Graduação apresentada à Faculdade de Tecnologia da UFAM como requisito parcial para a obtenção do grau de bacharel em Engenharia da Computação.

Orientador:

Dr. Edson Nascimento Silva Júnior

Universidade Federal do Amazonas

Manaus - AM

Novembro de 2023

Monografia de Graduação sob o título *IsenSys - Processador de Solicitações de Isenção de Taxa de Inscrição em Concursos Públicos* apresentada por Felipe André Souza da Silva e aceita pela Faculdade de Tecnologia da Universidade Federal do Amazonas, sendo aprovada por todos os membros da banca examinadora abaixo especificada:

---

Dr. Edson Nascimento Silva Júnior  
Instituto de Computação  
Universidade Federal do Amazonas

---

Dra. Fabíola Guerra Nakamura  
Instituto de Computação  
Universidade Federal do Amazonas

---

Dr. José Francisco Magalhães Netto  
Instituto de Computação  
Universidade Federal do Amazonas

Manaus - AM, 08 de Novembro de 2023.

*À minha mãe, mulher guerreira e fonte de inspiração e forças para conclusão deste curso de graduação.*

# Agradecimentos

Em primeiro lugar quero agradecer à minha família que é pequena, mas super unida, minha mãe Adriana, aos meus irmãos Anna Beatriz e Maurício Júnior (*in memoriam*) e nossa amiga Nira, pelo amor incondicional, apoio emocional e encorajamento constante que me proporcionaram durante todos esses anos. Vocês são minha âncora e minha maior motivação.

Aos meus grandes amigos: Alexsandro Evangelista, Arianne Kaist, Hítalo Viana, Leonardo Pinheiro, Maurianne Kaist, Milena Lizandra, Munhoz, Naiara, Nilba, Tiago e Will que compartilharam comigo as alegrias, desafios e preocupações deste percurso, agradeço pela amizade, colaboração, apoio mútuo e por não me permitir enlouquecer.

Também expresso minha gratidão ao meu orientador, Prof. Edson Jr. pela orientação, PACIÊNCIA, dedicação e perseverança ao longo deste processo. Suas orientações e *insights* e sua motivação foram fundamentais para o desenvolvimento deste trabalho e para o meu crescimento como acadêmico e profissional.

Por último, mas não menos importante, quero agradecer a todos os membros do corpo docente, distribuídos nos mais diversos departamentos por onde passei como estudante, cujo conhecimento e ensinamentos moldaram minha jornada acadêmica. Suas aulas foram inspiradoras e enriquecedoras, contribuindo muito para meu crescimento acadêmico.

A conclusão deste ciclo não teria sido possível sem a contribuição de cada uma dessas pessoas, e por isso, expresso minha profunda gratidão a todos. O conhecimento adquirido e as experiências vividas durante esta jornada acadêmica são inestimáveis, e estou ansioso para aplicá-los em meu futuro profissional.

Muito obrigado a todos por fazerem parte desta conquista!

*Software is a great combination between artistry and engineering.*

*Bill Gates*

# IsenSys - Processador de Solicitações de Isenção de Taxa de Inscrição em Concursos Públicos

Autor: Felipe André Souza da Silva

Orientador: Dr. Edson Nascimento Silva Júnior

## Resumo

Este documento versa sobre o desenvolvimento de um aplicativo computacional para processar solicitações de isenção de taxa de inscrição em concursos públicos de acordo com as normativas do Ministério do Desenvolvimento Social do Brasil e os interesses da Comissão Permanente de Concursos da UFAM. O sistema, que opera coletivamente com o Sistema de Isenção de Taxa de Concurso, do Ministério do Desenvolvimento Social do Brasil, permite que um órgão gestor prepare dados pessoais de candidatos solicitantes de isenção de taxa de inscrição para envio ao sistema do Ministério do Desenvolvimento, e após o processamento de tais solicitações pelo sistema, gere editais de publicação e relatórios com o objetivo de garantir a lisura e transparência deste processo tão democrático. No desenvolvimento foi utilizada a linguagem de programação *Java* e tecnologias de grande consolidação no mercado como o *Jasper Reports*, para geração de relatórios e o *Apache POI*, adicionando suporte a arquivos do *Microsoft Excel*.

*Palavras-chave:* taxa de inscrição, isenção, concurso público, Java.

# IsenSys - A Processor of Public Examination Subscription Fee Requests for Exemption

Author: Felipe André Souza da Silva

Supervisor: Dr. Edson Nascimento Silva Júnior

## Abstract

This document relates to the development of a computer application that facilitates applying for the waiving of fees when sitting public examinations. This is in accordance with the guidelines set by the Brazilian Ministry of Social Development and the requirements of the UFAM Permanent Commission for Examinations. The app system works with the System of Exemption Fees of the Ministry, allowing anyone concerned to handle the personal data of candidates applying for exemption, in order for it to be sent to the Ministry's system. After this process, it creates a public notice and reports, guaranteeing smoothness and transparency in the democratic process. *Java* programming language was used in the development of the app. Other compatible technologies, such as *Jasper Reports*, was used to generate reports and the *Apache POI*, for the processing of *Microsoft Excel* files.

*Keywords:* examination fee, exemption, public examination, Java.



# Lista de ilustrações

Figura 1 – Logomarca do <i>Java</i> . . . . .	17
Figura 2 – Logo do gerente de projetos <i>Apache Maven</i> . . . . .	19
Figura 3 – Logo da biblioteca <i>Apache POI</i> . . . . .	19
Figura 4 – <i>Eclipse IDE</i> . . . . .	20
Figura 5 – Suíte de desenvolvimento <i>JasperReports®</i> . . . . .	21
Figura 6 – Tela Inicial do <i>IsenSys</i> . . . . .	26
Figura 7 – Tela de Configurações do Sistema . . . . .	27
Figura 8 – Modelagem da entidade <i>Candidato</i> . . . . .	28
Figura 9 – Diagrama de Classe de <i>CandidatoBuilder</i> . . . . .	29
Figura 10 – Diagrama de Classe de <i>RowParseException</i> . . . . .	30
Figura 11 – Diagrama de Classe de <i>ParseResult</i> . . . . .	31
Figura 12 – Diagrama de Classe de <i>CSVSheetReader</i> . . . . .	32
Figura 13 – Diagrama de Classe de <i>ExcelSheetReader</i> . . . . .	33
Figura 14 – Diagrama de Classe de <i>CSVSheetWriter</i> . . . . .	33
Figura 15 – Diagrama de Classe de <i>ExcelSheetWriter</i> . . . . .	34
Figura 16 – Diagrama do Fluxo de Dados das Solicitações . . . . .	35
Figura 17 – Tela do Módulo de Envio . . . . .	36
Figura 18 – Tela do Módulo de Envio (arquivo carregado) . . . . .	36
Figura 19 – Tela do Módulo de Envio (preenchida) + Arquivos de Saída . .	37
Figura 20 – Modelagem da Entidade <i>Retorno</i> . . . . .	38
Figura 21 – Diagrama de classe de <i>ListaRetornos</i> . . . . .	39
Figura 22 – Diagrama de Classe de <i>Compilation</i> . . . . .	39
Figura 23 – Modelagem da Entidade <i>Situacao</i> . . . . .	39
Figura 24 – Diagrama de Classe de <i>SituacaoDAO</i> . . . . .	40
Figura 25 – Diagramas de Classe dos Geradores de Relatório . . . . .	40
Figura 26 – Diagrama do Fluxo de Dados no Retorno Preliminar . . . . .	41
Figura 27 – Tela do Módulo de Retorno Preliminar . . . . .	42
Figura 28 – Tela do Módulo de Retorno Preliminar (preenchida) . . . . .	43
Figura 29 – Arquivos de Saída do Módulo de Retorno Preliminar . . . . .	44
Figura 30 – Diagrama do Fluxo de Dados no Retorno Definitivo . . . . .	45

Figura 31 – Tela do Módulo de Retorno Definitivo . . . . .	46
Figura 32 – Diagrama de Classe de uma Similaridade . . . . .	48
Figura 33 – Diagrama do Fluxo de Dados no Relatório . . . . .	49

# Lista de tabelas

Tabela 1 – Dados pessoais de candidatos e seus formatos . . . . .	28
Tabela 2 – Diagrama de Classe de <i>JaroWinkler</i> . . . . .	48

# Lista de abreviaturas e siglas

CSV	<i>Comma-separated values</i> - valores separados por vírgula
COMPEC	Comissão Permanente de Concursos
MDS	Ministério do Desenvolvimento Social
NIS	Número de Identificação Social
SISTAC	Sistema de Isenção de Taxa de Concurso
UFAM	Universidade Federal do Amazonas

# Sumário

<b>1</b>	<b>INTRODUÇÃO</b>	<b>14</b>
<b>1.1</b>	<b>Objetivos</b>	<b>15</b>
1.1.1	Objetivo Geral	15
1.1.2	Objetivos Específicos	15
<b>1.2</b>	<b>Organização da Monografia</b>	<b>15</b>
<b>2</b>	<b>REFERENCIAL TEÓRICO</b>	<b>17</b>
<b>2.1</b>	<b>Java</b>	<b>17</b>
<b>2.2</b>	<b>Apache Commons Text™</b>	<b>18</b>
<b>2.3</b>	<b>Apache Maven</b>	<b>18</b>
<b>2.4</b>	<b>Apache POI</b>	<b>19</b>
<b>2.5</b>	<b>Eclipse IDE</b>	<b>19</b>
<b>2.6</b>	<b>JasperReports®</b>	<b>20</b>
<b>2.7</b>	<b>Jaro-Winkler</b>	<b>21</b>
<b>2.8</b>	<b>Joda-Time</b>	<b>22</b>
<b>2.9</b>	<b>Processo de Isenção</b>	<b>23</b>
<b>3</b>	<b>DESENVOLVIMENTO DO ISENSYS</b>	<b>24</b>
<b>3.1</b>	<b>Tela Inicial</b>	<b>26</b>
<b>3.2</b>	<b>Tela de Configurações do Sistema</b>	<b>26</b>
<b>3.3</b>	<b>Desenvolvimento do Módulo de Envio</b>	<b>27</b>
3.3.1	Modelagem de um Candidato	27
3.3.2	A classe <i>CandidatoBuilder</i>	28
3.3.3	A classe de exceção <i>FieldParseException</i>	29
3.3.4	A classe de exceção <i>RowParseException</i>	30
3.3.5	A classe <i>ParseResult</i>	30
3.3.6	Importadores de Dados de Candidato	31
3.3.6.1	O importador <i>CSVSheetReader</i>	32
3.3.6.2	O importador <i>ExcelSheetReader</i>	32
3.3.7	Exportadores de Dados	32

3.3.7.1	O exportador <i>CSVSheetWriter</i> . . . . .	33
3.3.7.2	O exportador <i>ExcelSheetWriter</i> . . . . .	33
3.3.8	Integração do Módulo de Envio . . . . .	34
3.3.9	Interface Gráfica do Módulo de Envio . . . . .	34
<b>3.4</b>	<b>Desenvolvimento dos Módulos de Retorno . . . . .</b>	<b>37</b>
3.4.1	Modelagem de um Retorno . . . . .	38
3.4.2	A classe <i>ListaRetornos</i> . . . . .	38
3.4.3	A classe <i>Compilation</i> . . . . .	38
3.4.4	Modelagem de uma Situação . . . . .	38
3.4.5	A classe <i>SituacaoDAO</i> . . . . .	39
3.4.6	Geradores de Relatório . . . . .	40
3.4.7	Integração dos Módulos de Retorno . . . . .	40
3.4.7.1	Retorno Preliminar . . . . .	40
3.4.7.2	Tela do Módulo de Retorno Preliminar . . . . .	41
3.4.7.3	Retorno Definitivo . . . . .	44
3.4.7.4	Tela do Módulo de Retorno Definitivo . . . . .	45
<b>3.5</b>	<b>Relatório de Similaridade . . . . .</b>	<b>47</b>
3.5.1	Modelagem de uma Similaridade . . . . .	47
3.5.2	A classe <i>JaroWinkler</i> . . . . .	48
3.5.3	Montagem do Relatório de Similaridade . . . . .	48
<b>3.6</b>	<b>Diretório de Recursos do <i>IsenSys</i> . . . . .</b>	<b>49</b>
<b>4</b>	<b>IMPLEMENTAÇÃO E RESULTADOS . . . . .</b>	<b>51</b>
<b>5</b>	<b>CONCLUSÕES . . . . .</b>	<b>52</b>
<b>5.1</b>	<b>Considerações Finais . . . . .</b>	<b>52</b>
<b>5.2</b>	<b>Propostas de Atualizações . . . . .</b>	<b>52</b>
	<b>REFERÊNCIAS . . . . .</b>	<b>54</b>

# 1 Introdução

Com a missão de cultivar o saber em todas as áreas do conhecimento por meio do ensino, pesquisa e da extensão, a Universidade Federal do Amazonas (UFAM) é uma das principais portas de entrada para o desenvolvimento pessoal e intelectual, contando com cerca de 29.000 alunos e 3.400 servidores distribuídos em seis *campi* ao redor do Estado do Amazonas, em 2023.

Tomando como objeto de estudo e inspiração para este trabalho, um setor específico desta universidade foi adotado: a Comissão Permanente de Concursos (COMPEC), que é um órgão suplementar responsável pela execução dos principais processos seletivos de graduação e concursos para provimento de cargos da universidade.

Uma das tarefas mais democráticas e delicadas executadas por este setor é o processo de isenção de pagamento de taxa de inscrição em concursos e processos seletivos.

Atualmente a COMPEC, como qualquer outro órgão do poder executivo do Brasil, adota três tipos de modalidades de isenção: por cadastro no Registro Brasileiro de Doadores Voluntários de Medula Óssea (REDOME), por comprovação de baixa renda e curso de nível médio de forma gratuita e por meio do Cadastro Único para Programas Sociais do Governo Federal (CadÚnico).

Um dos desafios enfrentados pela COMPEC é a gerência e correto processamento das solicitações de isenção, de forma a não prejudicar os candidatos, tampouco a imagem da UFAM e do funcionalismo público. Para ilustrar, apenas em 2023 a COMPEC realizou 10 concursos, mobilizando ao total 39.289 candidatos, onde 8.353 deles tiveram isenção de taxa de inscrição concedida.

Com o intuito de automatizar e otimizar tal processo, este trabalho apresenta uma aplicação de computador capaz de analisar dados, processar e gerar relatórios e editais de publicação, tomando como objeto de estudo a modalidade de isenção mais volumosa em termos de solicitação: a modalidade CadÚnico, regulamentada pelo Decreto nº 6.593, de 2 de outubro de 2008 [1].

A aplicação, denominada *IsenSys*, procura ainda fornecer uma interface simples e objetiva, com dicas e tratamentos de forma a instruir intuitivamente

sua utilização ao usuário, tomando ainda como alicerce no seu desenvolvimento, os cinco princípios fundamentais da Administração Pública do Brasil: legalidade, impessoalidade, moralidade, publicidade e eficiência.

## 1.1 Objetivos

### 1.1.1 Objetivo Geral

Esta monografia possui como objetivo apresentar um aplicativo processador de solicitações de isenção de taxa de inscrição de acordo com a regulamentação do CadÚnico [1], de forma a permitir agilidade e acurácia nos resultados, por parte de um órgão gestor do Poder Executivo do Brasil.

### 1.1.2 Objetivos Específicos

- Implementar à risca o motor do sistema utilizando as normativas do manual do SISTAC [2];
- Utilizar das boas práticas de programação para tornar o projeto simples e permitir colaboração;
- Implementar uma interface gráfica simples e intuitiva, com dicas e tratamentos de exceções.

## 1.2 Organização da Monografia

Esta monografia possui a seguinte estrutura de capítulos:

- **Capítulo 2:** aborda os fundamentos teóricos, principais tecnologias e bibliotecas utilizadas no desenvolvimento do *IsenSys*;
- **Capítulo 3:** detalha o processo de desenvolvimento do *IsenSys*, por meio de diagramas de classes, detalhamento de suas funcionalidades e exibição da interface gráfica;
- **Capítulo 4:** descreve resultados e métricas que permitem ter um comparativo entre o antes e depois da implementação do *IsenSys*;



- **Capítulo 5:** apresenta as considerações finais, conclusão e futuras atualizações no *IsenSys*.

## 2 Referencial Teórico

Este capítulo referencia as principais ferramentas e tecnologias utilizadas no desenvolvimento do *IsenSys*, tais como linguagem de programação, bibliotecas de funções e o ambiente de desenvolvimento. A última seção deste capítulo explica e detalha o funcionamento do processo de isenção adotado pela COMPEC.

### 2.1 Java

A linguagem de programação Java [3] é uma das mais bem conceituadas e utilizadas ao redor do mundo. Concebida em meados de 1995 pela empresa *Sun Microsystems*, tem conquistado o mundo pela sua simplicidade, boa curva de aprendizagem, forma de organização e versatilidade entre os vários sistemas operacionais e dispositivos.

Por ser uma linguagem independente de plataforma, ela permite que uma mesma aplicação possa ser executada em diversos equipamentos portando sistemas operacionais de diversas arquiteturas, sem a necessidade de adaptação ou reconstrução de código por parte do desenvolvedor, comportamento que torna suas aplicações escaláveis e robustas.

Figura 1 – Logomarca do *Java*



Fonte: <<https://www.oracle.com/br/java/technologies/java-se-glance.html>>

Atualmente mantida pela empresa *Oracle Corporation*, a linguagem continua sendo livre e gratuita para utilização pessoal e para algumas classes de aplicações, e possui uma rica e extensa comunidade de suporte e documentação. As atualizações regulares também são gratuitas e sempre trazem otimizações, novas funcionalidades e melhorias de segurança.

Provavelmente o leitor já tenha utilizado algumas das aplicações implementadas utilizando a tecnologia Java, programas como: declaração de imposto sobre a renda (IRPF e IRPJ), Processo Judicial Eletrônico (PJe), *MATLAB*, famoso no mundo da engenharia e até o próprio sistema *Android*, um dos principais sistemas operacionais para dispositivos móveis.

## 2.2 Apache Commons Text™

O *Apache Commons Text*™ [4] é uma biblioteca livre (*open-source*) escrita em linguagem Java, desenvolvida e mantida pela *Apache Software Foundation*. Ela oferece funcionalidades para manipulação e processamento de texto, tornando tarefas comuns de manipulação de strings mais eficientes e simples para desenvolvedores Java.

Algumas das funcionalidades da biblioteca são: manipulação e escapeamento de *Strings*, algoritmos de distância e similaridade, tokenização, geração de senhas aleatórias, formatação em geral, dentre muitas outras.

## 2.3 Apache Maven

Até o presente momento foram citadas algumas bibliotecas utilizadas como complemento de código para a linguagem Java, prática muito comum entre os desenvolvedores em qualquer linguagem de programação. Porém, muitas vezes sua gerência pode ser complicada se realizada de forma manual, pois bibliotecas resultam em arquivos, que podem sofrer corrompimento ou passar por atualizações de versão.

O *Apache Maven* [5] surge como uma ferramenta de gerenciamento e compreensão de projetos escritos em Java, onde é possível organizar bibliotecas e publicá-las de forma gratuita para utilização pela comunidade de desenvolvimento. Podemos esperar também funcionalidades como atualizações, verificação de integridade e autoinstalação.

A utilização deste gerente é muito simples, tendo em vista que muitas suítes de desenvolvimento oferecem suporte nativo ao *Maven*. Após instalada, basta organizar as dependências (bibliotecas) do projeto em um arquivo especí-

fico denominado 'pom.xml', salvar e esperar que o próprio *Maven* configure as bibliotecas no projeto.

Figura 2 – Logo do gerente de projetos *Apache Maven*



Fonte: <<https://maven.apache.org>>

## 2.4 Apache POI

O *Apache POI* [6] também é uma biblioteca gratuita escrita utilizando a linguagem Java, que adiciona suporte de leitura e escrita de dados em documentos do *Microsoft Office* diretamente das aplicações Java, sem a necessidade de aquisição e instalação dos aplicativos da *Microsoft*.

Este complemento é desenvolvido e mantido pela *fundação Apache*, que é uma organização mundial sem fins lucrativos criada para dar suporte de desenvolvimento a projetos de programação de código aberto, ou seja, livres para todos.

Por ser uma fundação composta por uma comunidade descentralizada, suas soluções estão em constante melhoria, fornecendo ao desenvolvedor final peças de *software* com qualidade garantida.

Figura 3 – Logo da biblioteca *Apache POI*



Fonte: <<https://poi.apache.org>>

## 2.5 Eclipse IDE

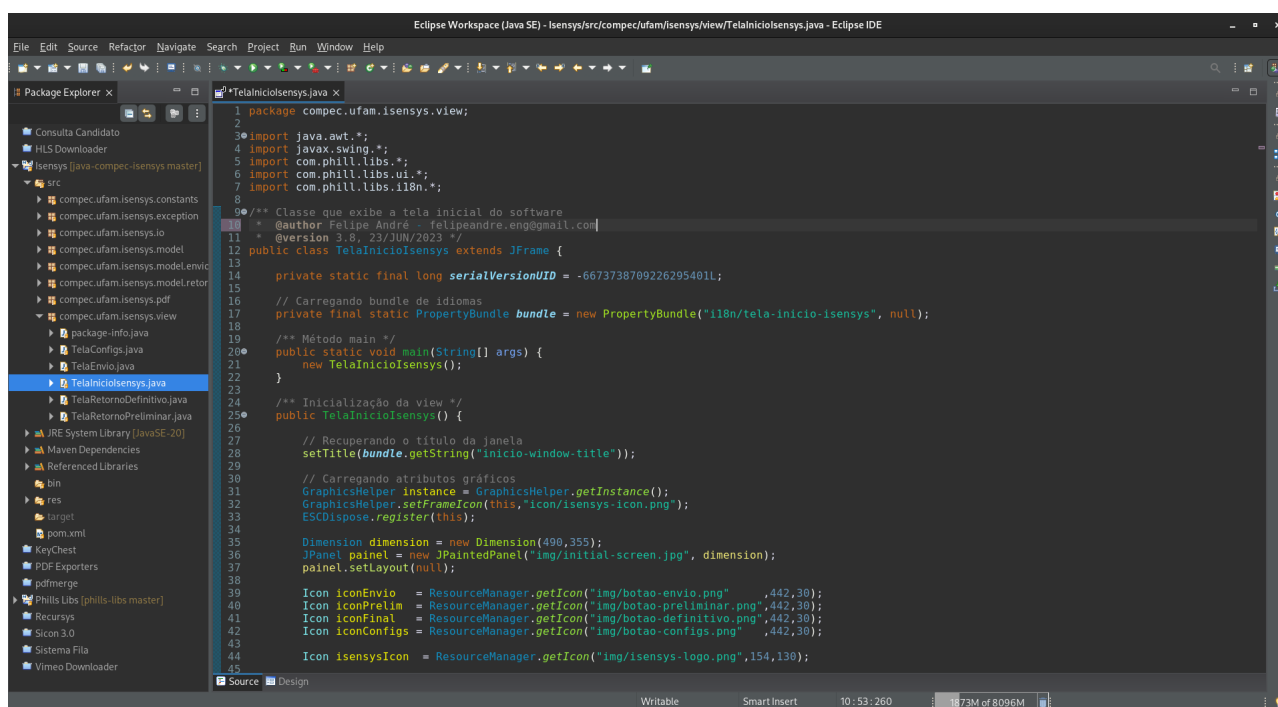
No mundo do desenvolvimento de software, agilidade e escalabilidade estão entre as qualidades mais requisitadas nas linguagens. Como forma de satisfazer tais demandas, existe no mercado uma vasta gama do que chamamos

de *IDE* - *integrated development environment* ou ambiente de desenvolvimento integrado, amplamente aceitos pela comunidade.

Inicialmente o projeto *Eclipse* [7] foi concebido pensando no desenvolvimento utilizando a linguagem Java, mas atualmente suporta extensões (*plugins*) que adicionam suporte a várias outras linguagens, tais como *PHP*, *Python*, *Arduino* etc e ainda por ser de código aberto, pode ser modificado para outras finalidades.

O aplicativo adiciona funcionalidades que auxiliam muito no desenvolvimento em uma linguagem, tais como autocomplemento de código, exibição de sugestões, dicas e tratamentos de erros, verificador de sintaxe e semântica e até integração com outras tecnologias como versionadores de código e o *Apache Maven* [5].

Figura 4 – *Eclipse IDE*



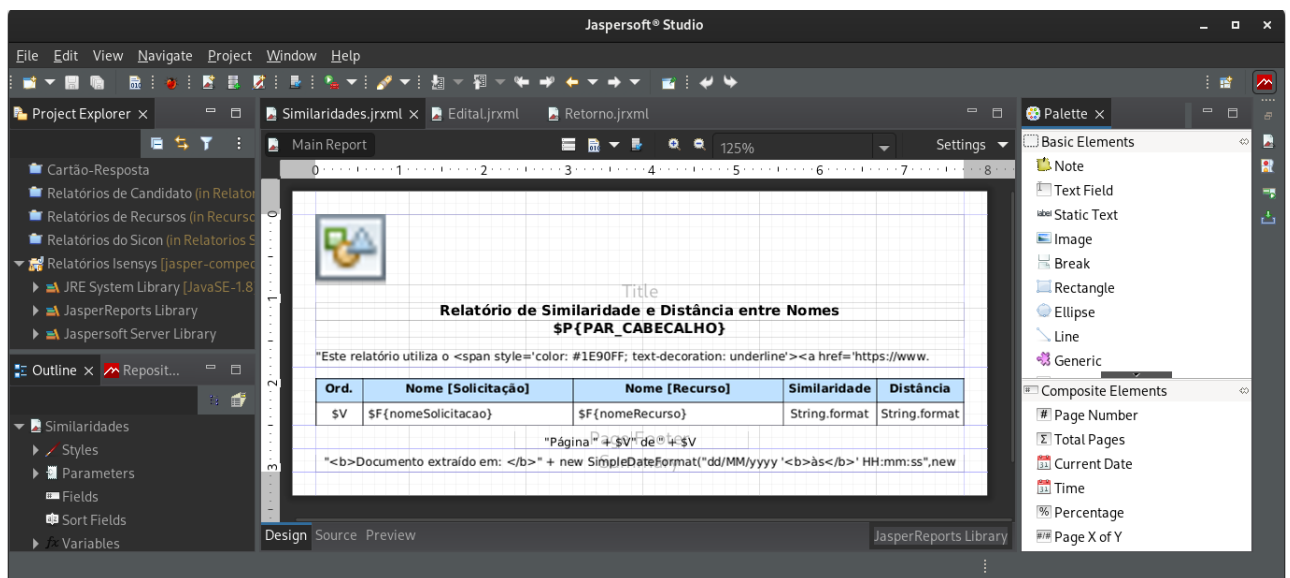
## 2.6 JasperReports®

A biblioteca *JasperReports*® [8] é gratuita e uma das mais robustas e populares utilizadas para criação de relatórios para aplicações Java. Possui sua própria suíte de design, onde o desenvolvedor pode rapidamente configurar

um novo relatório com poucos passos, seguindo guias e instruções diretamente na interface gráfica do editor.

Por ser desenvolvida utilizando a linguagem Java, ela também apresenta o mesmo comportamento e visualização em qualquer sistema operacional onde está disponível, produzindo relatórios de alta qualidade e escalabilidade, oferecendo opções de exportação para outros formatos conhecidos, tais como *PDF*, *Web* e *Microsoft Word*.

Figura 5 – Suíte de desenvolvimento *JasperReports®*



## 2.7 Jaro-Winkler

O algoritmo de *Jaro-Winkler* [9] é uma técnica de comparação de strings que mede a semelhança entre duas sequências de caracteres. Ele é frequentemente utilizado para encontrar correspondências aproximadas entre strings, especialmente em aplicações de deduplicação de dados e pesquisa de registros. A seguir, encontra-se um resumo da implementação do algoritmo.

Dados: Duas strings,  $s_1$  e  $s_2$

**Passo 1:** Encontre o comprimento das strings:

$$\text{len}_1 = \text{comprimento}(s_1)$$

$$\text{len}_2 = \text{comprimento}(s_2)$$

**Passo 2:** Defina o valor de  $m$  como o número de caracteres em comum (correspondentes) entre as duas strings, considerando uma janela de proximidade:

$$\text{Janela} = \lfloor \max(\text{len}_1, \text{len}_2)/2 \rfloor - 1$$

$m$  = número de caracteres em comum dentro da janela

**Passo 3:** Calcule o número de transposições:

$$t = \frac{1}{2} \times (\text{número de transposições de caracteres entre } s_1 \text{ e } s_2)$$

**Passo 4:** Calcule a medida de similaridade de Jaro:

$$\text{Jaro\_Sim} = \frac{m}{\text{len}_1} + \frac{m}{\text{len}_2} + \frac{m - t}{m}$$

**Passo 5:** Calcule o fator de ajuste de prefixo Winkler:

$$\text{Prefix\_Factor} = 0$$

$\text{Prefix\_Len}$  = número de caracteres idênticos no início das strings

Se  $\text{Prefix\_Len} > 4$ , limite  $\text{Prefix\_Len}$  a 4.

$$\text{Prefix\_Factor} = \text{Prefix\_Len} \times 0.1 \times (1 - \text{Jaro\_Sim})$$

**Passo 6:** Calcule a medida de similaridade de Jaro-Winkler:

$$\text{Jaro\_Winkler\_Sim} = \text{Jaro\_Sim} + \text{Prefix\_Factor}$$

**Passo 7:** A medida de similaridade final entre  $s_1$  e  $s_2$  é dada por  $\text{Jaro\_Winkler\_Sim}$ .

O valor resultante de  $\text{Jaro\_Winkler\_Sim}$  está dentro do intervalo  $[0, 1]$ , onde 0 indica nenhuma semelhança e 1 indica correspondência perfeita. Quanto maior o valor, maior a semelhança entre as duas strings.

## 2.8 Joda-Time

A biblioteca *Joda-Time* [10] facilita praticamente qualquer operação e manipulação com datas em um sistema Java, tarefa que é deveras complicada

utilizando apenas os recursos nativos da linguagem. Com uma boa documentação e as mais diversas classes para cálculos, formatação e operações com data, tem conquistado muitos desenvolvedores a utilizá-la em seus projetos.

Felizmente, *Joda-Time* é uma biblioteca livre para utilização pessoal e comercial, licenciada sobre a *Licença Apache 2.0*. Também é considerada consolidada e estável, estando até os dias atuais recebendo atualizações de otimização e novas funcionalidades.

## 2.9 Processo de Isenção

as



### 3 Desenvolvimento do *IsenSys*

Introduzidas as principais ferramentas utilizadas no projeto, aprofundar-se-á no processo de desenvolvimento do aplicativo *IsenSys*. Conceitos como especificações de sistema, requisitos para utilização e formas de aplicação serão demonstrados de forma objetiva.

O desenvolvimento do motor do sistema (*backend*) é regido pelas normas de formato de arquivo e diretivas definidas no documento de orientações gerais do SISTAC [11] e no manual de envio e recebimento de arquivos [2], versão 10.0, publicada em 24/06/2016.

Os requisitos funcionais (RF) do sistema são:

- O sistema deverá permitir a importação de dados de candidatos através de arquivos do tipo *csv* ou planilhas do *Microsoft Excel* (RF01);
- O sistema deverá permitir o cadastro e edição de dados do órgão gestor (RF02);
- O sistema deverá realizar testes de integridade nos dados de solicitações importadas (RF03);
- O sistema deverá gerar arquivos de importação para o SISTAC (RF04);
- O sistema só pode entregar um arquivo de importação com os dados de solicitações em sua plena completude e integridade (RF05);
- Dados de solicitações que foram informadas de forma incompleta ou inválida serão considerados erros e deverão ser exportados em uma planilha (RF06);
- O sistema deverá permitir a importação do arquivo de retorno do SISTAC (RF07);
- A partir do arquivo de retorno do SISTAC, o sistema deverá produzir editais de publicação e relatórios de resultados (RF08);

- O sistema também deve permitir a importação da planilha com erros, para inclusão nos editais públicos (RF09);
- O sistema deve gerar relatórios de estatísticas de solicitações deferidas e indeferidas (RF10);
- O sistema deverá gerar um relatório de similaridade entre nomes de candidatos recursantes (RF11).

Os requisitos não-funcionais (RF) do sistema são:

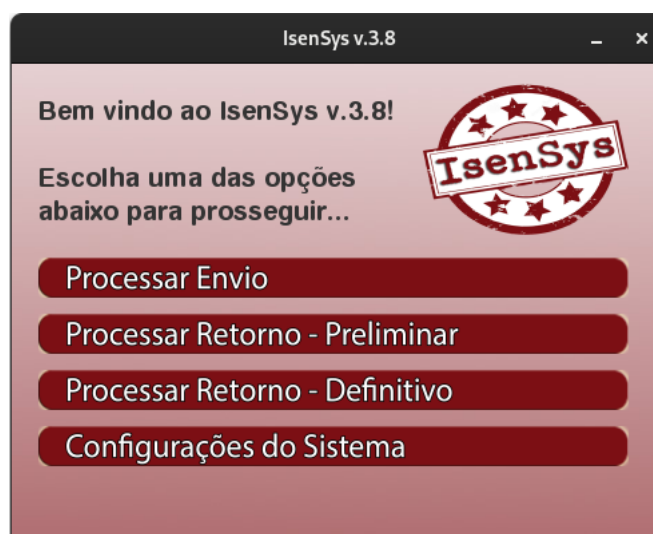
- A linguagem *Java Standard Edition (Java SE)* foi utilizada no desenvolvimento da aplicação (RNF01);
- A versão mínima da *Java Virtual Machine (JVM)* a ser utilizada é a 15 (RNF02);
- O projeto foi desenvolvido utilizando o *Java Development Kit (OpenJDK)* na versão 21 (RNF03);
- A suíte de interface gráfica utilizada é o *Java Swing* (RNF04);
- Por questões de simplicidade, todas as telas foram construídas utilizando layout absoluto com janela não redimensionável (RNF05);
- A versão do *Apache POI* utilizada é a 5.2.3, lançada em 17/09/2022 (RNF06);
- A versão do *JasperReports®* utilizada é a 6.26.0, lançada em 11/09/2023 (RNF07);
- O sistema pode ser executado em qualquer computador com suporte mínimo ao *Java Runtime Environment - JRE 15*, mínimo de 1GB de memória RAM disponível e monitor com resolução mínima de 800x600 (RNF08).

O sistema conta com três módulos distintos: um dedicado a preparar os dados de candidatos para envio ao SISTAC e os outros dois dedicados a processar os arquivos de retorno do SISTAC, produzir relatórios e editais de publicação preliminar e definitivo. As seções a seguir contém um estudo mais aprofundado sobre a implementação de cada módulo.

### 3.1 Tela Inicial

A tela inicial do *IsenSys* agrupa botões que dão acesso a todas as funcionalidades implementadas no sistema. Na primeira utilização é recomendável que o usuário configure os dados do órgão gestor, por meio do botão 'Configurações do Sistema'. A figura a seguir mostra a tela inicial do *IsenSys*.

Figura 6 – Tela Inicial do *IsenSys*



### 3.2 Tela de Configurações do Sistema

Algumas configurações são necessárias para o correto funcionamento do *IsenSys*. São elas:

- **CNPJ:** indica o número de CNPJ do órgão gestor;
- **Nome Fantasia:** indica o nome fantasia do órgão gestor, contendo no máximo 100 caracteres, segundo o manual do SISTAC [2];
- **Razão Social:** indica a razão social do órgão gestor, também contendo no máximo 100 caracteres, segundo o manual do SISTAC [2];
- **Índices da Planilha de Importação:** indica os índices dos campos de dados dispostos no arquivo de importação de solicitações. Tais índices começam em '0' e vão incrementando a cada coluna. Tomando como exemplo um arquivo de importação do *Microsoft Excel*, a coluna 'A', tem o

índice '0', a coluna 'C' tem o índice '2' e assim sucessivamente. Todos os índices devem ser preenchidos.

As configurações são salvas em um arquivo binário contido no diretório de recursos do sistema [3.6], sob o caminho '*config/program.dat*'.

Figura 7 – Tela de Configurações do Sistema

IsenSys v.3.8 - Configurações do Sistema

**Dados da Instituição**



CNPJ:

Nome Fantasia:

Razão Social:

**Índices da Planilha de Importação**

Nome	NIS	Dt. Nascimento	Sexo	RG	Data Emissão RG	Órgão Emissor RG	CPF	Nome da Mãe
2	3	4	5	6	7	8	9	10

Botões:  

### 3.3 Desenvolvimento do Módulo de Envio

Esta seção tem por objetivo detalhar o desenvolvimento do motor (*backend*) de preparação de dados de solicitações de isenção para envio ao SISTAC. Primeiramente precisamos entender como está distribuído o fluxo de informações pelo sistema. Para isto, serão detalhadas as entidades principais e auxiliares deste módulo.

#### 3.3.1 Modelagem de um Candidato

Os dados pessoais de um candidato são o objeto principal deste sistema, pois compõem uma solicitação de isenção que, após análise pelo SISTAC, tem uma resposta de deferimento ou não. Segundo o manual do SISTAC [2], os dados pessoais de candidatos necessários para o processamento estão dispostos na tabela a seguir.

De posse dos dados e tipos, foi concebida a modelagem da entidade *Candidato*, de acordo com o diagrama a seguir.

Tabela 1 – Dados pessoais de candidatos e seus formatos

<b>Campo</b>	<b>Descrição</b>	<b>Máximo de Caracteres</b>	<b>Tipo</b>	<b>Formato</b>
Nome	Nome completo do candidato sem caracteres especiais e sem abreviações	100	Texto	
NIS	Número de identificação social do candidato	11	Numérico	
Data de Nascim.	Data de nascimento do candidato	8	Numérico	ddmmaaaa
Sexo	Sexo do candidato	1	Texto	M ou F
RG	Número do Documento de Identidade do candidato	16	Alfanumérico	
Data de Emissão	Data de emissão do Documento de Identidade	8	Numérico	ddmmaaaa
Sigla RG	Sigla do órgão emissor do Documento de Identidade	30	Alfanumérico	
CPF	Número do CPF do candidato	11	Numérico	
Nome da Mãe	Nome completo da mãe do candidato, sem caracteres especiais e sem abreviações	100	Texto	

Figura 8 – Modelagem da entidade *Candidato*

<b>Candidato</b>
+nome: String +nis: String +dataNascimento: DateTime +sexo: char +rg: String +dataEmissaoRG: DateTime +orgaoEmissorRG: String +cpf: String +nomeMae: String

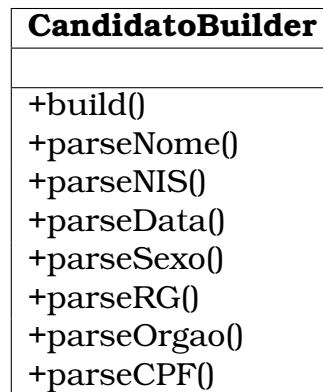
De acordo com o diagrama de classe da entidade *Candidato*, nota-se que esta apenas armazena dados. As validações nos campos são garantidas por uma classe intermediária, detalhada na seção seguinte.

### 3.3.2 A classe *CandidatoBuilder*

Esta classe é responsável por montar um *Candidato* com os dados extraídos de um arquivo de entrada. Durante esse processo ela realiza uma

série de validações nos dados, e caso haja pelo menos uma inconsistência, uma exceção com detalhes desta inconsistência é lançada, caso contrário, significa que foi possível construir um objeto respeitando todos os requisitos de formatos do SISTAC [2]. Segue o diagrama de classe do *CandidatoBuilder*.

Figura 9 – Diagrama de Classe de *CandidatoBuilder*



Os métodos *parse...()*, responsáveis por realizar validações específicas em cada campo de dados de um *Colaborador* e lançam exceções detalhadas para que, posteriormente, tanto o órgão gestor dos dados quanto o candidato possam ter conhecimento de quais campos enviaram fora de formato e se ainda cabe algum recurso.

Para registrar inconsistências em cada um dos campos de dados de um *Candidato*, é utilizada a classe de exceção *FieldParseException* que, por sua vez, é incorporada à classe de exceção *RowParseException*, montada com todas as exceções percebidas pelo método *build()*.

### 3.3.3 A classe de exceção *FieldParseException*

Como forma de descentralizar as tratativas de validação de dados de candidatos, a classe de exceção *FieldParseException* é responsável por armazenar informações sobre o motivo de um campo não ter sido validado e qual campo gerou esta exceção. Esta classe apenas estende a superclasse *Exception* e monta uma *String* formatada com o motivo e o nome do campo inválido.

### 3.3.4 A classe de exceção *RowParseException*

Com o objetivo de concentrar todas as exceções de validação dos campos de dados de um *Candidato*, a classe de exceção *RowParseException* armazena tais exceções em uma lista encadeada e ainda adiciona informações que ajudam a identificar qual foi o candidato que gerou a(s) exceção(ões) e ainda em qual posição do arquivo de entrada ele está.

A seguir é possível visualizar o diagrama de classe de *RowParseException*.

Figura 10 – Diagrama de Classe de *RowParseException*

<b>RowParseException</b>
+linha: int +nis: String +cpf: String +nome: String +listaExcecoes: List<FieldParseException>
+addException() +hasException() +getMessage() +getErrorSummaryArray() +getErrorSummaryString()

### 3.3.5 A classe *ParseResult*

Esta classe é responsável por concentrar o resultado da extração de dados do(s) arquivo(s) de entrada em duas listas:

1. Lista de *Candidato*: onde são armazenados apenas dados de candidatos solicitantes de isenção que passaram com sucesso por todas as validações de campos;
2. Lista de *RowParseException*: onde são armazenados os dados de solicitações julgados inválidos pela classe *CandidatoBuilder* [3.3.2].

A seguir podemos compreender melhor a classe por meio de seu diagrama.

Figura 11 – Diagrama de Classe de *ParseResult*

<b>ParseResult</b>
+listaCandidatos: List<Candidato> +listaExcecoes: List<RowParseException>
+addCandidato() +addExcecao() +getListaCandidatos() +getListaExcecoes() +sortLists()

### 3.3.6 Importadores de Dados de Candidato

O *IsenSys* é capaz de importar dados pessoais de candidatos solicitantes de isenção em dois formatos:

1. **Arquivo (.csv):** que é um arquivo de texto puro (sem formatação), contendo um cabeçalho na sua primeira linha e os dados pessoais requeridos pelo sistema nas outras linhas. O arquivo deve estar codificado em UTF-8, com separador por tabulação, vírgula ou ponto-e-vírgula.
2. **Planilha do Microsoft Excel (.xlsx):** a planilha deve conter um cabeçalho na primeira linha com os nomes dos campos e nas demais linhas os dados dos candidatos solicitantes de isenção.

Nos dois casos a ordem da disposição dos dados é extremamente importante para a correta importação. Tanto as colunas do arquivo *csv* quanto as da planilha do *Microsoft Excel* devem respeitar a seguinte ordem:

1. Nome completo;
2. NIS;
3. Data de nascimento;
4. Sexo;
5. Número de RG;
6. Data de emissão do RG;
7. Órgão emissor do RG;



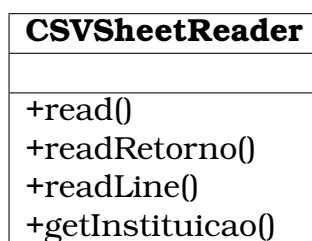
8. CPF;
9. Nome completo da mãe.

Para cada tipo de arquivo foi implementado um importador, contendo as especificidades de tratamento de cada formato. Os dados extraídos pelos importadores são então enviados ao *CandidatoBuilder* que irá construir um objeto *Candidato*, tornando todo o processo de tratativa de arquivos transparente às classes superiores.

#### 3.3.6.1 O importador *CSVSheetReader*

Para o arquivo no formato *.csv*, temos o importador descrito na classe *CSVSheetReader*, que é útil tanto para o módulo de envio, através do método *read()*, quanto pelo módulo de retorno, através do método *readRetorno()*. A princípio o papel deste importador é detectar o tipo de separador do arquivo *.csv* e extrair os dados linha-a-linha. A seguir temos seu diagrama de classe.

Figura 12 – Diagrama de Classe de *CSVSheetReader*



#### 3.3.6.2 O importador *ExcelSheetReader*

Este importador também é comum aos três módulos do sistema, mas em momentos distintos. Sua função no módulo de envio é iterar sobre as linhas e colunas de uma planilha com os dados de candidatos solicitantes de isenção e entregá-los ao *CandidatoBuilder*. Seu diagrama de classe está disposto na figura a seguir.

### 3.3.7 Exportadores de Dados

O estado final do processo de importação de dados dos candidatos solicitantes consiste na concentração deles na classe *ParseResult* [3.3.5]. A partir

Figura 13 – Diagrama de Classe de *ExcelSheetReader*

<b>ExcelSheetReader</b>
+read() +readErros() +readLine() +getCellContent()

daqui, os dados dos candidatos considerados válidos pela classe *Candidato-Builder* estão prontos para serem exportados para o arquivo no formato de envio do SISTAC [2]. Os dados de solicitações inválidas são exportados para uma planilha do *Microsoft Excel*, para futuro processamento nos módulos de retorno.

#### 3.3.7.1 O exportador *CSVSheetWriter*

Esta classe tem como função exportar os dados de candidatos válidos para o(s) arquivo(s) no formato de envio do SISTAC [2]. Na primeira linha do arquivo é impresso o cabeçalho com alguns dados do órgão gestor (configurados previamente) e nas demais, os dados são dispostos de acordo com a formatação exigida no manual.

Empiricamente foi descoberto que o SISTAC tem um limite máximo de solicitações por arquivo de envio definido em 2000. Portanto, o exportador automaticamente gera outros arquivos na sequência caso a quantidade de solicitações ultrapasse este limite.

Figura 14 – Diagrama de Classe de *CSVSheetWriter*

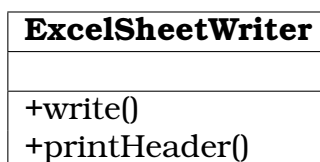
<b>CSVSheetWriter</b>
+write() +getSistacFilename()

#### 3.3.7.2 O exportador *ExcelSheetWriter*

Esta classe tem a função de exportar os dados de solicitações inválidas para uma planilha do *Microsoft Excel*, onde a primeira linha contém um ca-

beçalho com títulos das colunas de erros e nas demais linhas, os dados de identificação do candidato e a lista de campos que foram considerados inválidos pela classe *CandidatoBuilder*.

Figura 15 – Diagrama de Classe de *ExcelSheetWriter*



### 3.3.8 Integração do Módulo de Envio

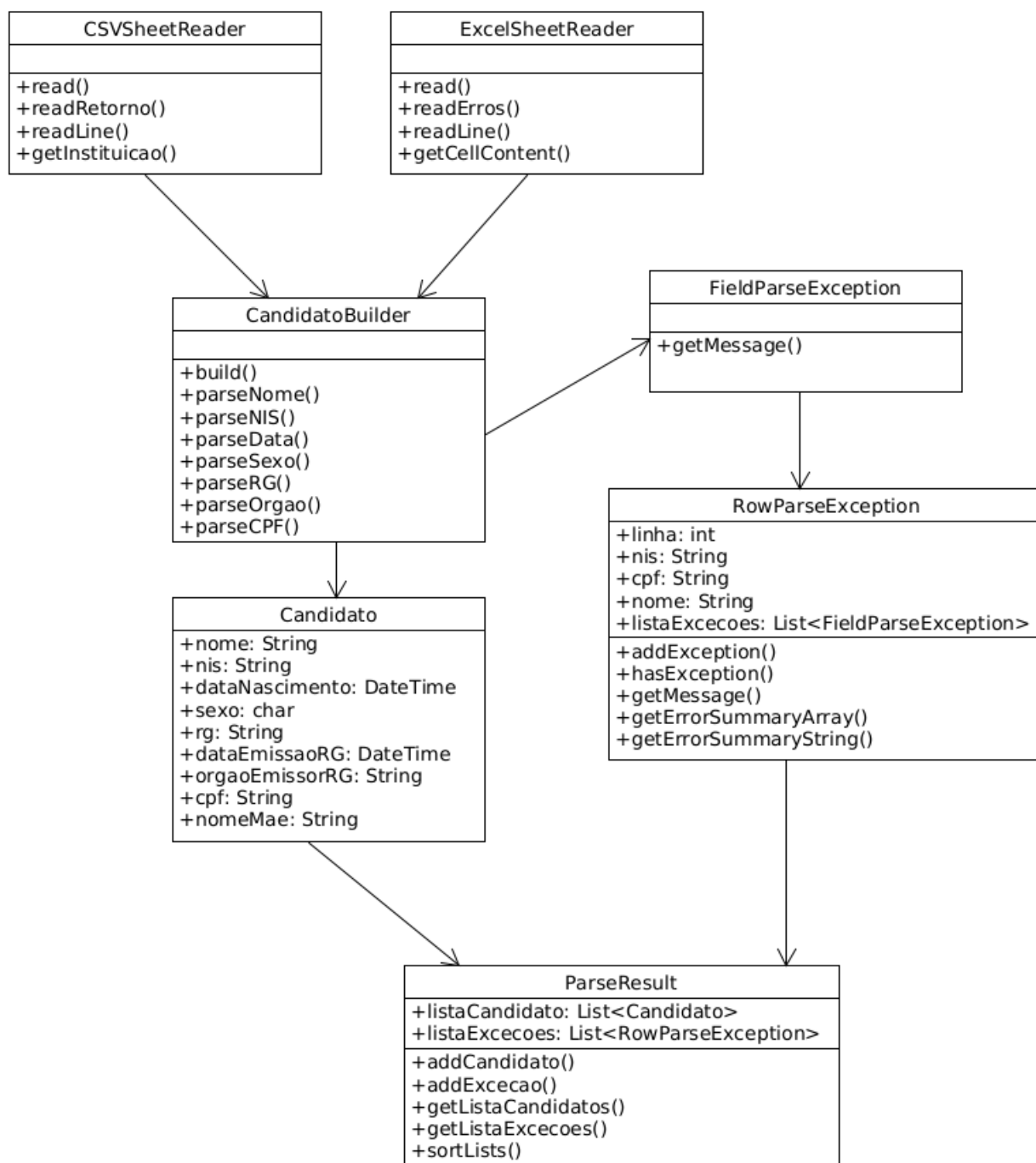
Após conhecer individualmente todos os agentes envolvidos no módulo de envio, far-se-á uma análise em conjunto de todos eles, possibilitando compreender o fluxo das informações e os estágios do processo de preparação dos dados das solicitações para envio ao SISTAC. A seguir temos uma visão geral do fluxo dos dados de uma solicitação de isenção, desde o arquivo de origem até o estágio onde todos estão devidamente processados.

Em resumo, os dados dos candidatos solicitantes são extraídos de acordo com o tipo de arquivo de origem, pelas classes *CSVSheetReader* e *ExcelSheetReader* que, por sua vez, chamam a classe *CandidatoBuilder* para validar os dados extraídos e, caso sejam 100% válidos, um novo objeto *Candidato* é gerado, do contrário, uma exceção do tipo *RowParseException* é lançada com informações de todos os campos inválidos (por meio da classe *FieldParseException*). Por fim, os dados são concentrados no objeto *ParseResult* e estão prontos para exportação.

### 3.3.9 Interface Gráfica do Módulo de Envio

Como premissas no desenvolvimento do *frontend* do *IsenSys*, simplicidade e usabilidade sobressaem-se, mas sem perder a essência de uma aplicação robusta e consolidada. Na figura a seguir, está ilustrada a única tela do módulo de envio.

Figura 16 – Diagrama do Fluxo de Dados das Solicitações



No painel 'Dados da Instituição', estão dispostas algumas informações exigidas no cabeçalho do arquivo de envio ao SISTAC, tais como CNPJ, nome fantasia e razão social do órgão gestor.

No painel 'Arquivo de Entrada' é possível selecionar o arquivo de origem dos dados de solicitações dos candidatos, atualmente planilhas do *Microsoft Excel* e arquivos *csv* são suportados. Também é possível recarregar o arquivo

Figura 17 – Tela do Módulo de Envio

The screenshot shows the 'IsenSys v.3.8 - Exportação' window. It has a dark title bar with standard window controls. The main area is divided into sections. The 'Dados da Instituição' section contains three labels: 'CNPJ: 01.234.567/0000-04', 'Nome Fantasia: NOME FANTASIA DO ORGAO GESTOR', and 'Razão Social: RAZAO SOCIAL DO ORGAO GESTOR'. The 'Arquivo de Entrada' section has a 'Nome:' label followed by an empty text box and three icons: a refresh icon, a paintbrush icon, and a magnifying glass icon. The 'Arquivos de Saída' section has two rows of controls. The first row has 'Num. do Edital:' followed by an empty text box and 'Sequência:' followed by a text box containing '1' and a spinner control. The second row has 'Pasta de Saída:' followed by an empty text box and two icons: a magnifying glass icon and a paintbrush icon. A save icon is located at the bottom right of the window.

previamente selecionado ou limpar sua seleção. Se um arquivo válido foi selecionado, algumas informações sobre o carregamento são mostradas na porção inferior deste painel, como ilustra a figura a seguir.

Figura 18 – Tela do Módulo de Envio (arquivo carregado)

This screenshot shows the same 'IsenSys v.3.8 - Exportação' window as Figure 17, but with data entered. In the 'Arquivo de Entrada' section, the 'Nome:' text box now contains 'solicitacoes.csv'. Below this text box, there is a summary line: 'Solicitações: 7651 (OK) 1 (ERRO) 7652 (TOTAL)'. The 'Arquivos de Saída' section remains the same as in Figure 17, with empty text boxes for 'Num. do Edital' and 'Pasta de Saída', and the 'Sequência' set to '1'.

Os dois primeiros campos do painel 'Arquivos de Saída' (Num. do Edital e Sequência) também são exigências descritas no manual de envio do SISTAC [2].

O campo 'Num. do Edital' faz parte do nome do arquivo de envio, juntamente com a identificação de sua sequência. Cada arquivo possui uma sequência geralmente iniciada em '1'. Se dois arquivos são gerados, temos então as sequências '1' e '2'. Sequências são reiniciadas a cada novo dia. Ainda neste painel, podemos escolher o diretório de escrita dos arquivos ou limpar sua seleção.

Por fim, após todos os dados serem fornecidos na tela, a exportação pode ser realizada clicando no ícone de salvar (disquete). As figuras a seguir ilustram a saída dos arquivos após alimentar a tela com algumas informações.

Figura 19 – Tela do Módulo de Envio (preenchida) + Arquivos de Saída



De posse dos arquivos exportados, o envio já pode ser realizado à plataforma do SISTAC. A planilha gerada pode ser consultada de forma avulsa, mas é recomendável mantê-la inalterada, pois é necessária para a montagem dos editais de publicação nos módulos de retorno.

### 3.4 Desenvolvimento dos Módulos de Retorno

Esta seção tem por objetivo detalhar o desenvolvimento do *backend* de processamento do(s) arquivo(s) de retorno do SISTAC. A partir dele(s) é possível gerar relatórios com algumas métricas e estatísticas úteis ao órgão gestor, bem como os editais de publicação.

Utilizando a mesma metodologia abordada na seção anterior, vamos começar compreendendo o fluxo de informações, conhecendo as classes envol-

vidas no processo e, posteriormente, a interface gráfica.

### 3.4.1 Modelagem de um Retorno

Para estes módulos, apenas alguns dados das solicitações dos candidatos são aproveitados. Temos então a concepção da classe *Retorno*, de acordo com a especificação descrita na figura a seguir:

Figura 20 – Modelagem da Entidade *Retorno*

<b>Retorno</b>
+situacao: char
+nome: String
+nis: String
+cpf: String
+motivo: int
+nomeAnterior: String
+deferre()
+isDeferido()
+compareTo()

### 3.4.2 A classe *ListaRetornos*

Esta classe concentra todos os *Retorno*'s extraídos do arquivo de retorno do SISTAC, com adição dos atributos do órgão gestor e dados do edital de publicação. Será detalhado ao decorrer desta monografia os dois módulos de retorno, que são: retorno preliminar e retorno definitivo. Por enquanto, temos que esta classe armazena dados úteis aos dois módulos.

### 3.4.3 A classe *Compilation*

Esta classe possui a incumbência de gravar ou carregar em arquivo um objeto da classe *ListaRetornos*, útil no processamento do retorno definitivo. Eis o seu diagrama de classe:

### 3.4.4 Modelagem de uma Situacao

De acordo com o manual do SISTAC [2], existem algumas situações de indeferimento definidas. No *IsenSys*, tais situações são armazenadas em um

Figura 21 – Diagrama de classe de *ListaRetornos*

<b>ListaRetornos</b>
+listaRetornos: List<Retorno> +cnpj: String +nomeFantasia: String +razaoSocial: String +edital: String +dataEdital: String +cabecalho: String
+get() +size() +getList() +clone() +add() +update() +sort()

Figura 22 – Diagrama de Classe de *Compilation*

<b>Compilation</b>
+save() +load()

arquivo (*csv*) com codificação UTF-8 no diretório de recursos do sistema 3.6. Durante a confecção dos editais de publicação, as situações são carregadas do arquivo e enviadas à classe geradora de relatórios, juntamente com os dados dos retornos.

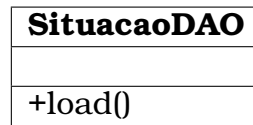
Figura 23 – Modelagem da Entidade *Situacao*

<b>Situacao</b>
+id: String +motivo: String +descricao: String

### 3.4.5 A classe *SituacaoDAO*

Basicamente esta classe é responsável por carregar as situações a partir do arquivo '*situacoes.csv*', contido no diretório de recursos do *IsenSys*, para uma lista de *Situacao*, útil na confecção dos editais de publicação.

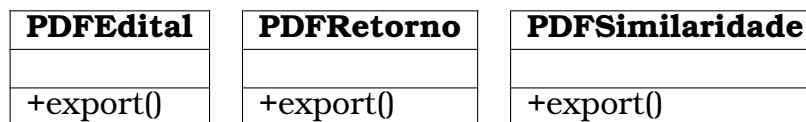


Figura 24 – Diagrama de Classe de *SituacaoDAO*

### 3.4.6 Geradores de Relatório

O *IsenSys* implementa três tipos de relatórios a partir dos dados provenientes do(s) arquivo(s) de retorno do SISTAC. Cada uma das classes que os implementam contam com o método '*export()*', que é capaz de compilar e exportar o relatório para o formato PDF. A figura a seguir ilustra as estruturas de classe dos três geradores.

Figura 25 – Diagramas de Classe dos Geradores de Relatório



### 3.4.7 Integração dos Módulos de Retorno

Conhecendo individualmente os agentes envolvidos no módulo de retorno, podemos realizar uma abordagem integrada dos dois tipos de retorno: retorno preliminar e retorno definitivo.

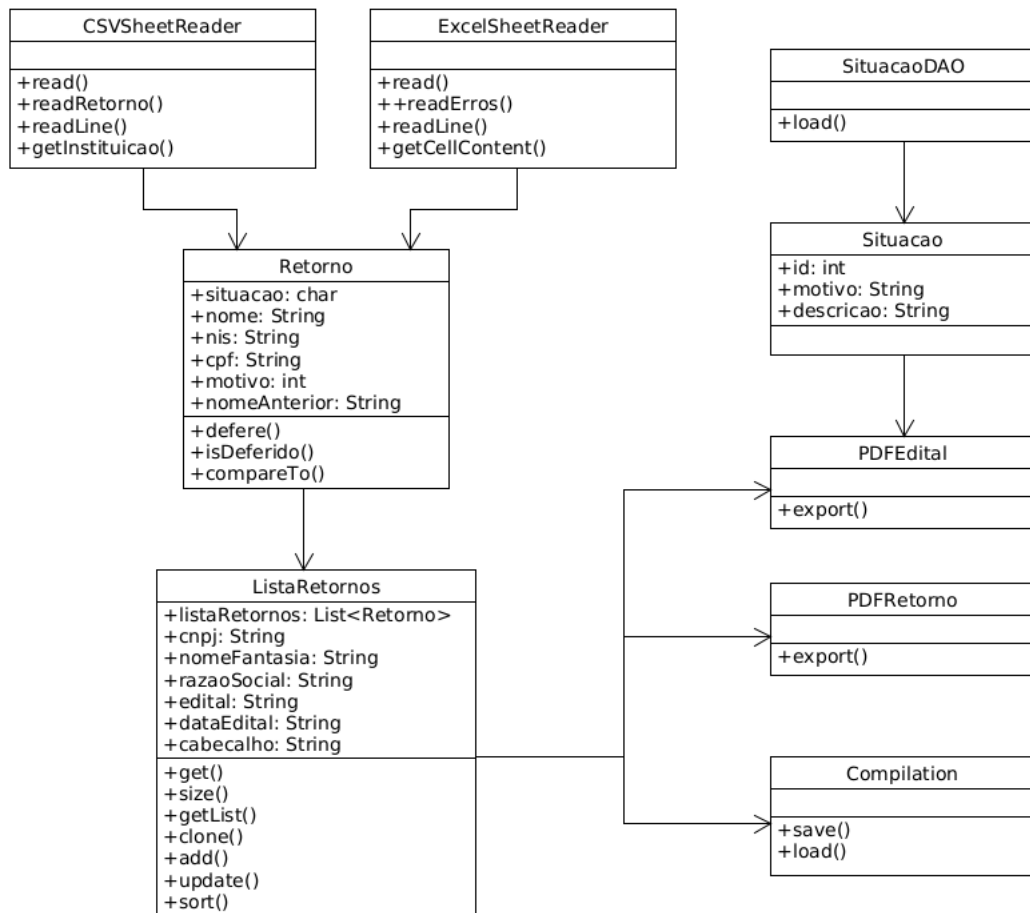
#### 3.4.7.1 Retorno Preliminar

Este é o módulo destinado ao processamento do resultado preliminar das solicitações de isenção. A figura a seguir ilustra o fluxo das informações através do módulo.

Começando pelos arquivos de entrada, as classes *CSVSheetReader* e *ExcelSheetReader* carregam o(s) arquivo(s) de retorno do SISTAC e planilha de erros, respectivamente, produzindo então objetos da classe *Retorno* que são agrupados em uma lista na classe *ListaRetornos*.

Na exportação, os dados armazenados na classe *ListaRetornos* são enviados aos geradores de PDF implementados em *PDFEdital* e *PDFRetorno* e,

Figura 26 – Diagrama do Fluxo de Dados no Retorno Preliminar



por fim, salvos em um arquivo binário denominado de 'compilação', através da classe *Compilation*.

#### 3.4.7.2 Tela do Módulo de Retorno Preliminar

A figura a seguir ilustra a implementação da interface gráfica do módulo de retorno preliminar do *IsenSys*.

Figura 27 – Tela do Módulo de Retorno Preliminar

IsenSys v.3.8 - Resultado Preliminar



**Dados da Instituição**



CNPJ: 01.234.567/0000-04

Nome Fantasia: NOME FANTASIA DO ORGAO GESTOR


Razão Social: RAZAO SOCIAL DO ORGAO GESTOR

**Arquivos de Entrada**



Retorno Sistac:   


Planilha Erros:   

**Edital**

Cabeçalho:  

**Arquivos de Saída**

Diretório:   



No painel 'Dados da Instituição', estão dispostas algumas informações exigidas no cabeçalho do arquivo de envio ao SISTAC [2], tais como CNPJ, nome fantasia e razão social do órgão gestor.

Em 'Arquivos de Entrada' é possível selecionar o(s) arquivo(s) de retorno do SISTAC e a planilha contendo os erros de validação (gerada no módulo de envio). O usuário também é capaz de limpar a seleção do(s) arquivo(s). A seguir estão enumerados alguns detalhes sobre os arquivos:

- **Retorno Sistac:** o usuário deve selecionar SEMPRE o primeiro arquivo de retorno do SISTAC, caso haja mais de um, o carregamento dos demais arquivos é feito de forma automática, basta que estejam no mesmo diretório do primeiro arquivo.
- **Planilha Erros:** aqui o usuário deve selecionar a planilha de erros gerada pelo módulo de envio do *IsenSys*.

Se um arquivo válido foi selecionado, algumas informações sobre o carregamento são mostradas na porção inferior deste painel, como ilustra a figura a seguir.

Figura 28 – Tela do Módulo de Retorno Preliminar (preenchida)

IsenSys v.3.8 - Resultado Preliminar

**Dados da Instituição**

CNPJ: 01.234.567/0000-04

Nome Fantasia: NOME FANTASIA DO ORGAO GESTOR

Razão Social: RAZAO SOCIAL DO ORGAO GESTOR

**Arquivos de Entrada**

Retorno Sistac: RETORNO\_01234567000004\_012023\_01082023\_005.txt

Planilha Erros: ERROS\_01234567000004\_012023\_01082023.xlsx

**Análise do Arquivo**

Deferidos: 2836      Indeferidos: 4816      Total: 7652

**Edital**

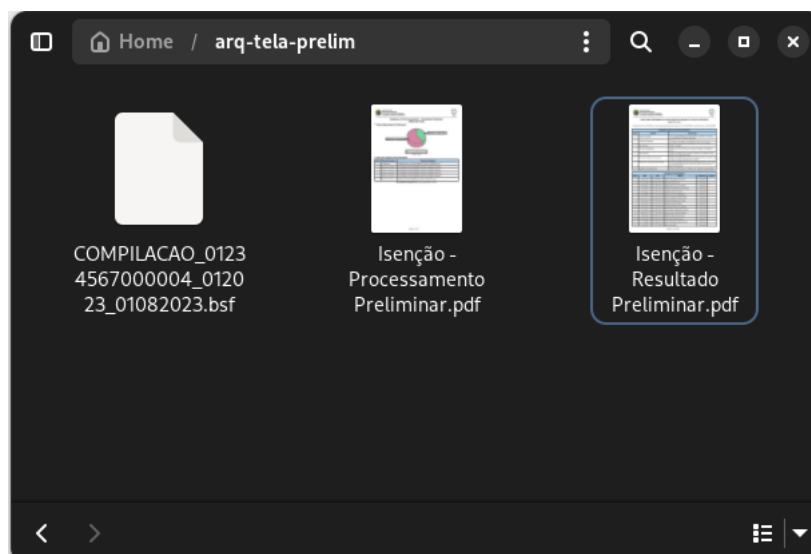
Cabeçalho: Edital de Teste

**Arquivos de Saída**

Diretório: /home/felipe/arq-tela-prelim

O nome do edital pode ser informado no campo de texto do painel 'Edital' e no painel 'Arquivos de Saída' é possível selecionar o diretório para saída dos arquivos gerados pelo módulo (edital de publicação, relatório de estatísticas e arquivo de compilação), como mostra a figura a seguir.

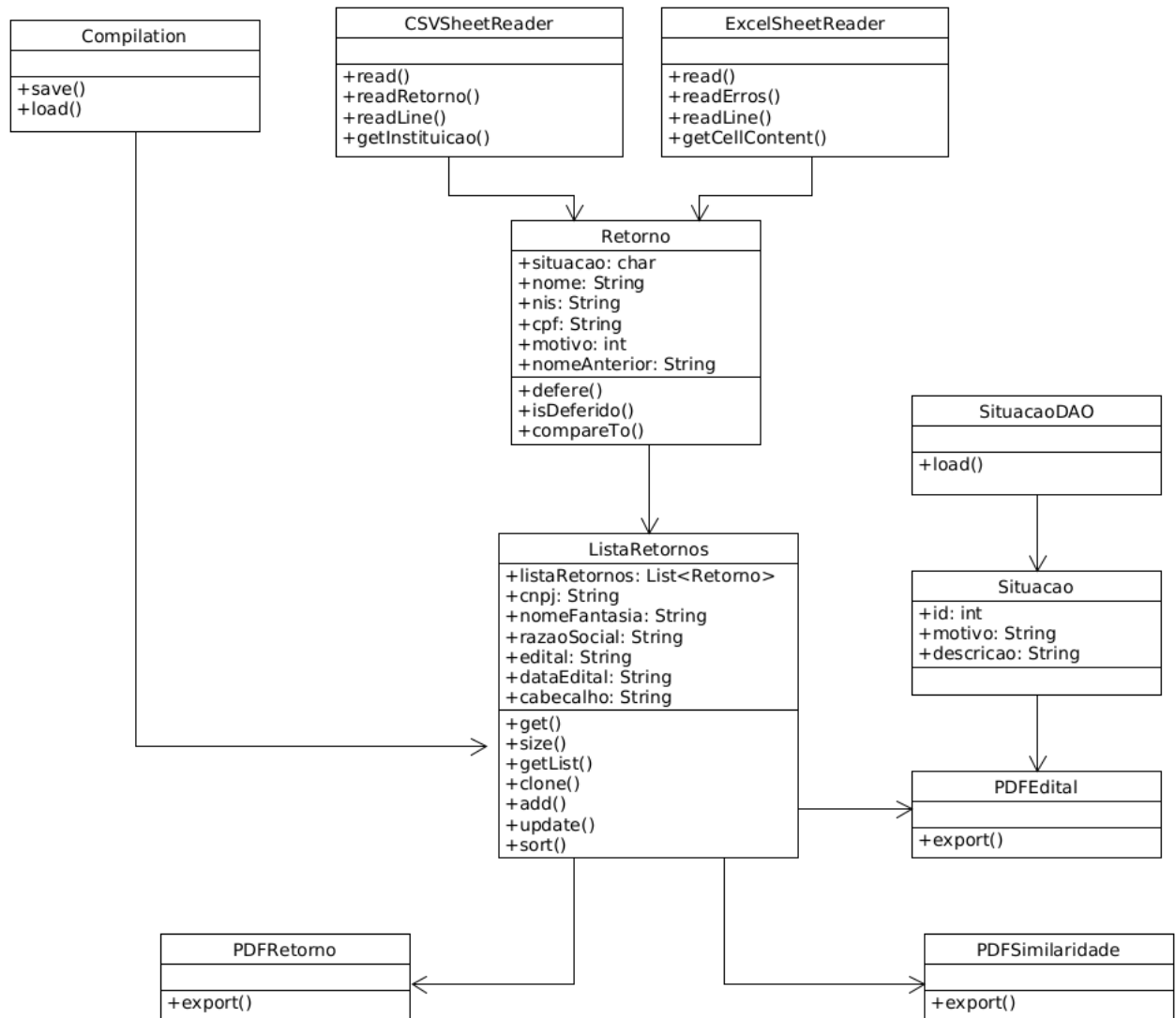
Figura 29 – Arquivos de Saída do Módulo de Retorno Preliminar



#### 3.4.7.3 Retorno Definitivo

Este é o módulo destinado ao processamento do resultado definitivo. É capaz de recuperar as informações do módulo de retorno preliminar, através de um arquivo de compilação (gerado por *Compilation* [3.4.3]) e importar o(s) novo(s) arquivo(s) de retorno do SISTAC.

Figura 30 – Diagrama do Fluxo de Dados no Retorno Definitivo



Para o processamento do retorno definitivo, a principal classe alimentadora é a *Compilation* que carrega os dados provenientes do módulo de resultado preliminar e, a princípio, já estão prontos para serem exportados como resultado definitivo.

#### 3.4.7.4 Tela do Módulo de Retorno Definitivo

A figura a seguir ilustra a implementação da interface gráfica do módulo de retorno definitivo do *IsenSys*.

Figura 31 – Tela do Módulo de Retorno Definitivo

IsenSys v.3.8 - Resultado Definitivo

**Dados da Instituição**

CNPJ: 01.234.567/0000-04

Nome Fantasia: NOME FANTASIA DO ORGAO GESTOR

Razão Social: RAZAO SOCIAL DO ORGAO GESTOR

**Resultado Preliminar**

Compilação:   

**Arquivos de Entrada**

Retorno Sistac:  

Planilha Erros:  

**Edital**

Cabeçalho:  

**Arquivos de Saída**

Diretório:   



Ao carregar o arquivo de compilação, algumas informações são exibidas na porção inferior do painel 'Resultado Preliminar'. O cabeçalho do edital também é carregado.

Da mesma forma que no módulo do resultado preliminar, o arquivo de retorno a ser selecionado deve ser sempre o primeiro, os demais são carregados automaticamente desde que estejam no mesmo diretório do primeiro arquivo.

Se algum arquivo de entrada for selecionado, tanto de retorno quanto de erros, os dados são mesclados com os provenientes do resultado preliminar da seguinte forma:

- **Retorno Sistac:** se o candidato retornado tiver sua solicitação **indeferida**, apenas seu status é atualizado na lista preexistente. Caso tenha sido **deferida**, além de ter seu estado alterado, o novo deferido fará parte de uma nova lista, para futuro cálculo de similaridade 3.5;

- **Erros:** aqui todos os erros são mesclados.

Após a seleção do diretório de saída o sistema está pronto para exportar os relatórios definitivos.

## 3.5 Relatório de Similaridade

Uma necessidade especial tornou necessária a implementação de um relatório que calculasse o quão similar é o nome de um candidato na fase de solicitação na fase de recurso.

Empiricamente foi detectado que um candidato qualquer que teve sua solicitação de isenção indeferida em primeira instância e recursou trocando seus dados pessoais pelos de alguém que tem ciência de isenção garantida, teve recurso deferido pelo SISTAC.

Como forma de evitar que tais candidatos praticantes deste ato ilícito tivessem isenção deferida sem ciência do órgão gestor, foi desenvolvido um relatório chamado de 'Relatório de Similaridade e Distância entre Nomes', que computa porcentagens das referidas métricas utilizando o algoritmo de *Jaro-Winkler*, entre o nome informado na fase de solicitação e recurso, pelo candidato.

O algoritmo é executado tomando como base candidatos que tiveram solicitação de isenção indeferida em primeira instância, mas deferida após recurso, gerando um relatório com o comparativo entre nomes e as duas métricas (similaridade e distância), ordenado pela ordem crescente de similaridade.

Algumas classes estão envolvidas na confecção deste relatório, são elas: *Similaridade* e *JaroWinkler*, além do gerador de PDF: *PDFSimilaridade*.

### 3.5.1 Modelagem de uma Similaridade

Esta classe contém alguns atributos úteis para confecção do relatório de similaridade entre nomes. A figura a seguir ilustra seu diagrama de classe.



Figura 32 – Diagrama de Classe de uma Similaridade

<b>Similaridade</b>
+nomeSolicitacao: String
+nomeRecurso: String
+distancia: double
+similaridade: double

### 3.5.2 A classe *JaroWinkler*

É uma classe auxiliar que computa o algoritmo de *Jaro-Winkler* [9] nos dados dos candidatos recursantes. Para cálculo das métricas de distância e similaridade, a biblioteca *Apache Commons Text*<sup>TM</sup> [2.2] foi utilizada. A seguir seu diagrama de classe.

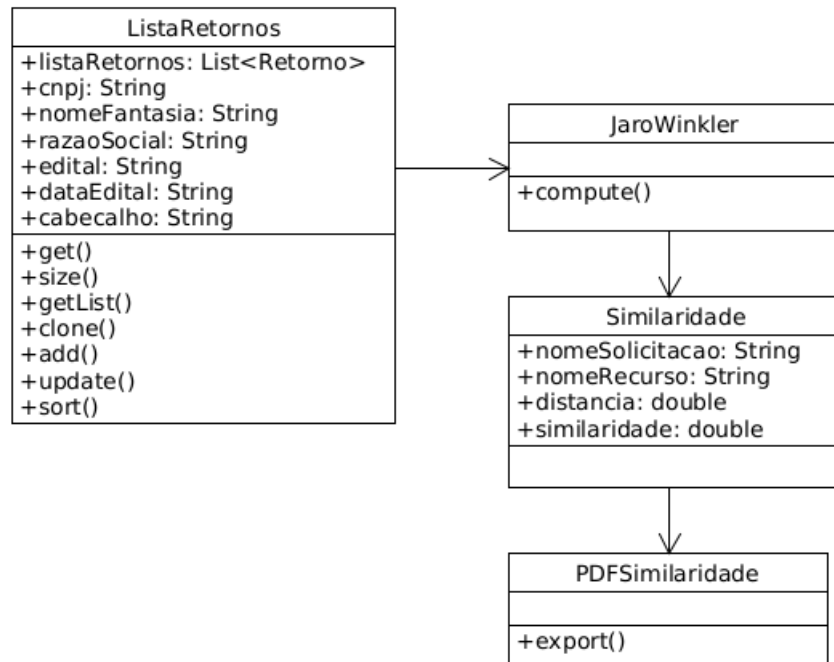
Tabela 2 – Diagrama de Classe de *JaroWinkler*

<b>JaroWinkler</b>
+compute()

### 3.5.3 Montagem do Relatório de Similaridade

Realizando uma abordagem integrada, conhecendo os agentes envolvidos na produção do relatório, temos o seguinte diagrama.

Figura 33 – Diagrama do Fluxo de Dados no Relatório



De uma forma prática e funcional, os dados dos candidatos solicitantes, provenientes de *ListaRetornos*, são encaminhados à classe *JaroWinkler* para computação das métricas de distância e similaridade e salvamento em objetos da classe *Similaridade*. Por fim, *PDFSimilaridade* aproveita estes dados para a confecção do relatório.

### 3.6 Diretório de Recursos do *IsenSys*

O *IsenSys* conta com alguns arquivos de configuração e recursos que estão dispostos em um diretório chamado 'res', existente no mesmo nível de diretório do arquivo executável do *IsenSys*.

A seguir estão listados os recursos e seus caminhos relativos ao diretório raiz da aplicação.

- **config/program.dat:** contém as configurações globais do sistema;
- **fonts/fonts-extension.jar:** fontes utilizadas na confecção dos relatórios;
- **i18n/\*.lng:** arquivos contendo as strings de interface do sistema;
- **icon/\*.png:** contém os ícones utilizados nas telas;

- **img/**: diretório contendo as imagens de fundo das telas;
- **relatorios/\*.jasper**: arquivos dos relatórios;
- **situacoes/situacoes.csv**: arquivo contendo as situações de indeferimento.

Para o correto funcionamento do sistema é recomendável que o usuário não altere o conteúdo destes arquivos manualmente, tampouco os exclua, renomeie ou os troque de diretório.

## 4 Implementação e Resultados

A versão corrente do sistema (v.3.8) foi implementada no início de Junho de 2023 na COMPEC, com o objetivo de ter como piloto o processamento das solicitações de isenção do Processo Seletivo Contínuo - PSC 2024 - Etapas 1 e 2, fruto dos Editais de nº 13 [12] e 14/GR/UFAM [13].

Por pura coincidência, o PSC 2024 é a primeira edição a possuir a modalidade de isenção via CadÚnico logo nas primeiras etapas do processo, opção que só era concedida aos candidatos da 3ª etapa.

Segundo os editais que regem o processo seletivo, o período de solicitações de isenção de taxa de inscrição foi entre as 10h do dia 01/06/2023 até as 17h do dia seguinte, onde foram recebidas **7.652 solicitações na 1ª etapa e 3.052 na 2ª etapa**, totalizando incríveis **10.704 solicitações ao total**, volume até o momento nunca recebido pela COMPEC.

A computação das solicitações foi realizada em um computador da marca Lenovo, modelo M93p com as seguintes especificações:

- Memória RAM DDR3-1600, com capacidade total de 16GB;
- Processador Intel® Core™ i5-4570 de 3.60 GHz;
- Sistema operacional Arch Linux de 64 bits, com *engine* gráfica GNOME;
- Monitor padrão da marca HP com resolução de 1920x1080 pixels.

Seguindo o fluxo padrão do processo de isenção, começamos pela preparação de dados para envio ao SISTAC, através da Tela de Envio. Toda a interação com a interface gráfica se mostrou estável e não foi detectado nenhum erro de execução ou travamentos, mesmo com alto volume de dados.

A exportação dos arquivos de envio geraram as seguintes métricas:

> implementando tabela <

## 5 Conclusões

### 5.1 Considerações Finais

Visando valores como agilidade, lisura e transparência no processamento das solicitações de isenção na modalidade CadÚnico, o *IsenSys* surgiu como agente concretizador destes anseios, possibilitando que um órgão gestor consiga preparar dados para envio ao SISTAC e, após seu processo de análise, gerar editais de publicação e relatórios com estatísticas e métricas.

Mesmo com sua implementação sendo relativamente simples, os resultados obtidos foram de grande satisfação tanto pela COMPEC, quanto pela sociedade, que tiveram um retorno ágil e preciso, respeitando todos os requisitos dispostos nos editais do concurso piloto e instruções normativas sobre o processo de isenção, de acordo com os manuais do SISTAC.

Pode-se afirmar então que o objetivo principal do *IsenSys* foi atingido. A aplicação foi construída utilizando uma linguagem de programação amplamente aceita e suportada no mercado, respeitando as boas práticas de programação e convenções internacionais, mantendo sempre a organização e comentários no código-fonte de forma a facilitar futuras colaborações.

### 5.2 Propostas de Atualizações

Não há nada que exista que não possa ser melhorado e, partindo desse pressuposto, algumas otimizações e adições de novos recursos já foram mapeadas durante o processo de desenvolvimento e que, devido ao curto tempo para implementação e conclusão desta monografia, ainda não puderam ser incorporados ao projeto. Listar-se-á algumas:

- Implementar layouts dinâmicos na interface gráfica, de forma a permitir o redimensionamento das janelas;
- Permitir que o número de edital contenha caracteres alfanuméricos de forma a abranger esta atualização recente do SISTAC;

- Adicionar interfaces gráficas para gerência dos recursos do sistema tais como logomarca e arquivo de situações;
- Integrar as outras modalidades de isenção de taxa de inscrição, como a modalidade de análise documental, definida pela Lei nº 12.799, de 10 de abril de 2013 [14], e doadores de medula óssea, regulamentado pela Lei nº 13.656, de 30 de abril de 2018 [15].

O código-fonte em sua íntegra está disponível no repositório GitHub sob os links:

- **IsenSys:** <<https://github.com/icompe-felipe/java-compec-isensys>>
- **Phills Libs (dependência):** <<https://github.com/icompe-felipe/phills-libs>>
- **Relatórios:** <<https://github.com/icompe-felipe/jasper-compec-isensys>>

## Referências

- 1 REPÚBLICA, P. da. *Decreto nº 6.593, de 2 de outubro de 2008*. 2023. Disponível em: <[https://www.planalto.gov.br/ccivil\\_03/\\_ato2007-2010/2008/decreto/d6593.htm](https://www.planalto.gov.br/ccivil_03/_ato2007-2010/2008/decreto/d6593.htm)>. Acesso em: 09 de Agosto de 2023. Citado 2 vezes nas páginas 14 e 15.
- 2 FOME, M. do Desenvolvimento Social e C. *Padrão para Envio das informações de candidatos para isenção de pagamento da taxa de inscrição em Concursos públicos realizados no âmbito do Poder Executivo Federal*. 2023. Disponível em: <[http://aplicacoes.mds.gov.br/sistac/publico/arquivos/Manual\\_Envio\\_Recebimento.pdf](http://aplicacoes.mds.gov.br/sistac/publico/arquivos/Manual_Envio_Recebimento.pdf)>. Acesso em: 09 de Agosto de 2023. Citado 9 vezes nas páginas 15, 24, 26, 27, 29, 33, 36, 38 e 42.
- 3 ORACLE. *O que é tecnologia Java e por que preciso dela?* 2023. Disponível em: <[https://www.java.com/pt-BR/download/help/whatis\\_java.html](https://www.java.com/pt-BR/download/help/whatis_java.html)>. Acesso em: 09 de Agosto de 2023. Citado na página 17.
- 4 FOUNDATION, A. *Commons Text*. 2023. Disponível em: <<https://commons.apache.org/proper/commons-text/>>. Acesso em: 09 de Agosto de 2023. Citado na página 18.
- 5 FOUNDATION, T. A. S. *Welcome to Apache Maven*. 2023. Disponível em: <<https://maven.apache.org/>>. Acesso em: 09 de Agosto de 2023. Citado 2 vezes nas páginas 18 e 20.
- 6 FOUNDATION, T. A. S. *Apache POI - the Java API for Microsoft Documents*. 2023. Disponível em: <<https://poi.apache.org/>>. Acesso em: 09 de Agosto de 2023. Citado na página 19.
- 7 FOUNDATION, E. *About the Eclipse Foundation*. 2023. Disponível em: <<https://www.eclipse.org/org/>>. Acesso em: 09 de Agosto de 2023. Citado na página 19.
- 8 JASPERSOFT. *JasperReports Library*. 2023. Disponível em: <<https://community.jaspersoft.com/project/jasperreports-library>>. Acesso em: 09 de Agosto de 2023. Citado na página 20.
- 9 STATOLOGY. *An Introduction to Jaro-Winkler Similarity (Definition Example)*. 2023. Disponível em: <<https://www.statology.org/jaro-winkler-similarity>>. Acesso em: 09 de Agosto de 2023. Citado 2 vezes nas páginas 21 e 48.
- 10 JODA-TIME. *Why Joda Time?* 2023. Disponível em: <<https://www.joda.org/joda-time>>. Acesso em: 09 de Agosto de 2023. Citado na página 22.
- 11 CIDADANIA, M. da. *Orientações Gerais do Sistema de Isenção de Taxas de Concursos (SISTAC)*. 2023. Disponível em: <[http://aplicacoes.mds.gov.br/sistac/publico/arquivos/Orientacoes\\_Gerais\\_2019\\_novo.pdf;jsessionid=46BA47F8F87D3A66614467AA04817B68](http://aplicacoes.mds.gov.br/sistac/publico/arquivos/Orientacoes_Gerais_2019_novo.pdf;jsessionid=46BA47F8F87D3A66614467AA04817B68)>. Acesso em: 09 de Agosto de 2023. Citado na página 24.

- 12 COMPEC-UFAM. *Editais nº 13/2023-GR, de 25 de abril de 2023 - Processo Seletivo Contínuo - PSC 2023 - Etapa 1, Projeto 2024*. 2023. Disponível em: <https://edoc.ufam.edu.br/bitstream/123456789/6870/1/Editais%2013%20de%202023.pdf>. Acesso em: 09 de Agosto de 2023. Citado na página 51.
- 13 COMPEC-UFAM. *Editais nº 14/2023-GR, de 25 de abril de 2023 - Processo Seletivo Contínuo - PSC 2023 - Etapa 2, Projeto 2024*. 2023. Disponível em: <https://edoc.ufam.edu.br/bitstream/123456789/6871/1/Editais%2014%20de%202023.pdf>. Acesso em: 09 de Agosto de 2023. Citado na página 51.
- 14 REPÚBLICA, P. da. *Decreto nº 12.799, de 10 de abril de 2013*. 2023. Disponível em: [http://www.planalto.gov.br/ccivil\\_03/\\_ato2011-2014/2013/lei/l12799.htm](http://www.planalto.gov.br/ccivil_03/_ato2011-2014/2013/lei/l12799.htm). Acesso em: 09 de Agosto de 2023. Citado na página 53.
- 15 REPÚBLICA, P. da. *Lei nº 13.656, de 30 de abril de 2018*. 2023. Disponível em: [https://www.planalto.gov.br/ccivil\\_03/\\_ato2015-2018/2018/lei/l13656.htm](https://www.planalto.gov.br/ccivil_03/_ato2015-2018/2018/lei/l13656.htm). Acesso em: 09 de Agosto de 2023. Citado na página 53.