

INSTITUTO SUPERIOR POLITÉCNICO DE TECNOLOGIAS E CIÊNCIAS

DEPARTAMENTO DE ENGENHARIAS E TECNOLOGIAS

CURSO DE ENGENHARIA INFORMÁTICA

ABEL NKELE CANAS

CARLOS NEVES TCHÍPIA



INSTITUTO SUPERIOR POLITÉCNICO DE TECNOLOGIAS E CIÊNCIAS

DEPARTAMENTO DE ENGENHARIAS E TECNOLOGIAS

CURSO DE ENGENHARIA INFORMÁTICA

ABEL NKELE CANAS

CARLOS NEVES TCHÍPIA

Relatório apresentado à disciplina de Programação II pelo Grupo nº 4 da turma EIN4\_T3 do curso de Engenharia Informática do ISPTEC, como requisito parcial para avaliação do projecto.

Docente: Sílvia da Conceição Veloso de Castro António

# RESUMO

O presente relatório descreve o desenvolvimento de um sistema de processamento de salários, elaborado no âmbito da disciplina de Programação II, do curso de Engenharia Informática. O sistema foi implementado em Java com o objetivo de simular as operações essenciais de um departamento de recursos humanos, incluindo o cadastro e gestão de colaboradores, definição de funções com salário base e bónus, cálculo automático de salário bruto, descontos fiscais (IRT e INSS) e salário líquido. Além disso, permite a emissão de holerites e a exportação de relatórios como também a importação e exportação de colaboradores e funções para o cadastro. A aplicação foi estruturada com base em princípios de modularidade, utilizando validação rigorosa de entradas e regras de negócio reais.

**Palavras-chave**: Processamento de salários, Colaboradores, Holerite, Cálculo de IRT.

# LISTA DE FIGURAS

[Figura 1 - Diagrama de Casos de Uso Geral do Sistema de Processamento de Salário 4](#_Toc202746219)

[Figura 2 - Diagrama de Casos de Uso detalhado do módulo 'Gerir Colaborador' 4](#_Toc202746220)

[Figura 3 - Diagrama de Casos de Uso detalhado do módulo 'Gerir Função' 5](#_Toc202746221)

[Figura 4 - Diagrama de Casos de Uso detalhado do módulo 'Processar Salário' 5](#_Toc202746222)

[Figura 5 - Diagrama de classes do programa 6](#_Toc202746223)

[Figura 6 - Tabela do IRT Angola (2023) publicado pela AGT 11](#_Toc202746224)

# SUMÁRIO

[1 INTRODUÇÃO 1](#_Toc202743984)

[2 Objectivos 1](#_Toc202743985)

[2.1 Objectivo Geral 1](#_Toc202743986)

[2.2 Objectivos Específicos 1](#_Toc202743987)

[3 Análise de Requisitos 2](#_Toc202743988)

[3.1 Requisitos Funcionais (RF) 2](#_Toc202743989)

[3.2 Requisitos Não-Funcionais (RNF) e Regras de Negócio (RN) 3](#_Toc202743990)

[4 Arquitectura e Desenho do Sistema (UML) 3](#_Toc202743991)

[4.1 Diagrama de Casos de Uso 3](#_Toc202743992)

[4.2 Diagrama de Classes 6](#_Toc202743993)

[5 METODOLOGIA 7](#_Toc202743994)

[5.1 Tecnologia e Ferramentas 7](#_Toc202743995)

[5.2 Divisão de Responsabilidades 7](#_Toc202743996)

[5.3 Estruturas Utilizadas e Decisões de Implementação 8](#_Toc202743997)

[6 CONCLUSÃO 9](#_Toc202743998)

[BIBLIOGRAFIA 10](#_Toc202743999)

[ANEXO A – TABELA DE IRT ANGOLA (2023) 11](#_Toc202744000)

# INTRODUÇÃO

Este relatório documenta o processo de concepção, desenvolvimento e implementação de um sistema de processamento de salários, desenvolvido no âmbito da disciplina de Programação II. O projeto foi elaborado com o objetivo de simular, de forma realista e automatizada, o processamento salarial numa empresa, utilizando a linguagem de programação Java.

Inicialmente, o sistema foi concebido para permitir a gestão de colaboradores e funções, oferecendo funcionalidades como cadastro, atualização, desativação e listagens organizadas. No entanto, com a evolução do trabalho, o projeto foi significativamente expandido para incluir operações reais de processamento de salários, como o cálculo automático de Imposto sobre Rendimento do Trabalho (IRT), Instituto Nacional de Segurança Social (INSS), salário bruto e líquido, além da emissão de comprovativos salariais (holerites), exportação de relatórios para ficheiros, importação e exportação de colaboradores e funções para automatizar o cadastro.

# Objectivos

## Objectivo Geral

Desenvolver uma aplicação em Java que simule o processamento de salários dos colaboradores de uma empresa

## Objectivos Específicos

* Permitir o cadastro, atualização e listagem de colaboradores e funções.
* Validar dados essenciais como emails, datas e vínculos funcionais.
* Calcular salário bruto, IRT, INSS e salário líquido.
* Emitir holerites mensais e gerar relatórios salariais em ficheiros.

# Análise de Requisitos

## Requisitos Funcionais (RF)

Os requisitos funcionais descrevem o que o sistema deve fazer.

* **RF-01: Gestão de Colaboradores**
  + **RF-01a:** O sistema deve permitir cadastrar um colaborador, identificado por um número, nome, morada, data de nascimento, função, data de admissão e um endereço de email único e válido.
  + **RF-01b:** O sistema deve permitir actualizar as informações de um colaborador existente.
  + **RF-01c:** O sistema deve permitir desactivar um colaborador, o que o torna inactivo, mas não o remove permanentemente.
  + **RF-01d:** O sistema deve permitir pesquisar e imprimir a informação de um colaborador específico.
  + **RF-01e:** O sistema deve permitir imprimir uma lista de todos os colaboradores activos.
  + **RF-01f:** O sistema deve ser capaz de gerar uma lista de colaboradores ordenada pela sua data de admissão.
  + **RF-01g:** O sistema deve permitir Importar colaboradores de um ficheiro.
  + **RF-01h:** O sistema deve permitir exportar colaboradores do sistema para um ficheiro para possível importação.
* **RF-02: Gestão de Funções**
  + **RF-02a:** O sistema deve permitir criar uma função, identificada por um código, nome, salário base e bónus.
  + **RF-02b:** O sistema deve permitir eliminar uma função.
  + **RF-02c:** O sistema deve imprimir a lista de todas as funções criadas.
  + **RF-01d:** O sistema deve permitir Importar funções de um ficheiro.
  + **RF-01e:** O sistema deve permitir exportar funções do sistema para um ficheiro para possível importação.
* **RF-03: Processamento de Salários**
  + **RF-03a:** O sistema deve permitir gerar holerite para cada colaborador
  + **RF-03b:** O sistema deve ser capaz de exportar holerites.
  + **RF-03b:** O sistema deve ser listar os holerites[[1]](#footnote-2) gerados.

## Requisitos Não-Funcionais (RNF) e Regras de Negócio (RN)

Estes requisitos definem como o sistema deve operar e as restrições a que está sujeito.

* **RNF-01: Tecnologia:** A solução deve ser desenvolvida na linguagem de programação **Java**.
* **RNF-02: Estrutura:** As entidades colaborador e função devem ser implementadas em classes próprias, e deve existir apenas uma classe principal com o método main.
* **RNF-03: Validação de Dados:**
  + **RNF-03a:** As datas inseridas no sistema devem ser validadas.
  + **RNF-03b:** O formato do endereço de email deve ser validado.
* **RNF-04 (RN):** Uma função só pode ser eliminada se não estiver associada a nenhum colaborador.
* **RNF-05 (RN):** As funções devem ser criadas antes da admissão de um colaborador nessa função.
* **RNF-06 (RN):** O sistema deve verificar se a função atribuída a um colaborador existe na lista de funções

# Arquitectura e Desenho do Sistema (UML)

Para visualizar a estrutura e o comportamento do sistema, foram criados os seguintes diagramas UML (Unified Modeling Language).

## Diagrama de Casos de Uso

Este diagrama ilustra as interacções entre o utilizador (Administrador de RH) e as principais funcionalidades do sistema.

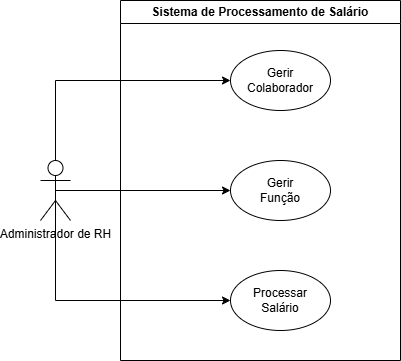


Figura 1 - Diagrama de Casos de Uso Geral do Sistema de Processamento de Salário

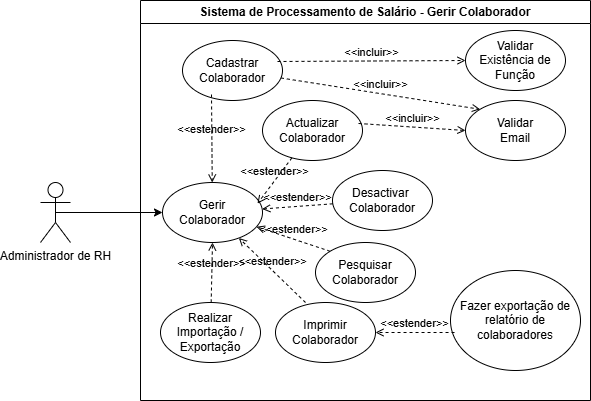


Figura 2 - Diagrama de Casos de Uso detalhado do módulo 'Gerir Colaborador'

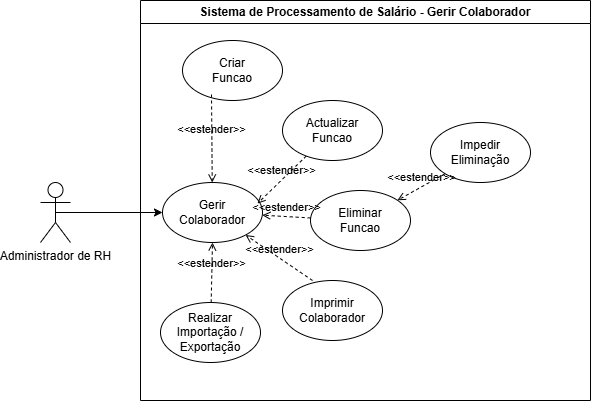


Figura 3 - Diagrama de Casos de Uso detalhado do módulo 'Gerir Função'

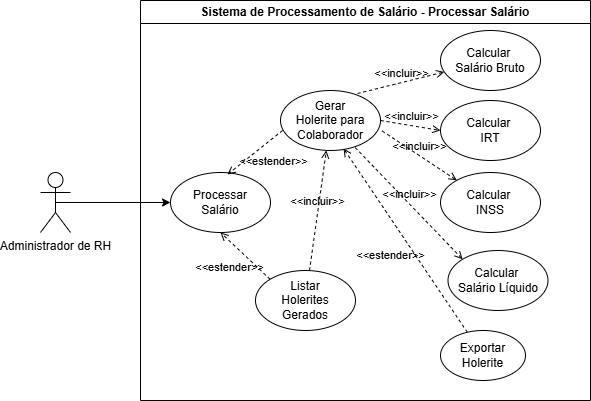


Figura 4 - Diagrama de Casos de Uso detalhado do módulo 'Processar Salário'

## Diagrama de Classes

Este diagrama detalha a estrutura estática do sistema, mostrando as classes, seus atributos, métodos e a relação entre elas.

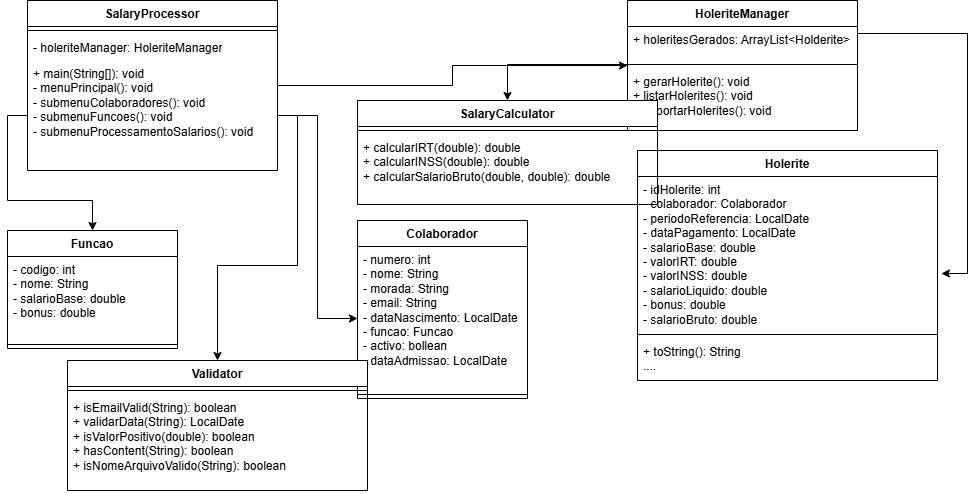


Figura 5 - Diagrama de classes do programa

# METODOLOGIA

A execução do projecto foi norteada por uma abordagem metódica, que incluiu a divisão de tarefas entre os membros da equipa e a selecção criteriosa das estruturas e técnicas de programação para atender aos requisitos do enunciado.

## Tecnologia e Ferramentas

* Linguagem: Java (JDK 23).
* IDE: Apache NetBeans IDE 25.
* Estruturas de Dados: java.util.ArrayList
* Controlo de Versões: Github

## Divisão de Responsabilidades

Para assegurar a eficiência e a qualidade do desenvolvimento, as tarefas foram distribuídas da seguinte forma entre os membros do Grupo 4:

* **Emanuel:**
  + Responsável pela arquitectura geral do projecto, incluindo a definição da estrutura de pacotes isptec.pii\_tp2.grupo4.projecto, conforme solicitado.
  + Implementação da classe principal (Main), contendo o menu de interacção com o utilizador e o fluxo principal da aplicação.
  + Coordenação da equipa, gestão da integração do código e garantia da coesão do projecto final.
* **Carlos:**
  + Responsável pela implementação completa da classe Colaborador e dos seus atributos (número, nome, morada, data de nascimento, função, data de admissão e email).
  + Desenvolvimento dos métodos para as operações de actualizar e desactivar um colaborador.
  + Implementação das rotinas de validação de dados, com especial atenção à validação do formato do endereço de email e à coerência das datas.
* **Abel:**
  + Responsável pela implementação da classe Funcao, com os seus respectivos atributos (código, nome, salário base e bónus).
  + Desenvolvimento dos métodos para criar, eliminar e imprimir a lista de todas as funções.
  + Implementação da lógica de negócio que impede a eliminação de uma função caso ela esteja associada a algum colaborador , bem como a verificação da existência da função no acto de cadastro de um colaborador

## Estruturas Utilizadas e Decisões de Implementação

**Linguagem:** O projecto foi integralmente desenvolvido na linguagem de programação **Java**, conforme estipulado no enunciado.

**Armazenamento de Dados em Memória:** Para o armazenamento dos objectos Colaborador e Funcao durante a execução do programa, optou-se pela utilização da estrutura de dados java.util.ArrayList. Esta escolha justifica-se pela sua natureza dinâmica, que permite a fácil inserção e remoção de elementos, adequando-se perfeitamente às necessidades de cadastro, eliminação e listagem do sistema.

**Validação de Dados:** Para garantir a integridade e a robustez dos dados inseridos, foram implementados vários mecanismos de validação:

* **Unicidade e Formato do Email:** O sistema verifica se o endereço de email fornecido no momento do cadastro já existe e se obedece a um formato válido (ex: utilizador@dominio.com).
* **Validação de Datas:** As datas de nascimento e de admissão são validadas para assegurar a sua correcção lógica e de formato.
* **Integridade Referencial:** Foi implementada uma verificação que garante que um colaborador só pode ser associado a uma função que exista previamente no sistema.

**Geração de Relatórios:** A funcionalidade de imprimir a lista de colaboradores por ordem de admissão foi implementada através da ordenação do ArrayList de colaboradores. Utilizou-se um algoritmo de ordenação para organizar a lista com base no atributo data de admissão antes de a apresentar ao utilizador.

# CONCLUSÃO

A conclusão deste trabalho prático representou um marco importante na consolidação das competências da equipa na disciplina de Programação II. O projecto permitiu-nos traduzir requisitos de um problema real num software funcional, aplicando de forma directa os conceitos teóricos leccionados.

Principais Aprendizagens: O desenvolvimento reforçou a nossa compreensão sobre a modelação de entidades através de classes, a manipulação de colecções de dados com ArrayLists, a importância crítica da validação de dados de entrada e a necessidade de planear e implementar regras de negócio para garantir a consistência da informação. O trabalho em equipa, a comunicação constante e a divisão de responsabilidades foram factores determinantes para o sucesso.

Principais Dificuldades Encontradas: A gestão da interdependência entre as classes Colaborador e Funcao (principalmente na funcionalidade de eliminação) foi um dos principais desafios, exigindo uma lógica de verificação cuidadosa. Outra dificuldade inerente ao enunciado foi a natureza volátil dos dados; como não há persistência em ficheiro ou base de dados, toda a informação é perdida ao encerrar a aplicação. Esta limitação, contudo, serviu para focar os esforços na lógica de programação em memória.

Em suma, o projecto foi concluído com sucesso, cumprindo todos os requisitos funcionais solicitados no enunciado e proporcionando uma experiência de aprendizagem prática e enriquecedora.

# BIBLIOGRAFIA

# ANEXO A – TABELA DE IRT ANGOLA (2023)



Figura 6 - Tabela do IRT Angola (2023) publicado pela AGT

1. **Holerite** é o termo utilizado para designar o comprovativo de pagamento emitido ao colaborador, contendo todos os valores relacionados ao seu salário. [↑](#footnote-ref-2)