

**CURSO DE ENGENHARIA DE SOFTWARE**

**RELATÓRIO – TRABALHO FINAL QUALIDADE DE SOFTWARE**

**<Swing JPA CRUD>**

**Equipe:**

**Fabiany de Sousa Costa**

**Professora:**

**Carla Ilane Moreira Bezerra**

**QUIXADÁ**

**Julho, 2021**

**SUMÁRIO**

[1 DESCRIÇÃO DO PROJETO 2](#_Toc63168868)

[2 AVALIAÇÃO DO PROJETO 2](#_Toc63168869)

[2.1 Medição 1 – Antes de refatorar o projeto 2](#_Toc63168870)

[2.2 Detecção dos Code Smells 3](#_Toc63168871)

[2.3 Medição 2 – Após Refatorar Code Smell X 4](#_Toc63168872)

[2.4 Medição 3 – Após Refatorar Code Smell Y 4](#_Toc63168873)

[2.5 Medição Z – Após a refatoração de todos os code smells do projeto 4](#_Toc63168874)

[3 COMPARAÇÃO DOS RESULTADOS 4](#_Toc63168875)

[REFERÊNCIAS 4](#_Toc63168876)

[APÊNDICE A 5](#_Toc63168877)

# DESCRIÇÃO DO PROJETO

O projeto é uma demonstração de aplicativo *desktop*, com o código aberto, desenvolvido com as tecnologias *Swing*, JPA (*Java Persistence* API) e *Hibernate*.

A aplicação utiliza o HSQLDB (HyperSQL DataBase), um banco de dados relacional escrito em Java, adequado para projetos com propósitos de estudos.

Essa aplicação disponibiliza um CRUD, com funcionalidades idênticas do projeto *swing-jdbc-crud* (<https://github.com/yaw/swing-jdbc-crud>). Além de utilizar uma tecnologia padrão para o mapeamento objeto relacional (ORM), essa aplicação define a arquitetura MVC (*Model View Controller*).

Tecnologias utilizadas na implementação:

* *Swing*: utilizamos o *framework* para construção das interfaces e componentes gráficos da aplicação (camada cliente);
* JPA: API alto nível, padrão da tecnologia Java, para definir o mapeamento objeto relacional (ORM).
* *Hibernate*: provedor JPA para mapeamento objeto relacional (ORM).
* *Collection*: reunimos uma relação de objeto em memória via coleções do Java;
* *Thread*: algumas ações (eventos) dos componentes da tela com o banco de dados são tratados em outra *thread* (*SwingUtilities*), de forma que o usuário tenha uma melhor experiência no uso da aplicação.

Para facilitar o uso de bibliotecas externas e a construção, o projeto utiliza o **Maven**.

Link do projeto: <https://github.com/yaw/swing-jpa-crud>

Tabela 1 – Características do Projeto

|  |  |  |  |
| --- | --- | --- | --- |
| **Projeto** | **LOC** | **# de classes** | **# de releases** |
| Swing JPA CRUD | 1.369 | 52 | No releases published |

# AVALIAÇÃO DO PROJETO

## Medição 1 – Antes de refatorar o projeto

Segue o link da Tabela com a medição das métricas de coesão, acoplamento, complexidade, herança e tamanho, antes do projeto ser refatorado. Foi utilizado a ferramenta *Understand* para a coleta das métricas. A Tabela 2 apresenta uma prévia da informações contidas na tabela completa referenciada no link.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Arquivo** | **Coesão** | **Complexidade** | | | | **Herança** | | | **Acoplamento** | **Tamanho** | | | |
| LCOM2 | ACC | SCC | EVG | MaxNet | DIT | NOC | IFANIN | CBO | LOC | CLOC | NIM | CDL |
| ConditionalAction | 60 | 1 | 8 | 1 | 1 | 2 | 0 | 1 | 1 | 1.369 | 556 | 178 | 52 |
| TransactionalAction | 56 | 2 | 16 | 1 | 2 | 2 | 0 | 1 | 1 |  |  |  |  |
| AbstractController | 82 | 1 | 27 | 1 | 1 | 1 | 1 | 3 | 3 |  |  |  |  |
| IncluirMercadoriaController | 50 | 1 | 4 | 1 | 2 | 3 | 0 | 1 | 6 |  |  |  |  |
| ListaMercadoriaController | 58 | 1 | 3 | 1 | 0 | 3 | 0 | 1 | 13 |  |  |  |  |

Tabela 2 – Medição dos atributos antes de refatorar o projeto.

## Detecção dos Code Smells

Utilizando as ferramentas JSPirit e JDeodorant foram detectados 39 *code smells* no projeto, sendo o tipo *Long Method* detectado pela ferramenta JDeodorant já que a JSPirit identificou apenas 4 tipos diferentes de *code smells*.

Tabela 3 – *Code smells* do projeto.

|  |  |
| --- | --- |
| **Nome do Code Smell** | **Quantidade** |
| Feature Envy | 10 |
| Dispersed Coupling | 2 |
| Refused Parent Bequest | 1 |
| Intensive Coupling | 1 |
| Long Method | 25 |

## Medição 2 – Após Refatorar Code Smell X

Nessa Seção você deve indicar os valores de todas as métricas da Tabela 2, após refatorar um determinado code smell. Esse code smell deve ser totalmente refatorado até não ser mais detectado pela JSPirit. Você deve também incluir a técnica de refatoração utilizada para retirar o code smell. Isso deve ser feito para cada code smell detectado no projeto. Após a refatoração de cada code smell deve ser realizada uma nova medição na ferramenta Understand. Deve ser realizada também uma análise dos 5 atributos de qualidade e que métricas pioram ou melhoram de acordo com a retirada desses code smells.

## Medição 3 – Após Refatorar Code Smell Y

.....

## Medição Z – Após a refatoração de todos os code smells do projeto

Após todos os code smells refatorados, deverá ser realizada a medição final do projeto conforme as métricas da Tabela 2. Deve também ser feita a análise final se as métricas pioraram ou melhoraram de acordo com a retirada dos code smells.

# COMPARAÇÃO DOS RESULTADOS

Leia o artigo: <https://www.sciencedirect.com/science/article/pii/S0950584920301142?casa_token=xcwL1BwaRFUAAAAA:wZjXB0Wx-0FiMSpZSzyi0b7iRe7ZJOr8FdwihzEkvzeQHh0Iz6mxPCF769JgRiZ69TyfI5l8BP0>

Faça uma comparação dos resultados do seu projeto de acordo com esse artigo.

# REFERÊNCIAS

AZEEM, Muhammad. Machine learning techniques for code smell detection: A systematic literature review and meta-analysis. Information and Software Technology, v. 108, p. 115-138, 2019.

SABIR, Fatima. A systematic literature review on the detection of smells and their evolution in object‐oriented and service‐oriented systems. Software: Practice and Experience, v. 49, n. 1, p. 3-39, 2019.

# APÊNDICE A

Incluir possíveis documentos que possam ser gerados no desenvolvimento do sistema.