Universidade Federal de Pernambuco
Centro de Informática

Graduação em Ciência da Computação

# Análise de bugs L10n em repositórios open-source no Github

Marcos Vinicius Prysthon Nascimento

Trabalho de Graduação

Recife - PE
03/04/2025

MARCOS VINICIUS PRYSTHON NASCIMENTO

# Análise de bugs L10n em repositórios open-source no Github

Trabalho de Conclusão de Curso apresentado ao Curso de Graduação em Ciência da Computação da Universidade Federal de Pernambuco, como requisito parcial para obtenção do título de bacharel em Ciência da Computação.

Orientador (a): Breno Alexandro Ferreira de Miranda

Recife

2024

Ficha de identificação da obra elaborada pelo autor,
através do programa de geração automática do SIB/UFPE

Nascimento, Marcos Vinicius Prysthon.
Análise de bugs L10n em repositórios open-source no Github / Marcos Vinicius Prysthon Nascimento. - Recife, 2025.
31p

Orientador(a): Breno Alexandro Ferreira de Miranda
Trabalho de Conclusão de Curso (Graduação) - Universidade Federal de Pernambuco, Centro de Informática, Ciências da Computação - Bacharelado, 2025.

1. Localization (L10n). 2. Dataset. 3. Issues de localização. 4. Análise de issues. I. Miranda, Breno Alexandro Ferreira de. (Orientação). II. Título.

000 CDD (22.ed.)

MARCOS VINICIUS PRYSTHON NASCIMENTO

**Análise de bugs L10n em repositórios open-source no Github**

Trabalho de Conclusão de Curso apresentado ao Curso de Graduação em Ciência da Computação da Universidade Federal de Pernambuco, como requisito parcial para obtenção do título de bacharel em Ciência da Computação.

Aprovado em: 03/04/2025

**BANCA EXAMINADORA**

_____

Prof. Dr. Breno Alexandro Ferreira de Miranda (Orientadora)

Universidade Federal de Pernambuco

_____

Prof. Dr. Breno Alexandro Ferreira de Miranda (Examinador Interno)

Universidade Federal de Pernambuco

_____

Prof. Dr. Leopoldo Motta Teixeira (Examinador Interno)

Universidade Federal de Pernambuco

# Agradecimentos

Agradeço imensamente à minha família, pelo apoio incondicional em cada etapa desta jornada. Ao meu orientador, Breno Miranda, pela orientação valiosa e paciência ao longo do desenvolvimento deste trabalho. À minha faculdade, pela oportunidade de crescimento acadêmico e pelas ferramentas que me permitiram chegar até aqui. A todos, minha mais sincera gratidão.

# Resumo

Este trabalho tem como objetivo a criação de um dataset contendo issues relacionadas a problemas de localização (L10n) em projetos open source no GitHub. A principal motivação foi disponibilizar um repositório de dados que auxilie o desenvolvimento de ferramentas automatizadas para a detecção de problemas de L10n. Para isso, foi desenvolvida uma ferramenta de coleta automatizada que utilizou a API do GitHub para buscar issues por meio de palavras-chave e labels específicas relacionadas à localização, como "L10n", "localization bug", "localization issue", entre outras. As issues coletadas foram filtradas para garantir a presença de screenshots, elemento fundamental para a validação dos problemas reportados.

Os dados coletados foram organizados e armazenados em uma tabela pública no Google BigQuery, permitindo consultas diretas aos dados estruturados. Além disso, foi desenvolvida uma API para facilitar a recuperação dos dados e um aplicativo em Streamlit, que fornece uma interface interativa para que usuários possam visualizar e filtrar as issues coletadas.

O resultado final foi um dataset contendo aproximadamente 2400 issues, classificadas em diferentes tipos de problemas de L10n, como falta de tradução, truncamento de texto, problemas de pontuação, entre outros. A análise dos dados revelou que o tipo de issue mais recorrente foi a tradução incorreta. Com isso, o dataset gerado fornece uma base robusta para futuros estudos e o desenvolvimento de soluções automáticas de detecção de problemas de localização.

**Palavras-chave:** Localization (L10n), Dataset, Issues de localização, Análise de issues

# Abstract

This work aims to create a dataset containing issues related to localization (L10n) problems in open-source projects on GitHub. The main motivation was to provide a data repository to support the development of automated tools for detecting L10n issues. To achieve this, an automated collection tool was developed using the GitHub API to search for issues through specific keywords and labels related to localization, such as "L10n", "localization bug", "localization issue", among others. The collected issues were filtered to ensure the presence of screenshots, a fundamental element for validating the reported problems.

The collected data was organized and stored in a public table on Google BigQuery, allowing direct queries to the structured data. Additionally, an API was developed to facilitate data retrieval, along with a Streamlit application that provides an interactive interface for users to visualize and filter the collected issues.

The final result was a dataset containing approximately 2,400 issues, classified into different types of L10n problems, such as missing translations, text truncation, punctuation issues, among others. Data analysis revealed that the most frequent issue type was mistranslation. Thus, the generated dataset provides a robust foundation for future studies and the development of automated L10n issue detection solutions.

**Keywords:** Localization (L10n), Dataset, Localization issues, Issue analysis

# Sumário


**1 Introdução** **1**
- 1.1 Motivação 1
- 1.2 Definição do problema 1
- 1.3 Objetivo 2
- 1.4 Justificativa 2
- 1.5 Estrutura do Trabalho 3

**2 Fundamentação Teórica** **5**
- 2.1 Introdução à Localization 5
- 2.2 Desafios da Localization em projetos Open Source 6
  - 2.2.1 Colaboração descentralizada 6
  - 2.2.2 Falta de recursos e prioridade 6
  - 2.2.3 Identificação e correção de erros de Localization 6
  - 2.2.4 Manutenção contínua 6

**3 Metodologia** **9**
- 3.1 Coleta de dados 9
  - 3.1.1 Definição de palavras-chave 10
  - 3.1.2 Extração de issues 10
  - 3.1.3 Busca por labels 11
- 3.2 Validação dos Dados Coletados 11
  - 3.2.1 Filtragem de issues relevantes 11
  - 3.2.2 Validação da presença de screenshot 11
- 3.3 Classificação dos Problemas 11
- 3.4 Estruturação e Publicação do Dataset 12
  - 3.4.1 Estrutura do Dataset 12
  - 3.4.2 Publicação e Acesso ao Dataset 13
  - 3.4.3 Inspiração e Validação da Interface do Aplicativo 13

**4 Resultados** **17**
- 4.1 Introdução 17
- 4.2 Distribuição dos Tipos de Issues 17
- 4.3 Principais Tendências Observadas 18
  - 4.3.1 Distribuição dos Tipos de Issues 18
  - 4.3.2 Impacto dos Diferentes Tipos de Issues 19
- 4.4 Conclusão 19

**5 Trabalhos Futuros** **21**
5.1 Desenvolvimento de Ferramentas Automáticas de Detecção de Issues de L10n 21
5.2 Criação de Dados Sintéticos para Ampliação da Base de Dados 22
5.3 Expansão do Dataset 22
5.4 Integração com Outras Bases de Dados 22

**6 Conclusão** **25**

CAPÍTULO 1

# Introdução

## 1.1 Motivação

A globalização e a crescente necessidade de software em várias partes do mundo ressaltam a importância da Localização (Localization - L10n) no desenvolvimento de softwares. Localization é o processo de ajustar um produto, como um aplicativo de software, para corresponder às necessidades e preferências de usuários de diferentes localidades, considerando elementos como idioma, formatos de datas, cultura e outras particularidades regionais.

Nos projetos de software open source, onde a colaboração ocorre de forma distribuída e global, a implementação adequada da Localization é essencial para assegurar que o software seja acessível e eficaz para um público variado. Apesar do aumento da conscientização sobre sua importância, muitos projetos open source ainda enfrentam desafios consideráveis nesta área. Tais desafios geralmente aparecem sob a forma de bugs ou outros problemas de Localization, que podem afetar negativamente a experiência do usuário e, em certos casos, impedir a adoção do software em mercados internacionais.

O GitHub, principal plataforma de hospedagem de código-fonte e colaboração para projetos open source, serve como um recurso valioso para identificar e analisar esses problemas. As issues relatadas nos repositórios do GitHub frequentemente incluem detalhes importantes sobre os bugs de Localization, abrangendo descrições, discussões e, muitas vezes, capturas de tela que demonstram os problemas. Contudo, localizar e analisar esses bugs não é uma tarefa simples, devido ao grande volume de repositórios e issues disponíveis.

## 1.2 Definição do problema

Embora haja uma consciência crescente sobre a relevância da Localization (L10n) no desenvolvimento de softwares, muitos projetos open source ainda enfrentam desafios consideráveis para implementar essa prática de maneira adequada. Esses obstáculos frequentemente resultam em bugs de Localization que afetam a usabilidade e a acessibilidade do software para usuários de diversas regiões e culturas.

Um dos principais desafios para os desenvolvedores e mantenedores de projetos open source é a identificação e correção desses bugs. Com frequência, os problemas de Localization são reportados pelos usuários nas issues dos repositórios no GitHub. Nestes relatos, os usuários descrevem os problemas encontrados e, muitas vezes, incluem capturas de tela que evidenciam esses erros. No entanto, a enorme quantidade de issues e a falta de uma categorização adequada complicam a coleta e análise sistemática dos erros de Localization.

Os bugs podem envolver erros de tradução, uso incorreto de formatos regionais, como datas, números e moedas, além de outros problemas que surgem quando o software é adaptado para diferentes contextos culturais e linguísticos. A falta de um banco de dados centralizado que compile informações visuais sobre esses problemas dificulta o estudo e a resolução dessas questões pela comunidade de desenvolvedores e pesquisadores.

## 1.3 Objetivo

O principal objetivo deste estudo é desenvolver um banco de dados acessível e bem organizado, composto por imagens que documentam bugs de Localization identificados nas issues de repositórios open source no GitHub. Esse banco de dados será construído por meio de uma análise sistemática de issues que apresentem problemas de Localization evidenciados por capturas de tela.

Para alcançar esse objetivo, foi desenvolvida uma ferramenta específica de web scraping para coletar essas imagens das issues no GitHub. Embora a criação dessa ferramenta represente uma etapa crucial do processo, o foco central do trabalho reside na construção e disponibilização desse banco de dados. Esse recurso é projetado para se tornar uma referência valiosa para desenvolvedores, pesquisadores e outros interessados em aprimorar as práticas de Localization em projetos de software.

O banco de dados resultante funcionará como uma plataforma para estudos futuros, possibilitando análises mais detalhadas sobre os tipos mais comuns de erros de Localization, suas causas e possíveis soluções. Além disso, este trabalho contribuirá para aumentar a conscientização sobre a importância da Localization em projetos open source e incentivará a adoção de práticas que melhorem a experiência do usuário em um contexto global.

## 1.4 Justificativa

A criação de um banco de dados centralizado que compila imagens de bugs de Localization representa uma iniciativa inovadora com várias vantagens para a comunidade de desenvolvedores e pesquisadores. Primeiramente, este recurso possibilita uma análise sistemática dos tipos de erros de Localization mais frequentes, facilitando o desenvolvimento de ferramentas automatizadas para a detecção e correção desses problemas.

Além disso, ao disponibilizar esse banco de dados para a comunidade open source, incentiva-se a colaboração na resolução dos problemas identificados. Com acesso a um repositório visual de bugs, os desenvolvedores podem priorizar a correção desses erros com base em evidências concretas, melhorando assim a eficiência e a qualidade das soluções propostas.

Do ponto de vista acadêmico, o banco de dados servirá como uma rica base de dados para futuras pesquisas, permitindo o desenvolvimento de estudos comparativos entre diferentes projetos, a análise de padrões recorrentes em erros de Localization e a avaliação da eficácia das práticas de Localization ao longo do tempo. Este recurso também poderá estimular a criação de novos frameworks ou metodologias que visem aprimorar a adaptação cultural e linguística de softwares em escala global.

Portanto, este trabalho justifica-se não apenas pela inovação que introduz ao disponibilizar um recurso sem precedentes, mas também pelo impacto prático que pode ter na melhoria da experiência do usuário e na promoção de um software verdadeiramente inclusivo e global.

## 1.5 Estrutura do Trabalho

Este trabalho está estruturado em cinco capítulos, além desta introdução. Observe:

- Capítulo 2 - Fundamentação Teórica: Este capítulo apresenta os conceitos fundamentais relacionados à Localization (L10n), abordando suas definições, a importância e os desafios específicos em projetos open source. Também são discutidos trabalhos relacionados e ferramentas existentes que auxiliam na identificação e correção de erros de Localization.

- Capítulo 3 - Metodologia: Aqui, detalhamos a metodologia adotada para a coleta e análise dos dados. O capítulo descreve o processo de desenvolvimento do web scraper e as técnicas utilizadas para identificar e extrair as imagens de bugs de Localization das issues dos repositórios no GitHub, além de explicar os critérios para seleção e categorização das imagens.

- Capítulo 4 - Desenvolvimento e Implementação: Este capítulo foca na construção do banco de dados. São explicadas as etapas de organização, armazenamento e disponibilização das imagens coletadas, além dos desafios enfrentados durante a implementação e as soluções encontradas. Além disso, o capítulo aborda, de forma secundária, o desenvolvimento da API e da interface criada para facilitar a consulta e o acesso aos dados, destacando sua integração com o banco de dados.

- Capítulo 5 - Análise dos Resultados: Apresentamos uma análise preliminar dos dados coletados, destacando padrões e tendências observadas nos bugs de Localization. Este capítulo também explora as implicações dos resultados para a prática de Localization em projetos open source e sugere possíveis áreas para estudos futuros.

- Capítulo 6 - Conclusão: O capítulo final resume os principais pontos abordados ao longo do trabalho, reforçando a contribuição do banco de dados criado e sugerindo direções para futuras pesquisas que possam expandir ou aprofundar este estudo.

CAPÍTULO 2

# Fundamentação Teórica

## 2.1 Introdução à Localization

A Localization (L10n) é um processo crucial no desenvolvimento de softwares destinados a mercados globais. Ela envolve a adaptação de um produto, como um software, para atender às especificidades de um público-alvo, considerando elementos culturais, linguísticos e regionais. Diferente da Internacionalização (i18n), que prepara o software para suportar múltiplos idiomas e formatos regionais, a Localization se concentra na personalização detalhada do produto para um mercado específico, assegurando que ele seja pertinente e acessível para os usuários locais.

A Localization vai além da mera tradução de textos, como descrito por Santini [4]. Apesar de a tradução ser um componente vital, a L10n engloba uma série de ajustes, incluindo a adaptação de unidades de medida, formatos de data e hora, números, moedas, e até símbolos e cores que podem possuir significados distintos em diferentes culturas. Por exemplo, enquanto a cor branca pode ser associada à pureza em muitas culturas ocidentais, em algumas culturas asiáticas, ela é ligada ao luto. Desconsiderar esses fatores pode levar a uma experiência de usuário confusa ou ofensiva.

Um aspecto fundamental da Localization é a adaptação do layout e da interface do usuário (UI). Em línguas como o árabe e o hebraico, que são lidas da direita para a esquerda, toda a interface precisa ser espelhada para adequar-se à direção de leitura. Além disso, a localização de botões, menus e outros elementos interativos pode necessitar ajustes para preservar a usabilidade e a fluidez na navegação.

A importância da Localization no desenvolvimento de softwares globais é indiscutível. Um produto que não é adequadamente localizado pode enfrentar barreiras significativas de adoção em novos mercados. Por exemplo, se um aplicativo financeiro não exibir os formatos de moeda ou os números conforme os padrões locais, os usuários podem perceber o produto como confuso ou não confiável, prejudicando a confiança no software e na marca.

Além disso, a L10n tem um impacto direto na percepção da qualidade do software. Usuários que veem um produto como sendo cuidadosamente adaptado para o seu contexto local tendem a valorizá-lo mais, sentindo-se respeitados e atendidos por uma empresa que compreende suas necessidades específicas. Por outro lado, falhas na Localization, como traduções errôneas ou inadequadas, podem resultar em frustração e até no abandono do produto.

Portanto, a Localization é um elemento essencial para o sucesso de softwares em um mercado global, garantindo que os produtos sejam não apenas compreendidos, mas também valorizados e adotados por usuários de diversas regiões. Em um mercado de software competitivo, a habilidade de adaptar produtos eficazmente para diferentes mercados pode ser um diferencial crucial para o sucesso internacional.

## 2.2 Desafios da Localization em projetos Open Source

A implementação eficaz da Localization (L10n) em projetos de software já é um desafio significativo em contextos tradicionais de desenvolvimento. Nos projetos open source, esses desafios são ainda mais acentuados devido às características intrínsecas ao modelo de desenvolvimento colaborativo e descentralizado.

### 2.2.1 Colaboração descentralizada

Os projetos open source são caracterizados por equipes de desenvolvimento distribuídas, muitas vezes compostas por voluntários de diferentes partes do mundo. Esta distribuição geográfica e cultural pode dificultar a consistência e a qualidade da Localization. Os contribuidores podem ter diferentes níveis de familiaridade com as práticas de L10n, o que pode resultar em inconsistências na tradução e na adaptação cultural. Além disso, a comunicação entre os membros da equipe pode ser limitada, o que leva a mal-entendidos ou à falta de coordenação na implementação das práticas de Localization.

Um exemplo claro é o uso de humor ou expressões idiomáticas na interface do software. Uma frase engraçada ou informal que funciona bem em inglês pode não ter o mesmo impacto, ou pode até ser mal interpretada, quando traduzida para outra língua. Isso pode criar barreiras de usabilidade e tornar a interface menos acessível para usuários de diferentes culturas.

### 2.2.2 Falta de recursos e prioridade

Em muitos projetos open source, a Localization não é vista como uma prioridade, especialmente nos estágios iniciais de desenvolvimento. Isso ocorre, em parte, pela escassez de recursos, tanto em termos de tempo quanto de pessoal. Muitos desenvolvedores em projetos open source são voluntários que contribuem em seu tempo livre, e pode não haver especialistas dedicados em Localization na equipe. Como resultado, a L10n pode ser adiada ou tratada de forma superficial, o que aumenta a probabilidade de erros e inconsistências.

### 2.2.3 Identificação e correção de erros de Localization

Outro desafio significativo é a identificação e correção de erros de Localization em projetos open source. Esses erros podem variar desde traduções incorretas até problemas mais sutis, como o uso inadequado de formatos regionais ou a falta de suporte para scripts de escrita da direita para a esquerda. No GitHub, esses problemas são frequentemente reportados como issues pelos usuários, mas a triagem e correção desses erros podem ser complicadas pela grande quantidade de issues abertas e pela falta de ferramentas automatizadas para identificar problemas específicos de Localization.

### 2.2.4 Manutenção contínua

A manutenção contínua da Localization é um desafio considerável em projetos open source. Conforme o software se desenvolve e novas funcionalidades são incorporadas, é necessário atu-

alizar as traduções e outras adaptações culturais correspondentes. Em ambientes open source, onde as atualizações podem ocorrer de forma frequente e rápida, assegurar que todas as alterações sejam adequadamente refletidas nas localizações se torna uma tarefa complicada, especialmente na ausência de uma infraestrutura robusta para gerenciar o processo de L10n.

CAPÍTULO 3

# Metodologia

Este trabalho tem como objetivo a construção de um banco de dados robusto contendo capturas de tela de problemas de Localization (L10n) em repositórios open source disponíveis no GitHub. O dataset resultante será utilizado como base para o desenvolvimento de ferramentas automatizadas que possam detectar problemas de L10n em sistemas de software. A ideia central é que, com o avanço em técnicas de aprendizado de máquina e processamento de imagem, esse dataset possa ser utilizado por futuros pesquisadores ou engenheiros de software para treinar modelos capazes de identificar automaticamente erros de L10n em interfaces de usuário.

O critério básico para a inclusão de uma issue no dataset foi a presença de um problema claramente relacionado à L10n, juntamente com uma captura de tela que ilustrasse o erro. Dessa forma, as issues que não apresentassem evidências visuais ou que não estivessem relacionadas a L10n foram excluídas.

O processo de construção do dataset envolveu quatro etapas distintas, inspiradas na metodologia de construção de datasets para aprendizado de máquinas descrito por Silva e Santos [5], as quais serão detalhadas a seguir. Cada uma dessas etapas foi implementada com o objetivo de garantir que o dataset fosse o mais completo e detalhado possível, oferecendo uma base sólida para futuras pesquisas e desenvolvimentos de ferramentas automáticas de detecção de erros de L10n.

Agora, vamos detalhar cada uma dessas etapas.

## 3.1 Coleta de dados

A coleta de dados foi a primeira etapa no processo de construção do dataset de problemas de Localization (L10n). Para isso, foi implementada uma ferramenta automatizada, inspirada na ferramenta PyDriller [6], desenvolvida em Python, capaz de rastrear repositórios open source no GitHub que contivessem issues relacionadas a problemas de L10n. A estratégia de coleta baseou-se no uso de palavras-chave específicas, como "Localization", "L10n", "Translation", entre outras, para identificar as issues relevantes nos repositórios.

Para o desenvolvimento deste trabalho, foram utilizadas diversas ferramentas e bibliotecas de Python, com o objetivo de automatizar o processo de coleta, validação e organização dos dados de issues relacionados a problemas de L10n:

- **Pandas**: Utilizado para a criação e manipulação de dataframes com os dados extraídos das issues do GitHub, permitindo a organização e análise das informações coletadas.
- **PyDrive**: Empregado para fazer a transição dos arquivos de captura de tela (prints) ar-

mazenados localmente para pastas específicas no Google Drive.

- **GSpread**: Utilizado para hospedar e publicar os dados finais das issues em uma planilha do Google Sheets, permitindo um acesso e visualização mais conveniente dos resultados.
- **Requests**: Biblioteca usada para se conectar com a API do GitHub, fazendo buscas automáticas por issues relacionadas a L10n e recuperando os dados dessas issues.

O processo de coleta consistiu em:

### 3.1.1 Definição de palavras-chave

Para garantir a precisão da coleta, foram selecionadas as seguintes palavras-chave diretamente relacionadas a problemas de L10n:

- L10n
- Localization
- localization bug
- localization issue
- L10n bug
- L10n issue

Essas palavras-chave foram usadas para realizar buscas automáticas dentro das issues dos repositórios no GitHub.

### 3.1.2 Extração de issues

A partir das palavras-chave, a ferramenta buscou por issues que mencionassem problemas de L10n. Cada issue extraída continha as seguintes informações:

- **Nome do repositório:** O repositório de onde a issue foi extraída.
- **Link da issue:** URL direto para a issue no GitHub.
- **Número da issue:** Identificação única da issue dentro do repositório.
- **Título:** Título da issue no Github.
- **Descrição:** Descrição da issue no Github.
- **Data de abertura:** Data de abertura da issue no Github.
- **Data de fechamento:** Data de fechamento da issue no Github.

### 3.1.3 Busca por labels

Além da busca por palavras-chave, foi implementada uma segunda abordagem baseada na busca por labels. Essa abordagem consistiu no uso da API do GitHub para pesquisar repositórios que utilizassem labels relacionadas a problemas de L10n.

A consulta foi realizada com o seguinte formato:

```
labels:<palavra-chave>
```

Foram utilizadas as mesmas palavras-chave da busca textual para filtrar labels, o que permitiu identificar repositórios que já categorizavam issues de L10n de maneira estruturada. Dentro desses repositórios, a ferramenta automatizada analisou todas as issues associadas às labels identificadas, buscando capturas de tela e metadados relevantes.

Essa abordagem complementou a pesquisa por palavras-chave, aumentando a abrangência da coleta, uma vez que muitas issues podem estar corretamente rotuladas sem necessariamente conter os termos exatos no título.

## 3.2 Validação dos Dados Coletados

### 3.2.1 Filtragem de issues relevantes

Após a extração inicial, a ferramenta realizou uma verificação adicional para garantir que cada issue realmente se referisse a problemas de L10n. Esse processo incluiu análise textual da descrição da issue e validação manual das issues que continham screenshots ilustrando o problema relatado.

### 3.2.2 Validação da presença de screenshot

O critério fundamental para inclusão de uma issue no dataset foi a presença de uma captura de tela que demonstrasse o problema. A ferramenta verificou automaticamente se a issue continha um link para uma imagem ou anexo visual. Issues sem screenshots foram descartadas, o que resultou numa base inicial de cerca de 5000 issues.

## 3.3 Classificação dos Problemas

Uma vez validadas, as issues foram classificadas com base no tipo de problema de L10n apresentado. A classificação foi essencial para organizar o dataset de forma estruturada, facilitando o uso futuro para treinamento de modelos e análises automáticas. Os principais tipos de problemas identificados incluíram:

- **Missing translation**: Ausência de tradução em partes da interface, com partes do texto exibidas no idioma original.
- **Truncation:** Textos cortados ou truncados na interface, impedindo a visualização completa.

- **Ellipsis:** Problemas com o uso de reticências, geralmente relacionados ao corte inadequado de textos.
- **Display issue:** Problemas visuais na exibição do texto, como alinhamento ou espaçamento incorreto.
- **Ponctuation issue:** Erros de pontuação em textos traduzidos.
- **Date issue**: Problemas relacionados à formatação de datas, como formato incorreto para a localidade.
- **Capitalization:** Uso inadequado de letras maiúsculas ou minúsculas em traduções.
- **Mistranslation:** Traduções incorretas ou que alteram o significado do texto original.
- **Text balance:** Desequilíbrio no layout ou na distribuição do texto em diferentes partes da interface.
- **Special character:** Problemas relacionados ao uso de caracteres especiais, como acentos ou símbolos que não são corretamente renderizados.

## 3.4 Estruturação e Publicação do Dataset

### 3.4.1 Estrutura do Dataset

A última etapa envolveu a organização dos dados coletados e a sua disponibilização para uso futuro. O dataset foi estruturado em uma tabela no Google BigQuery [1] contendo as seguintes informações para cada issue:

- **github project:** Projeto do Github o qual a issue pertence.
- **github issue link:** Link direto para a issue do Github.
- **github issue id:** ID da issue no Github.
- **id:** ID único para identificação da issue na base de dados.
- **issue title:** Título da issue no Github.
- **opened at:** Data na qual a issue foi aberta no Github.
- **closed at:** Data na qual a issue foi fechada no Github.
- **issue type:** Tipo de problema L10n idenficado no issue.
- **description:** Descrição do problema L10n identificado na issue.
- **screenshot link:** Link direto para a screenshot do problema L10n idenficado na issue.
- **locale:** Locale do problema L10n identificado na issue.
- **domain:** Domínio no qual o problema L10n é reproduzido.

### 3.4.2 Publicação e Acesso ao Dataset

Para tornar o dataset acessível a diferentes públicos, foi desenvolvido um aplicativo utilizando a tecnologia **Streamlit** [7], que fornece uma interface visual intuitiva para consulta dos dados. Esse aplicativo permite que os usuários filtrem as issues por critérios como tipo de problema, idioma e domínio do sistema impactado. Os resultados são apresentados de forma interativa, exibindo os detalhes da issue e suas respectivas capturas de tela.

Além da interface visual, foi desenvolvida uma **API** que se conecta diretamente ao BigQuery para fornecer acesso programático ao dataset. A API permite que pesquisadores e desenvolvedores realizem consultas estruturadas e obtenham os dados de forma dinâmica, possibilitando integrações com outras ferramentas e aplicações.

A combinação dessas tecnologias torna o dataset não apenas uma coleção de dados estática, mas um recurso interativo e acessível para futuras pesquisas, análises e desenvolvimento de ferramentas automatizadas de detecção de erros de L10n.

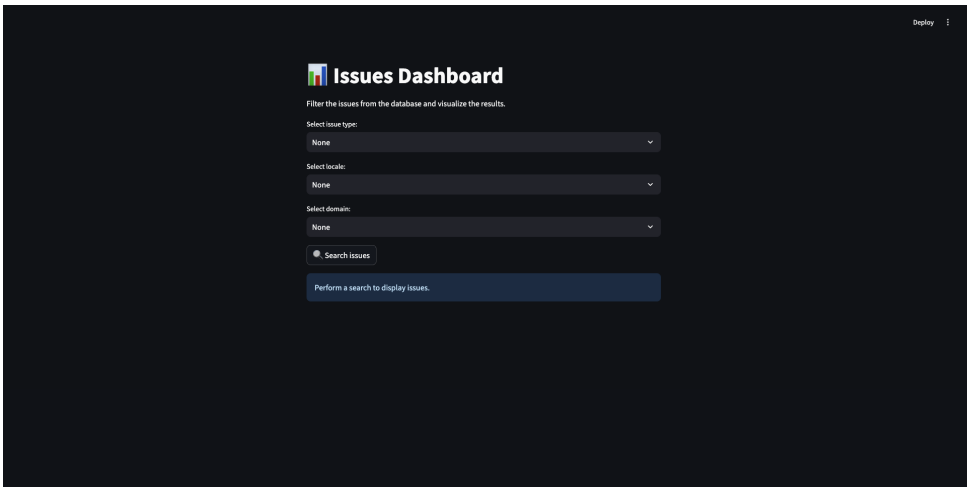


Figura 3.1: Interface do aplicativo Streamlit.

### 3.4.3 Inspiração e Validação da Interface do Aplicativo

A interface do aplicativo desenvolvido com Streamlit foi pensada para ser simples, intuitiva e de fácil uso, mesmo para usuários sem familiaridade técnica. A inspiração para o design partiu da análise de aplicações com estruturas semelhantes, que apresentavam três elementos principais:

- **Menus suspensos (dropdowns):** utilizados para filtrar os dados por tipo de issue, locale e domínio, permitindo ao usuário explorar o dataset com flexibilidade.

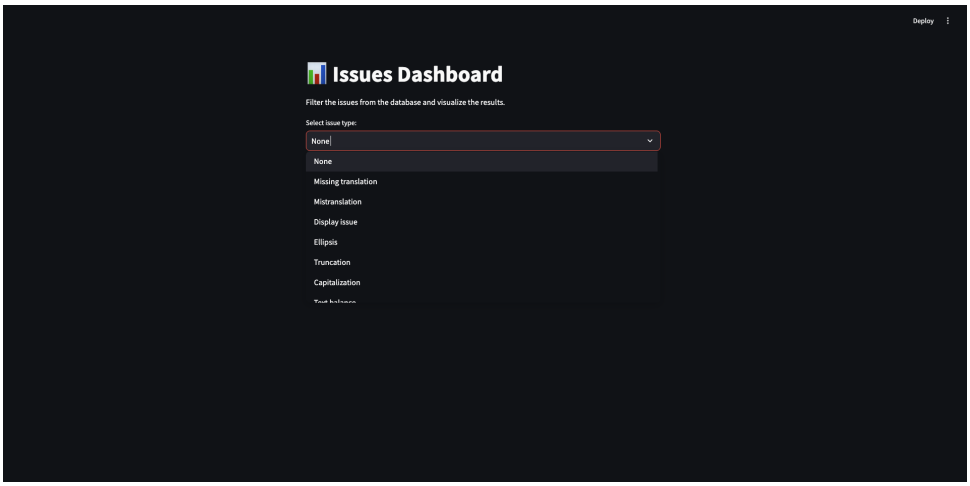


Figura 3.2: Filtro de tipos de issue.

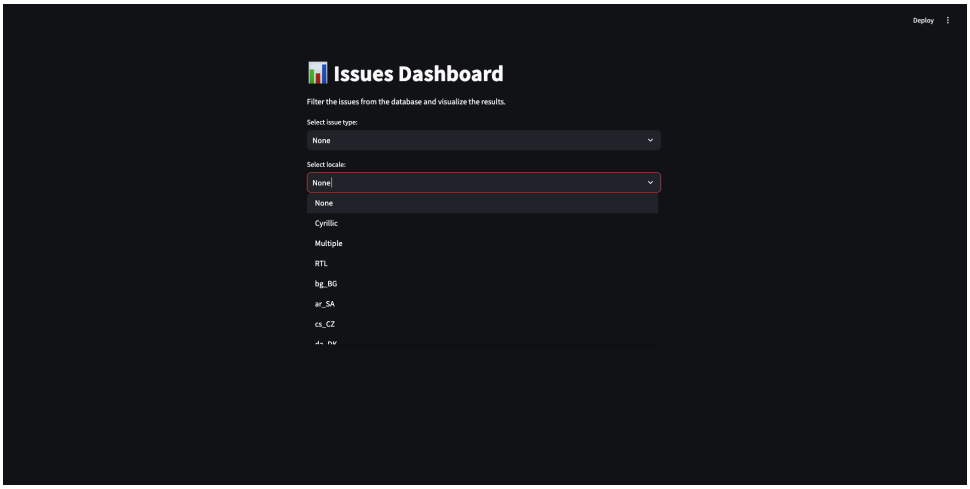


Figura 3.3: Filtro de locale da issue.

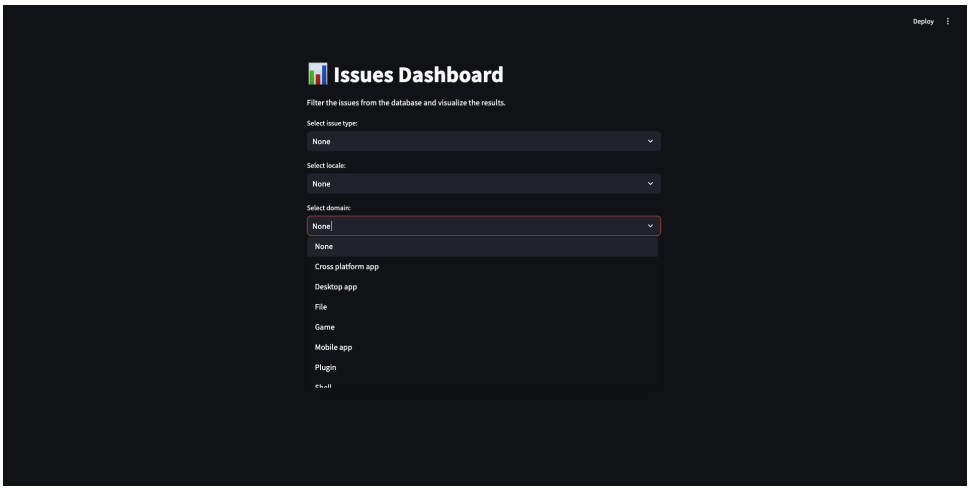


Figura 3.4: Filtro de domínio da issue.

- **Tabela interativa:** responsável por exibir os resultados filtrados, facilitando a navegação e análise das informações.

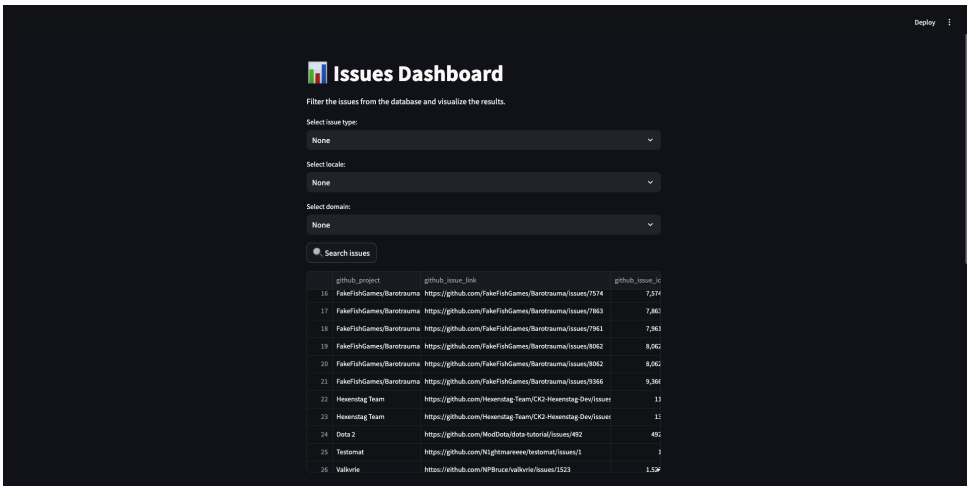


Figura 3.5: Tabela com as issues retornadas.

- **Carrossel de imagens:** para visualização dos screenshots relacionados às issues, tornando a identificação de erros de L10n mais acessível e visual.

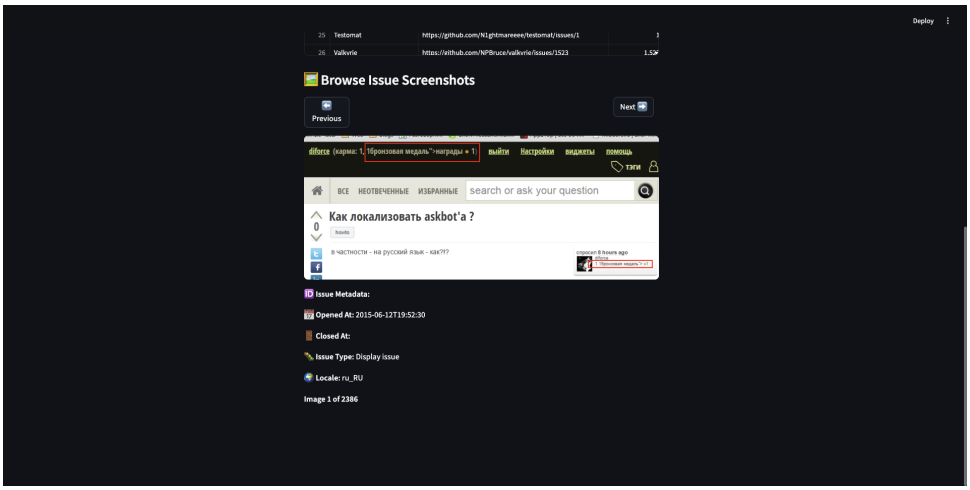


Figura 3.6: Carossel de screenshots das issues.

A validação da interface foi realizada de maneira informal, por meio de testes com familiares e amigos. A usabilidade foi considerada satisfatória, com os participantes conseguindo navegar e interagir com o aplicativo sem dificuldades, confirmando que a proposta de uma interface simples e funcional foi bem-sucedida.

CAPÍTULO 4

# Resultados

## 4.1 Introdução

Neste capítulo, são apresentados os resultados obtidos a partir da coleta e validação dos dados de issues relacionados a problemas de Localization (L10n) em repositórios do GitHub. As informações estão organizadas em tabelas e gráficos que facilitam a visualização da distribuição e frequência dos tipos de problemas encontrados. Além disso, serão discutidas as principais conclusões e interpretações baseadas nos dados coletados.

## 4.2 Distribuição dos Tipos de Issues

Após a classificação dos problemas, foi possível identificar e quantificar os diferentes tipos de issues relacionadas a L10n. A tabela a seguir mostra a distribuição dos principais tipos de problemas encontrados:

| **Tipo de Issue** | **Quantidade** |
|---|---|
| Missing Translation | 891 |
| Truncation | 101 |
| Ellipsis | 30 |
| Display Issue | 204 |
| Punctuation Issue | 3 |
| Date Issue | 18 |
| Capitalization | 85 |
| Mistranslation | 938 |
| Text Balance | 53 |
| Special Character | 63 |
| **Total de Issues Analisadas** | **2386** |

Tabela 4.1: Distribuição dos tipos de issues de L10n coletadas.

Esta tabela revela a frequência de cada tipo de problema de L10n encontrado nas issues analizadas. Além da tabela, a visualização gráfica desses dados permite uma compreensão mais intuitiva das proporções entre os diferentes tipos de issues. A Figura 1 apresenta um gráfico de barras comparando a quantidade de cada tipo de problema:

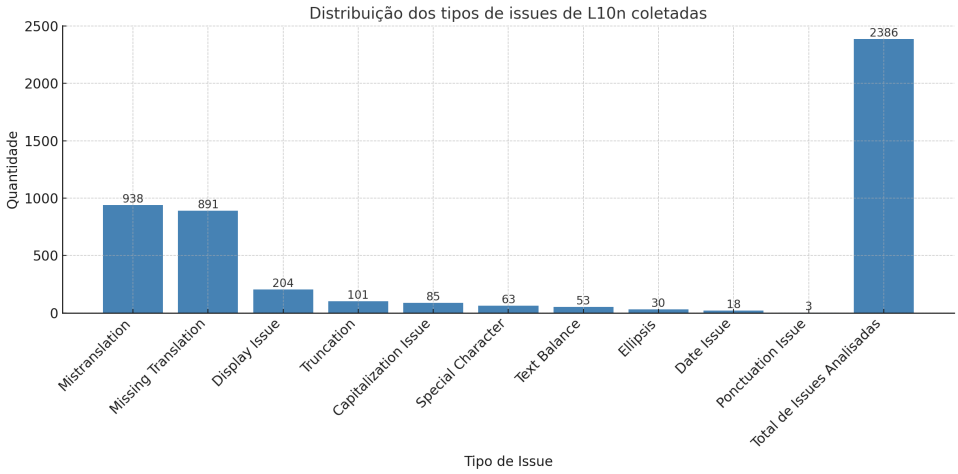


Figura 4.1: Distribuição gráfica dos tipos de issues de L10n.

Este gráfico destaca os tipos de issues que foram mais recorrentes durante o processo de coleta e validação, permitindo identificar áreas com maiores incidências de problemas.

## 4.3 Principais Tendências Observadas

Com base nos dados coletados, foram observadas algumas tendências significativas:

### 4.3.1 Distribuição dos Tipos de Issues

A análise dos dados coletados revelou que o tipo de issue mais recorrente no conjunto é **Mistranslation**, com 938 ocorrências, seguido de perto por **Missing Translation**, com 891 registros. Juntas, essas categorias representam a maior parte dos problemas reportados, evidenciando que a qualidade das traduções continua sendo um dos principais desafios enfrentados pelos projetos de software open source. A ausência de traduções ou presença de traduções erradas compromete diretamente a experiência do usuário, podendo gerar confusão e dificultar o entendimento da interface.

Além dessas categorias dominantes, outros tipos de problemas também foram identificados em quantidades relevantes. **Display Issue** (204 ocorrências) e **Truncation** (101 ocorrências) são indicativos de que a apresentação visual da interface ainda enfrenta desafios significativos. Muitos sistemas não são projetados para acomodar variações linguísticas, o que pode resultar em elementos sobrepostos, textos cortados ou desalinhados, prejudicando a usabilidade da aplicação.

Outras categorias de problemas, como **Capitalization** (85 ocorrências), **Special Character** (63 ocorrências) e **Text Balance** (53 ocorrências), demonstram que detalhes aparentemente menores também podem impactar a percepção do usuário. Erros de capitalização e o uso inadequado de caracteres especiais podem transmitir uma sensação de falta de profissionalismo e reduzir a confiabilidade do software.

Por outro lado, **Ponctuation Issue** (3 ocorrências) e **Date Issue** (18 ocorrências) foram

observados com menor frequência, sugerindo que esses aspectos são menos problemáticos ou menos frequentemente reportados pelos usuários. No entanto, mesmo que sejam menos prevalentes, esses problemas podem afetar a clareza da informação, especialmente em aplicações que dependem fortemente de comunicação textual.

### 4.3.2 Impacto dos Diferentes Tipos de Issues

Cada tipo de issue identificado apresenta um impacto diferente na experiência do usuário. Issues de **Missing Translation** e **Mistranslation** têm consequências diretas, pois podem dificultar a compreensão da interface e comprometer a acessibilidade do software em diferentes idiomas. Um usuário que encontra partes não traduzidas ou mal traduzidas pode ter dificuldades para interagir com a aplicação, o que pode gerar frustração e até levar ao abandono da ferramenta.

Os problemas de **Display Issue** e **Truncation** afetam principalmente a apresentação visual do conteúdo. Textos cortados ou desalinhados prejudicam a legibilidade e podem comprometer a experiência do usuário, tornando a navegação menos fluida. Isso é especialmente relevante em aplicativos que precisam exibir grandes quantidades de informação ou que fazem uso intensivo de botões e menus interativos.

Erros de **Capitalization** e **Special Character** podem parecer menos críticos, mas afetam a percepção de qualidade da interface. Um sistema que apresenta inconsistências nessas áreas pode ser interpretado como menos profissional ou menos confiável pelos usuários. Da mesma forma, problemas com **Text Balance** podem resultar em interfaces desorganizadas, onde alguns elementos textuais ganham mais destaque do que deveriam.

Embora menos frequentes, os problemas de **Ponctuation Issue** e **Date Issue** podem gerar confusão, especialmente em sistemas que lidam com informações sensíveis, como dados financeiros ou registros de transações. Formatos de data inconsistentes podem causar erros na interpretação de prazos, horários e eventos, impactando diretamente a usabilidade do software.

## 4.4 Conclusão

A análise dos dados coletados revelou uma distribuição significativa entre os tipos de problemas de L10n encontrados nas issues do GitHub. A predominância de problemas como **Mistranslation** e **Missing Translation** indica que ainda há grandes desafios a serem enfrentados na localização de software, enquanto os problemas de exibição e truncamento reforçam a necessidade de melhorias no design adaptativo das interfaces.

CAPÍTULO 5

# Trabalhos Futuros

Este capítulo delineia várias direções para o uso e a expansão do dataset gerado, visando auxiliar pesquisadores e desenvolvedores em seus trabalhos futuros. O dataset de issues de L10n não apenas serve como base de estudo para a detecção de falhas de localização, mas também pode ser utilizado em diversas outras aplicações que se beneficiam de uma base de dados robusta. Além disso, são destacadas as possibilidades de expansão e melhoria contínua do dataset.

## 5.1 Desenvolvimento de Ferramentas Automáticas de Detecção de Issues de L10n

A principal aplicação do dataset está relacionada ao desenvolvimento de ferramentas automáticas de detecção de problemas de localização. Com a complexidade crescente das interfaces de software globalizadas, ferramentas que utilizam aprendizado de máquina e processamento de linguagem natural podem se beneficiar significativamente dos dados fornecidos por este projeto. O dataset oferece uma variedade de exemplos de erros de L10n, ideais para treinar algoritmos que identifiquem automaticamente falhas, como traduções ausentes, truncamento de texto e problemas de formatação.

Trabalhos anteriores, como Infrared [5], já demonstraram a viabilidade da aplicação de técnicas automatizadas para identificação de problemas em software. Com a disponibilização deste dataset, espera-se contribuir para o avanço na detecção de problemas de L10n de maneira escalável e automatizada.

Ferramentas futuras poderiam:

- Automatizar a análise de novas versões de software, identificando issues de L10n antes do lançamento.
- Ajudar a priorizar correções, com base na gravidade e na frequência dos problemas detectados.
- Integrar-se a pipelines de CI/CD (Integração Contínua e Entrega Contínua) para monitoramento automático.

## 5.2 Criação de Dados Sintéticos para Ampliação da Base de Dados

A geração de dados sintéticos representa uma estratégia eficaz para expandir a base de dados de problemas de L10n, permitindo a criação de cenários variados que simulam issues reais observadas nos repositórios analisados. Esses dados podem ser utilizados para aprimorar ferramentas automatizadas e modelos de aprendizado de máquina, tornando-os mais robustos e adaptáveis a diferentes contextos.

A partir das características dos problemas identificados, é possível gerar automaticamente novos exemplos que reflitam os desafios comuns enfrentados no desenvolvimento de software localizado. Esses dados sintéticos podem ser empregados para:

- Ampliar a diversidade dos exemplos disponíveis para o treinamento de modelos de aprendizado profundo, melhorando sua capacidade de generalização e identificação de problemas de L10n.
- Avaliar a robustez de ferramentas de detecção automática ao expô-las a diferentes variações de erros de L10n, possibilitando a identificação de padrões recorrentes e a antecipação de potenciais problemas em novos projetos.

A utilização de dados sintéticos permite não apenas o aumento do volume de exemplos disponíveis, mas também o desenvolvimento de estratégias mais eficazes para a detecção e correção de problemas de L10n. Essa abordagem contribui significativamente para o aprimoramento das soluções automatizadas, promovendo a evolução contínua da qualidade da localização de software.

## 5.3 Expansão do Dataset

A expansão do dataset é outra direção crucial para trabalhos futuros. Atualmente, o dataset está limitado a um conjunto de repositórios open source no GitHub, mas pode ser expandido de várias maneiras:

- **Incluir outros repositórios:** Ampliar o escopo para incluir repositórios privados, projetos de software comercial e outras plataformas como GitLab e Bitbucket.
- **Explorar diferentes linguagens:** Expandir a cobertura para mais idiomas e regiões, aumentando a diversidade dos problemas de L10n capturados.
- **Coleta contínua:** Implementar uma coleta contínua de issues, monitorando repositórios open source constantemente e adicionando novos problemas ao dataset conforme surgirem.

## 5.4 Integração com Outras Bases de Dados

Integrar o dataset de L10n com outras bases de dados relacionadas a problemas de software ou localização específica também é uma possibilidade:

- **Enriquecer o contexto:** Combinar os dados de L10n com issues funcionais para entender melhor como os problemas de localização podem estar relacionados a outras falhas no sistema.
- **Construção de modelos mais complexos:** Desenvolver modelos preditivos que considerem múltiplas variáveis e tipos de erros de software, prevendo falhas futuras com base em padrões históricos.

CAPÍTULO 6

# Conclusão

Este trabalho apresentou a criação de um dataset contendo issues de localização (L10n) extraídas de projetos open source no GitHub. O objetivo principal foi construir uma base de dados estruturada, acessível e útil para pesquisadores e desenvolvedores interessados na identificação e análise de problemas de localização em software. Para isso, foram utilizadas abordagens automatizadas para coletar e validar issues relevantes, garantindo que cada entrada no dataset estivesse acompanhada de evidências visuais do problema relatado.

O processo de coleta e estruturação dos dados resultou em um dataset público armazenado no Google BigQuery, permitindo consultas eficientes e integradas com outras ferramentas analíticas. Além disso, foi desenvolvida uma API que facilita o acesso programático ao dataset, bem como um aplicativo interativo baseado em Streamlit, que fornece uma interface intuitiva para a visualização e exploração dos dados. Essas soluções ampliam significativamente a usabilidade do dataset, tornando-o acessível a diferentes públicos, desde pesquisadores acadêmicos até desenvolvedores que buscam melhorar a qualidade da localização de seus softwares.

As contribuições deste trabalho vão além da construção do dataset em si. A metodologia implementada pode ser reutilizada e expandida para novas coletas de dados, garantindo a evolução contínua do dataset. Além disso, a integração com BigQuery e a disponibilização da API abrem espaço para futuras aplicações, como o desenvolvimento de modelos de aprendizado de máquina para a detecção automatizada de problemas de L10n.

Como trabalhos futuros, propõe-se a expansão do dataset para incluir issues de outras plataformas além do GitHub, aumentando sua abrangência. Além disso, a criação de dados sintéticos pode ser explorada como uma estratégia para aumentar a diversidade e representatividade dos exemplos disponíveis, aprimorando ainda mais o treinamento de ferramentas automatizadas.

O código desenvolvido para a coleta e processamento dos dados está disponível no repositório GitHub, e o dataset pode ser acessado publicamente tanto através do BigQuery quanto através do app Streamlit.

Dessa forma, este trabalho representa um passo significativo para a sistematização da análise de problemas de L10n, fornecendo um recurso valioso para a comunidade de desenvolvedores e pesquisadores interessados na melhoria da qualidade da localização de software.

# Referências Bibliográficas


[1] Google Cloud. Bigquery documentation - official guide. https://cloud.google.com/bigquery/docs, 2024.

[2] GitHub. Rest api endpoints for issues - documentação oficial. https://docs.github.com/en/rest/issues/issues, 2024.

[3] Mathieu Nayrolles et al. Infrared: A meta bug detector. https://arxiv.org/abs/2209.08510, 2018.

[4] Silvia Santini. Semiotic internationalization and localization of computer programs. https://arxiv.org/abs/1805.04342, 2018.

[5] João Silva and Maria Santos. Roteiro para criação de dataset de imagens para modelos de aprendizagem profunda. https://eailab.labmax.org/2024/09/24/roteiro-para-criacao-de-dataset-de-imagens-para-modelos-de-aprendizage 2020.

[6] Davide Spadini et al. Pydriller: Python framework for mining software repositories. https://dl.acm.org/doi/10.1145/3236024.3264598, 2018.

[7] Streamlit. Streamlit documentation. https://docs.streamlit.io/, 2024.