



**UNIVERSIDADE
FEDERAL DO CEARÁ**
CAMPUS DE QUIXADÁ

CURSO DE ENGENHARIA DE SOFTWARE

RELATÓRIO – TRABALHO FINAL QUALIDADE DE SOFTWARE

Neodash

Equipe:

Antônio Hugo Ribeiro Pereira Lobo

Felipe Rodrigues de Santana Freitag

Professora:

Carla Ilane Moreira Bezerra

QUIXADÁ

Outubro, 2023

SUMÁRIO

1	DESCRIÇÃO DO PROJETO	2
2	<i>ISSUES</i> DO PROJETO DE SOFTWARE LIVRE	2
3	AVALIAÇÃO DO PROJETO	4
3.1	Medição 1 – Antes de refatorar o projeto	4
3.2	Detecção dos Code Smells	5

1 DESCRIÇÃO DO PROJETO

[Neodash](#) é uma ferramenta de código aberto, feita utilizando a biblioteca React e utilizando TypeScript, para visualizar dados do Neo4j, que é um sistema de gerenciamento de banco de dados orientado a grafos e é projetado para armazenar, recuperar e manipular dados que possuem relações complexas e interconexões. Além disso, a ferramenta já possui 41 *releases* e mais de 650 *commits* na branch principal. Ademais, o projeto está sendo mantido atualmente e possui várias *branches* ativas, tais *branches* buscam adicionar funcionalidades para a aplicação, para mais e 31 *pull requests* esperando análise.

O projeto [Neodash](#) organiza suas *issues* com alguns rótulos, como: bug, “*enhancement*”, “*documentation*”, entre outros. Com isso, eles conseguem uma melhor organização e fazem com que os contribuidores tenham mais facilidade ao procurar uma *issue* para corrigir. Além disso, cada *issue* possui um título e uma descrição para um melhor entendimento do problema e com essa descrição os contribuidores ou até mesmo quem criou a *issue* podem por meio de uma *pull request* corrigir o problema.

Link do projeto: <https://github.com/neo4j-labs/neodash>

Tabela 1 – Características do Projeto

Projeto	LOC	# de classes	# de funções	# de releases
Neodash	24.317	7	1.633	41

Por se tratar de um código em React o número de classes é baixo, já que em suas versões atuais os componentes são implementados utilizando funções e, por isso, o número de funções está sendo reportado.

2 ISSUES DO PROJETO DE SOFTWARE LIVRE

O projeto possui 131 *issues* abertas e os possíveis tipos são: *bug*, *documentation*, *duplicate*, *enhancement*, *good first issue*, *help wanted*, *priority*, *question*, *to do* e *wontfix*. O significado delas é:

- **Bug:** utilizado para *issues* que são frutos de bugs no código.

- **Documentation:** utilizado para quando se trata de uma melhoria ou adição na documentação.
- **Duplicate:** utilizado para quando a *issue* é igual a outra que já está aberta.
- **Enhancement:** utilizado para pedidos de melhoria ou adição de novas funcionalidades.
- **Good first issue:** *issues* boas para novatos que estão querendo contribuir com o projeto.
- **Help wanted:** utilizado para quando os contribuidores precisam de ajuda para resolver a *issue*.
- **Priority:** a *issue* deve ser considerada antes das demais.
- **Question:** utilizado para quando a *issue* aberta se trata de uma pergunta.
- **To do:** foi definido o que precisa ser feito, mas ainda precisa ser implementado.
- **Wontfix:** utilizado para *issues* que os contribuidores não pretendem resolver.

De todas as *issues* abertas, nenhuma delas parecia muito fácil de resolver por se tratar de um sistema um pouco mais complexo, então arimos o sistema e procuramos algum bug simples que conseguíssemos resolver. Após algum tempo procurando conseguimos achar um bug relacionado a *user experience*, o *bug* era seguinte: caso o usuário clicasse em criar um dashboard ou acessar um dashboard sem querer e quisesse voltar, não havia um botão para isso no modal, ou seja, o único jeito dele voltar era atualizando a página. Para abrir a *issue* tivemos que primeiro selecionar se o que iríamos relatar se tratava de um *bug* ou sugestão de melhoria, após marcamos *bug* fomos apresentados as diretrizes impostas que eram: *issues* devem ser para avisar sobre *bugs* ou sugerir melhorias, o usuário deve procurar nas outras *issues* se já não há algo relacionado, o usuário deve dizer a versão do sistema, entre outras.

Figura 1 - Problema de navegação encontrado

Connect to Neo4j

Protocol

neo4j

Hostname

localhost

Port

7687

Database (optional)

neo4j

Username

neo4j

Password

••••••••

Connect

A *issue* pode ser encontrada nesse [link](#), e a *pull request* aberta pode ser vista [aqui](#).

3 AVALIAÇÃO DO PROJETO

3.1 Medição 1 – Antes de refatorar o projeto

Utilizando a ferramenta Understand as seguintes medições foram coletadas e apresentadas na tabela abaixo.

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

Sistema	Complexidade				Tamanho		
	CC	ACC	SCC	MaxNest	LN	CLOC	CountDeclFunction
S1 antes da refatoração	3.176	1,78	8.525	14	49.636	3.072	1.633

Significado das siglas e termos da Tabela 2: CC - Complexidade Ciclométrica, ACC - Complexidade Ciclométrica Média, SCC - Complexidade Ciclométrica Total, MaxNest - aninhamento máximo, LN - número de linhas físicas, CLOC - número de linhas de comentário, CountDeclFunction - número de declarações de funções.

3.2 Detecção dos Code Smells

Os dados coletados abaixo a respeito dos *code smells* foram coletados utilizando a ferramenta Reactsniffer capaz de detectar *code smells* em códigos feitos utilizando a biblioteca React.

Tabela 3 – *Code smells* do projeto.

Nome do <i>Code Smell</i>	Quantidade
<i>Large Component</i>	46
<i>Too many props</i>	43
<i>Any Type</i>	10
<i>Missing Union Type</i>	4
<i>Enum Implicit Values</i>	4
<i>Direct DOM Manipulation</i>	2
<i>Inheritance instead of composition</i>	2
<i>Uncontrolled component</i>	1