Discentes: Jonatan Fragoso, Elias Cacau

Título: Planejamento de Desenvolvimento Orientado a Testes (TDD) Assistido por IA: Estratégia, Métricas e Ferramentas.

1. Introdução

Este documento descreve o planejamento para o desenvolvimento da solução do cálculo salarial (Cálculo de Salário Líquido, INSS e IRRF). A abordagem adotada será o *Test Driven Development* (TDD), utilizando a liguagem Java, e, como IA de auxílio a plataforma **Google Gemini** , usando os diferentes modelos para garantir a qualidade do código.

2. Atributos de Qualidade Selecionados

Os seguintes atributos de qualidade foram priorizados:

- **Adequação Funcional:** Devido à natureza financeira do problema (cálculos tributários), a prioridade máxima é a *Corretude*. O sistema deve garantir precisão decimal exata e adesão estrita às faixas de INSS/IRRF, sem comportamentos imprevistos.
- **Manutenibilidade:** O foco será na *Testabilidade*. O código deve ser estruturado de forma que cada regra de negócio (ex: cálculo de uma faixa de imposto) seja isolada e verificável individualmente, facilitando futuras atualizações.

3. Métricas de Software

Para monitorar a qualidade interna e a eficácia do TDD, serão utilizadas as seguintes métricas, aferidas através dos recursos de análise estática nativos da IDE:

- **Cobertura de Código (Code Coverage):**
    - *Ferramenta:* Executor de cobertura nativo do IntelliJ IDEA.
    - *Justificativa:* Utilizada no TDD para validar visualmente se as regras de negócio implementadas foram executadas pelos testes. O foco é garantir que não existam linhas de lógica de cálculo não cobertas por testes.
- **Complexidade Ciclomática (Cyclomatic Complexity):**
    - *Ferramenta:* Análise estática nativa da IDE (Linter).
    - *Justificativa:* O código deve permanecer legível e simples. Serão observados os alertas nativos da IDE que indicam excesso de condicionais ou aninhamento profundo, servindo como atenção para a refatoração.

4. Ferramentas de IA e Desenvolvimento

- **Linguagem:** Java.
- **Estratégia de IA (Google Gemini):**
    1. **Gemini (Modo Pensante/Raciocínio):** Será utilizado na fase de planejamento. Seu foco será analisar as regras de negócio para deduzir cenários de teste complexos e casos de exceção (ex: salários negativos, limites de isenção) antes de qualquer código ser escrito.
    2. **Gemini (Versão Pro):** Será utilizado na fase de implementação e refatoração (*Green/Refactor Phase*). Seu foco será na velocidade de escrita da sintaxe Java, implementação dos métodos para fazer os testes passarem e sugestões de limpeza de código.

Link do Repositório: https://github.com/jonatanfragoso/Trabalho-TDD