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

Agradecimentos dirigidos àqueles que contribuíram de maneira relevante à elaboração do trabalho, sejam eles pessoas ou mesmo organizações.

Lorem ipsum

Autor

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

Advisor: 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 figuras

Figura 1 – Logomarca do *Java* 17
Figura 2 – Suíte de desenvolvimento *JasperReports®*. 18
Figura 3 – Logo da biblioteca *Apache POI*. 19
Figura 4 – Logo do gerente de projetos *Apache Maven*. 20
Figura 5 – *Eclipse IDE*. 21
Figura 6 – Modelagem da entidade *Candidato* 25
Figura 7 – Modelagem da entidade *CandidatoBuilder* 25
Figura 8 – Modelagem da entidade *RowParseException* 27
Figura 9 – Modelagem da entidade *ParseResult* 27
Figura 10 – Modelagem do importador de dados *CSVSheetReader* 29
Figura 11 – Modelagem do importador de dados *ExcelSheetReader* 29
Figura 12 – Diagrama de classe do exportador de dados *CSVSheetWriter* . 30
Figura 13 – Diagrama de classe do exportador de dados *ExcelSheetWriter* 30
Figura 14 – Diagrama do Fluxo de Dados dos Candidatos 31
Figura 15 – Interface Gráfica do Módulo de Envio 32
Figura 16 – Interface Gráfica do Módulo de Envio (arquivo carregado) . . 33
Figura 17 – Tela de Envio preenchida e arquivos de saída 34
Figura 18 – Modelagem da Entidade *Retorno* 35
Figura 19 – Diagrama de classe de *ListaRetornos* 35
Figura 20 – Diagrama de classe de *Compilation* 36
Figura 21 – Modelagem da Entidade *Situacao* 36
Figura 22 – Diagrama de classe de *SituacaoDAO* 37
Figura 23 – Diagramas de classe dos Geradores de Relatório 37
Figura 24 – Diagrama do Fluxo de Dados no Retorno Preliminar 38
Figura 25 – Interface Gráfica do Submódulo de Retorno Preliminar 39
Figura 26 – Interface Gráfica do Submódulo de Retorno Preliminar 40
Figura 27 – Arquivos de Saída do submódulo 'Retorno Preliminar' 41
Figura 28 – Diagrama do Fluxo de Dados no Retorno Definitivo 42

Figura 29 – Interface Gráfica do Submódulo de Retorno Definitivo 43

Lista de tabelas

Tabela 1 – Dados pessoais de candidatos e seus formatos. 24

Lista de abreviaturas e siglas

CSV	*Comma-separated values* - 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

1 INTRODUÇÃO 14

1.1 Objetivos 15

1.1.1 Objetivo Geral 15

1.1.2 Objetivos Específicos 15

1.2 Organização da Monografia 15

2 REFERENCIAL TEÓRICO 17

2.1 Java 17

2.2 JasperReports® 18

2.3 Apache POI 19

2.4 Apache Maven 19

2.5 Eclipse IDE 20

3 DESENVOLVIMENTO DO *ISENSYS* 22

3.1 Desenvolvimento do Módulo de Envio 24

3.1.1 Modelagem de um Candidato 24

3.1.2 A classe *CandidatoBuilder* 25

3.1.3 A classe de exceção *FieldParseException* 26

3.1.4 A classe de exceção *RowParseException* 26

3.1.5 A classe *ParseResult* 27

3.1.6 Importadores de Dados de Candidato 27

3.1.6.1 O importador *CSVSheetReader* 29

3.1.6.2 O importador *ExcelSheetReader* 29

3.1.7 Exportadores de Dados 29

3.1.7.1 O exportador *CSVSheetWriter* 30

3.1.7.2 O exportador *ExcelSheetWriter* 30

3.1.8 Integração do Módulo de Envio 31

3.1.9 Interface Gráfica do Módulo de Envio 32

3.2 Desenvolvimento do Módulo de Retorno 34
3.2.1 Modelagem de um Retorno 34
3.2.2 A classe *ListaRetornos* 35
3.2.3 A classe *Compilation* 36
3.2.4 Modelagem de uma Situacao 36
3.2.5 A classe *SituacaoDAO* 36
3.2.6 Geradores de Relatório 37
3.2.7 Integração do Módulo de Retorno 37
3.2.7.1 Retorno Preliminar 37
3.2.7.2 Interface Gráfica do Submódulo de Retorno Preliminar 38
3.2.7.3 Retorno Definitivo 41
3.2.7.4 Interface Gráfica do Submódulo de Retorno Definitivo 42

Referências 45

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 outra entidade do poder executivo do Brasil, adota três tipos de categorias 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, processar e gerar relatórios e editais de publicação, tomando como objeto de estudo a categoria de isenção mais volumosa em termos de solicitação: a categoria via CadÚnico, regulamentada pelo decreto n° 6.593, de 2 de outubro de 2008.

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

Este projeto possui como objetivo apresentar um aplicativo processador de solicitações de isenção de taxa de inscrição de acordo com a regulamentação do CadÚnico, de forma a permitir agilidade e acurácia nos resultados, por parte de uma unidade gestora do governo federal do Brasil.

1.1.2 Objetivos Específicos

- Implementar à risca o motor do sistema utilizando as normativas do manual do SISTAC;
- Utilizar das boas práticas de programação para tornar o projeto exportável futuramente;
- 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**: apresenta resultados e métricas que permitem ter um comparativo entre o antes e depois da implementação do *IsenSys*;

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

2 Referencial Teórico

Este capítulo referencia as principais ferramentas utilizadas no desenvolvimento da aplicação *IsenSys*, tais como linguagem de programação e bibliotecas de funções.

2.1 Java

A linguagem de programação Java [1] 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, forma de organização e versatilidade entre os vários sistemas operacionais.

Por ser uma linguagem independente de plataforma, ela permite que uma mesma aplicação possa ser executada em diversos 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.

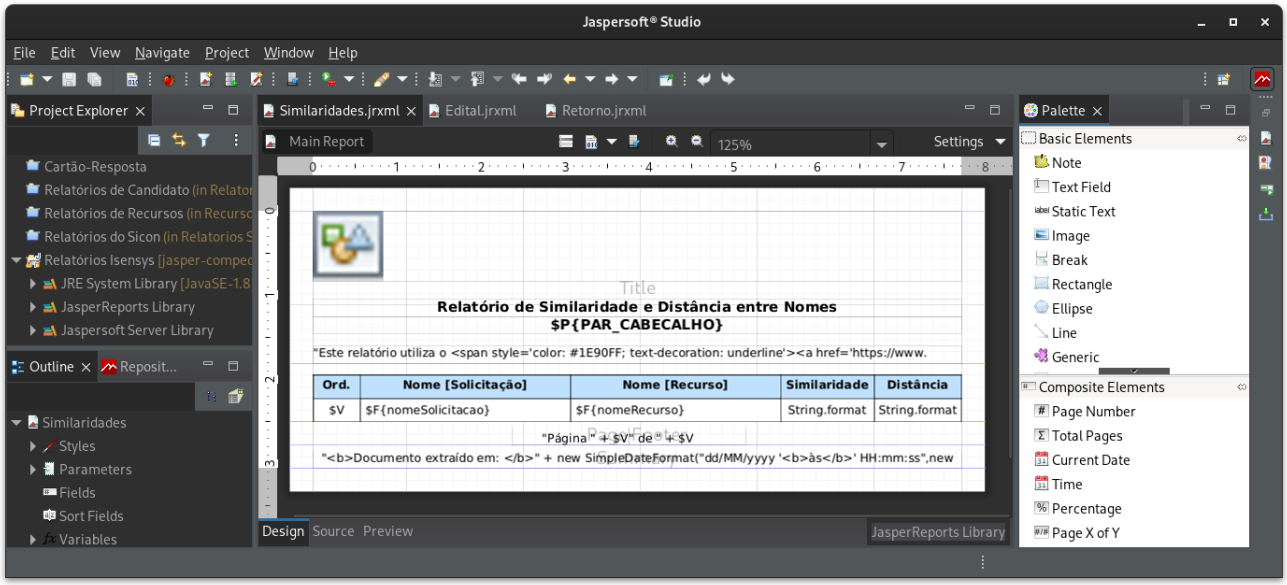
Provavelmente o leitor já tenha utilizado algumas das aplicações implementadas utilizando a tecnologia Java, tais como os programas de declaração de imposto sobre a renda (IRPF e IRPJ), Processo Judicial Eletrônico (PJe), *MATLAB*, famoso no mundo da engenharia e até o próprio sistema *Android*, um dos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

2.2 JasperReports®

A biblioteca *JasperReports*® [2] é gratuita e uma das mais robustas e populares utilizadas para criação de relatórios em *Java*. Possui sua própria suíte de design, onde o desenvolvedor pode rapidamente configurar um novo relatório com poucos passos, seguindo guias e instruções no editor.

Por ser desenvolvida utilizando a linguagem *Java*, ela também apresenta o mesmo comportamento 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 2 – Suíte de desenvolvimento *JasperReports*®.



Fonte: Produzida pelo autor

2.3 Apache POI

O *Apache POI* [3] 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>

2.4 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* [4] 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 específico denominado 'pom.xml'.

Figura 4 – Logo do gerente de projetos *Apache Maven*.

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

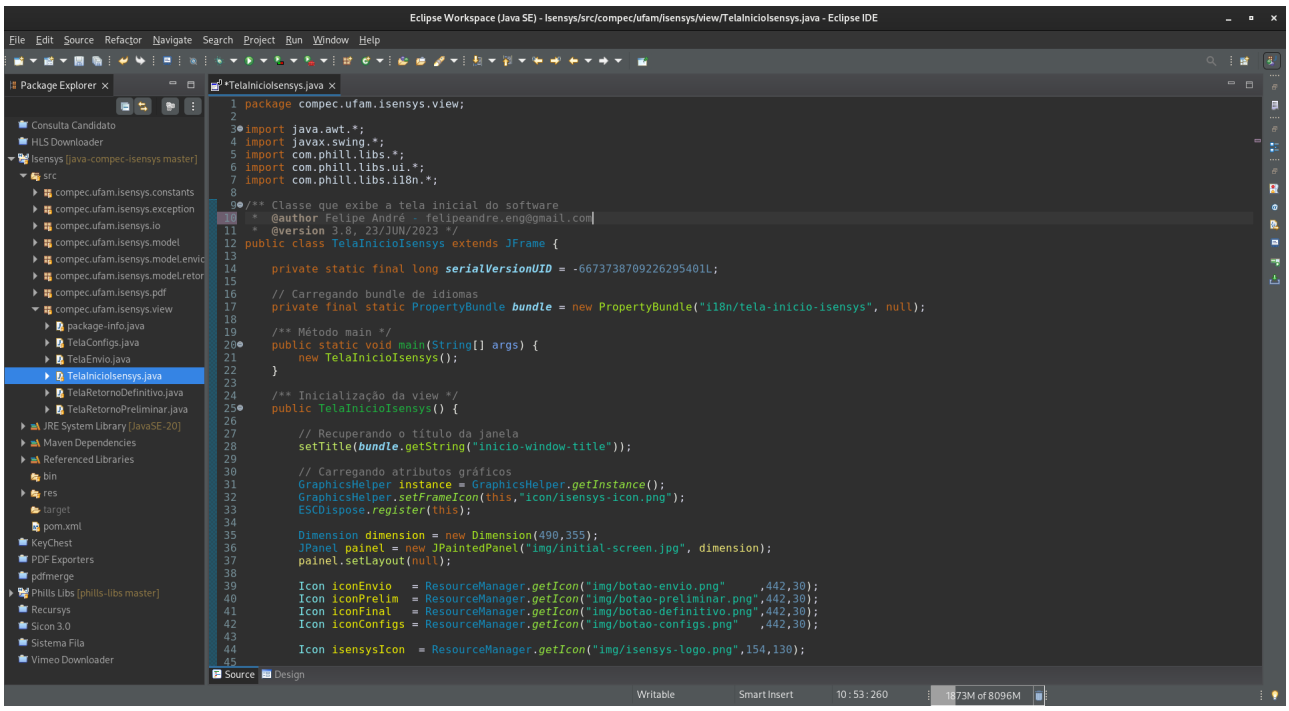
2.5 Eclipse IDE

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* [5] 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.

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 até o já mencionado *Apache Maven* [4].

Figura 5 – *Eclipse IDE*.



Fonte: Produzida pelo autor

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 completa porém, concisa.

O desenvolvimento do *motor* do sistema é regido pelas normas de formato de arquivo e diretivas definidas no documento de orientações gerais do SISTAC [6] e no manual de envio e recebimento de arquivos [7], 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 da instituição gerente (RF02);
- O sistema deverá realizar testes de integridade nos dados importados (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 candidatos em sua plena completude e integridade (RF05);
- Dados de candidatos que foram informados de forma incompleta ou inválida deverão ser exportados em uma planilha de erros (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 de resultados (RF08);

- O sistema também deve permitir a importação de uma planilha com erros, para inclusão nos editais públicos (RF09);
- O sistema deve gerar relatórios de estatísticas de candidatos deferidos e indeferidos (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 Edittion (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 à *JDK 15*, mínimo de 1GB de memória RAM disponível e monitor com resolução mínima de 800x600 (RNF08).

O sistema conta com dois módulos distintos: um dedicado a preparar os dados de candidatos para envio ao SISTAC e outro dedicado a processar os

arquivos de retorno do SISTAC e produzir relatórios e editais de publicação. As seções a seguir contém um estudo mais aprofundado sobre a implementação de cada módulo.

3.1 Desenvolvimento do Módulo de Envio

Esta seção tem por objetivo detalhar o desenvolvimento do motor *(backend)* de preparação de dados de candidatos 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.

3.1.1 Modelagem de um Candidato

Os dados pessoais de um candidato são o objeto principal deste sistema, pois são eles quem solicitam isenção e têm uma resposta de deferimento ou não destas. Segundo o manual do SISTAC [7], os dados pessoais de candidatos necessários para o processamento estão dispostos na tabela a seguir.

Tabela 1 – Dados pessoais de candidatos e seus formatos.

Campo	**Descrição**	**Máximo de Caracteres**	**Tipo**	**Formato**
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	

De posse dos dados e tipos, foi concebida a modelagem da entidade

Candidato, de acordo com o diagrama a seguir.

Figura 6 – Modelagem da entidade *Candidato*

Candidato
+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.

3.1.2 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 do SISTAC [7]. Segue o diagrama de classe do *CandidatoBuilder*.

Figura 7 – Modelagem da entidade *CandidatoBuilder*

CandidatoBuilder
+build() +parseNome() +parseNIS() +parseData() +parseSexo() +parseRG() +parseOrgao() +parseCPF()

Os métodos *parse...()*, responsáveis por realizar validações específicas em cada campo de dados de um *Colaborador*, 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()*.

3.1.3 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 que não foi validado.

3.1.4 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 8 – Modelagem da entidade *RowParseException*

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

3.1.5 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 que passaram com sucesso por todas as validações de campos;

2. Lista de *RowParseException*: onde são armazenadas a identificação do candidato e o(s) motivos(s) de invalidação de campo(s).

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

Figura 9 – Modelagem da entidade *ParseResult*

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

3.1.6 Importadores de Dados de Candidato

O *IsenSys* é capaz de importar dados pessoais de candidatos 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.

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:

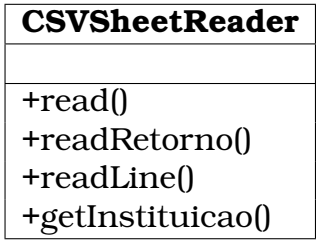
1. Nome completo;
2. NIS;
3. Data de nascimento;
4. Sexo;
5. Número de RG;
6. Data de emissão do RG;
7. Órgão emissor do RG;
8. CPF;
9. Nome completo da mãe.

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.

3.1.6.1 O importador *CSVSheetReader*

Para o arquivo *.csv* temos o importador descrito na classe *CSVSheetReader*, que é util 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 de uma linha. A seguir temos seu diagrama de classe.

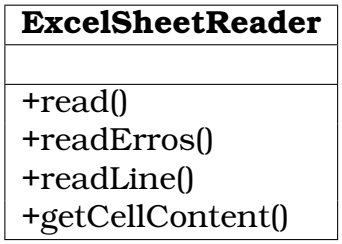
Figura 10 – Modelagem do importador de dados *CSVSheetReader*



3.1.6.2 O importador *ExcelSheetReader*

Este importador também é comum aos dois 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 e entregá-los ao *CandidatoBuilder*. Seu diagrama de classe está disposto na figura a seguir.

Figura 11 – Modelagem do importador de dados *ExcelSheetReader*



3.1.7 Exportadores de Dados

O estado final do processo de importação de dados dos candidatos solicitantes consiste na concentração deles na classe *ParseResult* [3.1.5]. A partir daqui, os dados dos candidatos considerados válidos pela classe *CandidatoBuilder* estão prontos para serem exportados para o arquivo no formato de envio

do SISTAC [7]. Os dados de solicitações inválidas são exportados para uma planilha do *Microsoft Excel*, para futuro processamento no módulo de retorno.

3.1.7.1 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. Na primeira linha do arquivo é impresso o cabeçalho com alguns dados da instituição gestora (configurados previamente) e nas demais, os dados são dispostos de acordo com o formatação exigida no manual do SISTAC [7].

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

Figura 12 – Diagrama de classe do exportador de dados *CSVSheetWriter*

CSVSheetWriter
+write() +getSistacFilename()

3.1.7.2 O exportador *ExcelSheetWriter*

Esta classe tem a função de exportar os dados de candidatos inválidos para uma planilha do *Microsoft Excxel*, onde a primeira linha contém um cabeçalho 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*.

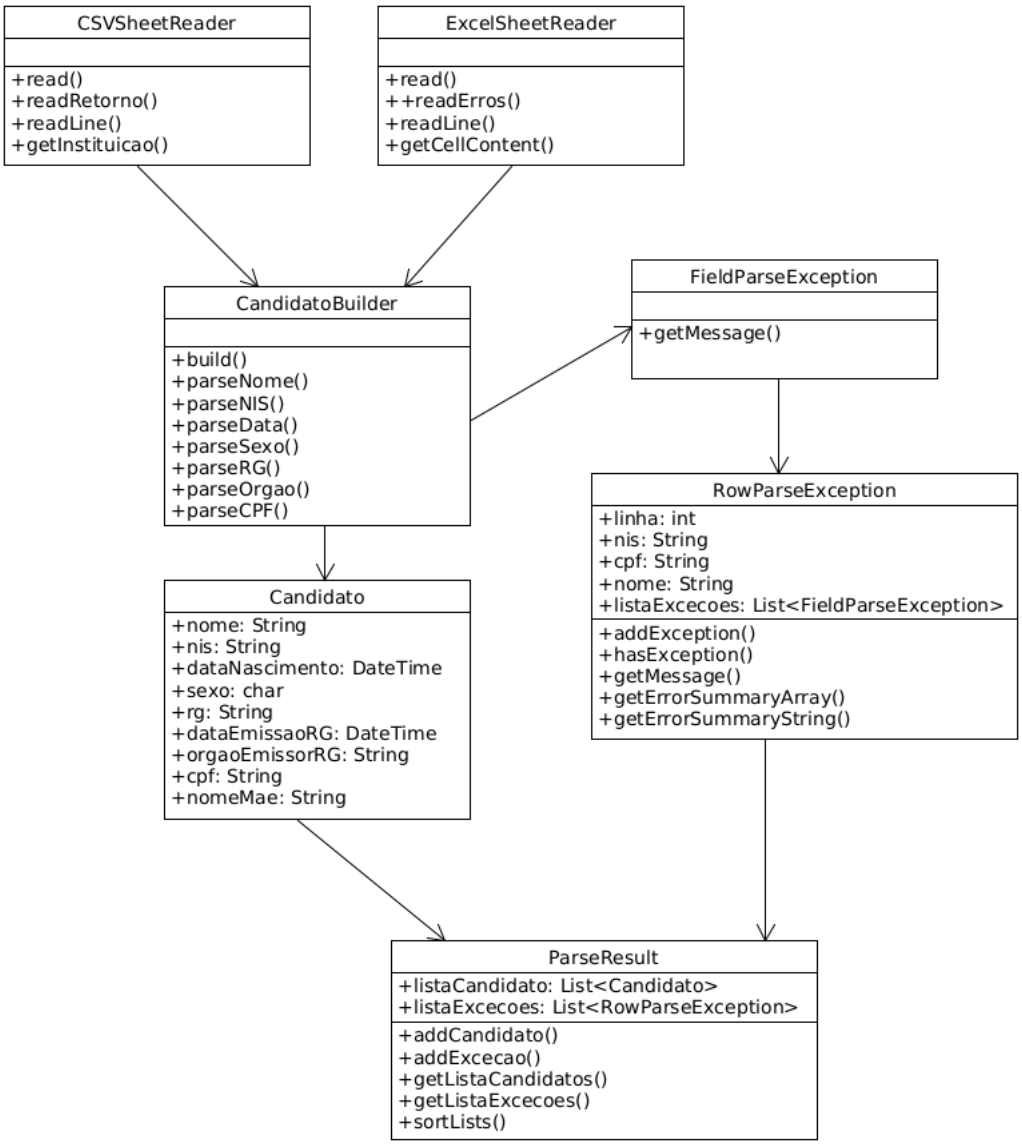
Figura 13 – Diagrama de classe do exportador de dados *ExcelSheetWriter*

ExcelSheetWriter
+write() +printHeader()

3.1.8 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 dos candidatos para envio ao SISTAC. A seguir temos uma visão geral do fluxo dos dados de um candidato, desde o arquivo de origem até o estágio onde todos estão devidamente processados.

Figura 14 – Diagrama do Fluxo de Dados dos Candidatos



Fonte: Produzida pelo autor

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 de *FieldParseException*). Por fim, os dados são concentrados no objeto *ParseResult* e estão prontos para exportação.

3.1.9 Interface Gráfica do Módulo de Envio

Como premissa 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 15 – Interface Gráfica do Módulo de Envio

Fonte: Produzida pelo autor

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 da instituição gestora.

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 previamente selecionado ou limpar sua seleção. Se um arquivo válido foi selecionado, algumas informações sobre o carregamento são mostradas no fundo deste painel, como ilustra a figura a seguir.

Figura 16 – Interface Gráfica do Módulo de Envio (arquivo carregado)

Fonte: Produzida pelo autor

Os dois primeiros campos do painel 'Arquivo de Saída' (Num. do Edital e Sequência) também são exigências descritas no manual de envio do SISTAC. 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, a exportação pode ser realizada clicando no ícone de salvar (disquete). As figuras abaixo ilustram a saída dos arquivos após alimentar a interface com algumas informações.

Figura 17 – Tela de Envio preenchida e arquivos de saída

Fonte: Produzida pelo autor

De posse dos arquivos exportados, o envio já pode ser realizado à plataforma do SISTAC. A planilha pode ser consultada de forma avulsa, mas será necessária para a montagem do edital de publicação no módulo de retorno.

3.2 Desenvolvimento do Módulo de Retorno

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

Utilizando a mesma metodologia abordada na seção anterior, vamos começar compreendendo o fluxo de informações conhecendo as classes envolvidas no processo e, posteriormente, conhecer a interface gráfica.

3.2.1 Modelagem de um Retorno

Para este módulo, 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 18 – Modelagem da Entidade *Retorno*

Retorno
+situacao: char +nome: String +nis: String +cpf: String +motivo: int +nomeAnterior: String
+defere() +isDeferido() +compareTo()

Fonte: Produzida pelo autor

3.2.2 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 que o módulo de retorno subdivide-se em dois: retorno preliminar e retorno definitivo. Por enquanto, temos que esta classe armazena dados úteis que facilitam a transição entre os dois tipos de retorno.

Figura 19 – Diagrama de classe de *ListaRetornos*

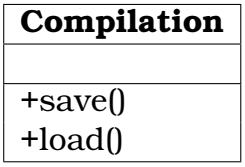
ListaRetornos
+listaRetornos: List<Retorno> +cnpj: String +nomeFantasia: String +razaoSocial: String +edital: String +dataEdital: String +cabecalho: String
+get() +size() +getList() +clone() +add() +update() +sort()

Fonte: Produzida pelo autor

3.2.3 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:

Figura 20 – Diagrama de classe de *Compilation*

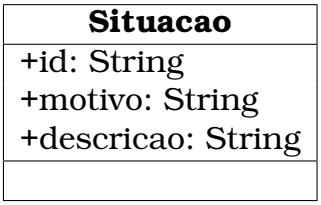


Fonte: Produzida pelo autor

3.2.4 Modelagem de uma Situacao

De acordo com o manual do SISTAC [7], existem algumas situações de indeferimento definidas previamente. No *IsenSys*, tais situações são armazenadas em um arquivo (*csv*) com codificação UTF-8 no diretório de recursos do sistema. Durante a confecção do edital de publicação, as situações são carregadas do arquivo e enviadas à classe geradora de relatórios, juntamente com os dados dos retornos.

Figura 21 – Modelagem da Entidade *Situacao*



Fonte: Produzida pelo autor

3.2.5 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 do edital de publicação.

Figura 22 – Diagrama de classe de *SituacaoDAO*

SituacaoDAO
+load()

Fonte: Produzida pelo autor

3.2.6 Geradores de Relatório

O *IsenSys* implementa três tipos de relatórios a partir dos dados provenientes do(s) retorno(s) 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 abaixo ilustra a estrutura de classe dos três geradores.

Figura 23 – Diagramas de classe dos Geradores de Relatório

PDFEdital
+export()

PDFRetorno
+export()

PDFSimilaridade
+export()

Fonte: Produzida pelo autor

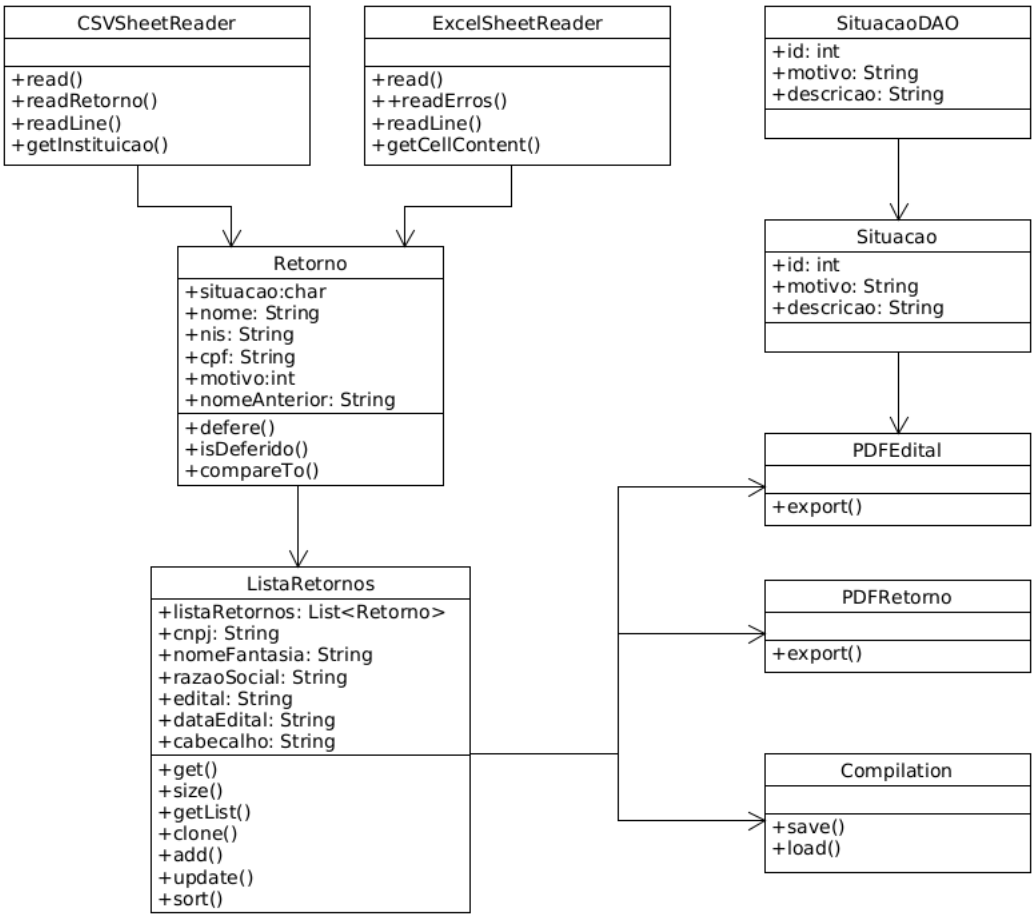
3.2.7 Integração do Módulo de Retorno

Conhecendo individualmente os agentes envolvidos no módulo de retorno, podemos realizar uma abordagem integrada. Existem dois tipos de retorno implementados no *IsenSys*:

3.2.7.1 Retorno Preliminar

Este é o submódulo destinado ao processamento do resultado preliminar das solicitações de isenção. A figura abaixo ilustra o fluxo das informações do submódulo.

Figura 24 – Diagrama do Fluxo de Dados no Retorno Preliminar



Fonte: Produzida pelo autor

Começando pelos arquivos de entrada, as classes *CSVSheetReader* e *ExcelSheetReader* carregam os arquivos 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, por fim, salvos em um arquivo binário denominado de 'compilação', através da classe *Compilation*.

3.2.7.2 Interface Gráfica do Submódulo de Retorno Preliminar

A figura abaixo ilustra a implementação da interface gráfica do submódulo de retorno preliminar do *IsenSys*.

Figura 25 – Interface Gráfica do Submódulo de Retorno Preliminar

Fonte: Produzida pelo autor

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 da instituição gestora.

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 dos arquivos. Abaixo 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*.

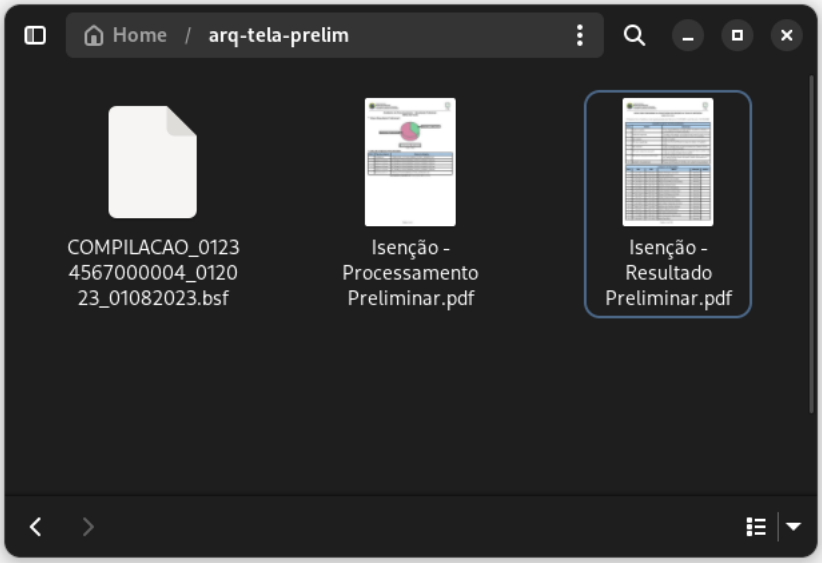
Se um arquivo válido foi selecionado, algumas informações sobre o carregamento são mostradas no canto inferior deste painel, como ilustra a figura a seguir.

Figura 26 – Interface Gráfica do Submódulo de Retorno Preliminar

Fonte: Produzida pelo autor

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 submódulo (edital de publicação, relatório de estatísticas e arquivo de compilação), como mostra a figura a seguir.

Figura 27 – Arquivos de Saída do submódulo 'Retorno Preliminar'

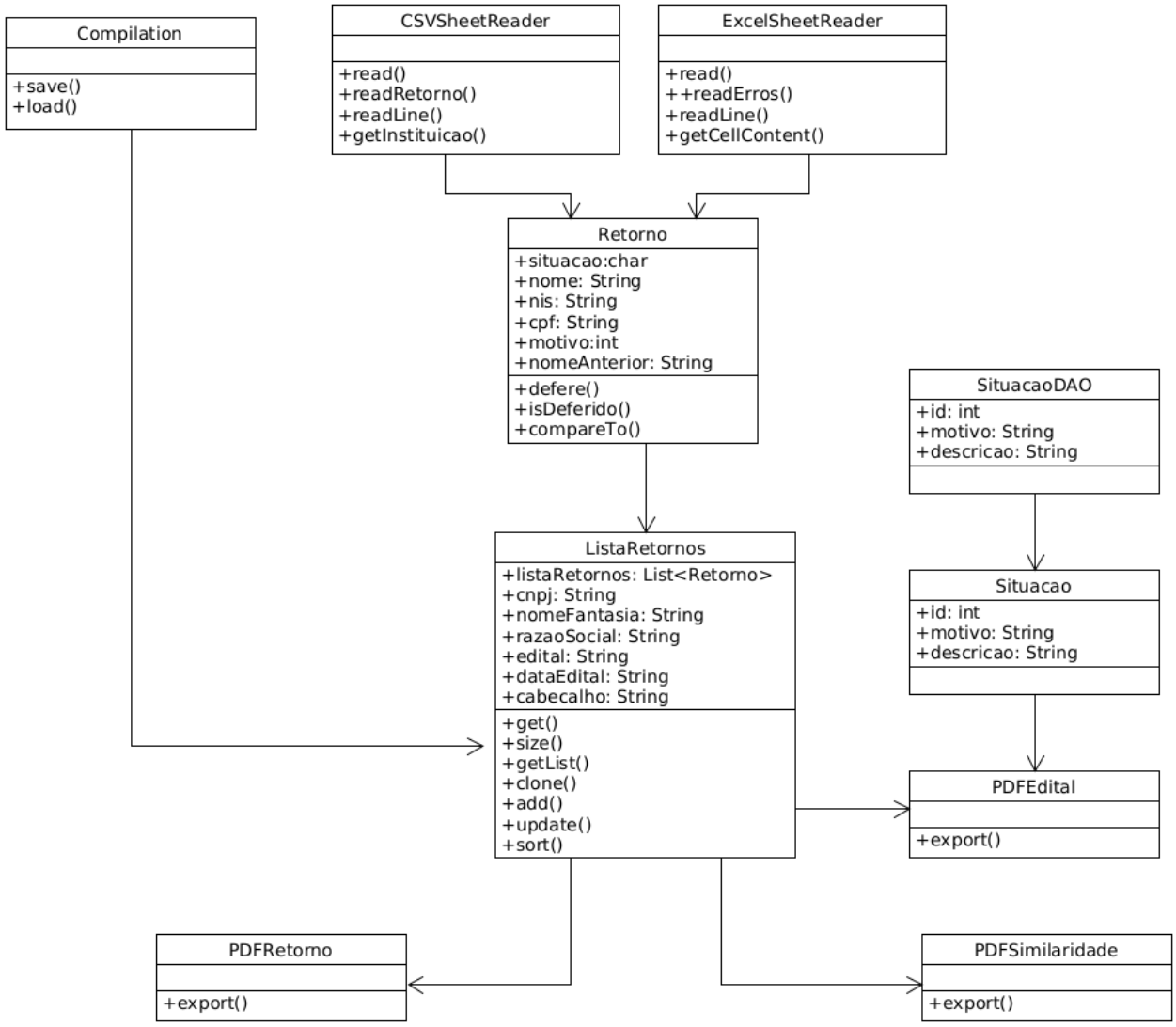


Fonte: Produzida pelo autor

3.2.7.3 Retorno Definitivo

Este é o submódulo destinado ao processamento do resultado definitivo. É capaz de recuperar as informações do submódulo preliminar, através de um arquivo de compilação (gerado por *Compilation*) e importar os novos retornos do SISTAC.

Figura 28 – Diagrama do Fluxo de Dados no Retorno Definitivo



Fonte: Produzida pelo autor

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

3.2.7.4 Interface Gráfica do Submódulo de Retorno Definitivo

A figura abaixo ilustra a implementação da interface gráfica do submódulo de retorno definitivo do *IsenSys*.

Figura 29 – Interface Gráfica do Submódulo de Retorno Definitivo

Fonte: Produzida pelo autor

Ao carregar o arquivo de compilação, alguma informações são exibidas no canto inferior do painel 'Resultado Preliminar'. O cabeçalho do edital também é carregado.

Da mesma forma que no submódulo 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;

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

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

Referências

1 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>. Acesso em: 09 de Agosto de 2023. Citado na página 17.

2 JASPERSOFT. *JasperReports Library*. 2023. Disponível em: <https://community.jaspersoft.com/project/jasperreports-library>. Acesso em: 09 de Agosto de 2023. Citado na página 18.

3 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.

4 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 19 e 20.

5 FOUNDATION, E. *About the Eclipse Foundation*. 2023. Disponível em: <https://www.eclipse.org/org>. Acesso em: 09 de Agosto de 2023. Citado na página 20.

6 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>. Acesso em: 09 de Agosto de 2023. Citado na página 22.

7 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>. Acesso em: 09 de Agosto de 2023. Citado 5 vezes nas páginas 22, 24, 25, 30 e 36.