PONTIFÍCIA UNIVERSIDADE CATÓLICA DE MINAS GERAIS
Instituto de Ciências Econômicas e Gerenciais

Enzo Barcelos Rios Ferreira
Igor Miranda Santos
João Paulo de Sales Pimenta

CARACTERÍSTICAS DE REPOSITÓRIOS POPULARES NO GITHUB:

Uma investigação sobre maturidade, colaboração, frequência de atualização e uso de linguagens em projetos open-source

Belo Horizonte
2025

SUMÁRIO

1 INTRODUÇÃO

O desenvolvimento de software *open-source* tem desempenhado um papel fundamental na evolução da tecnologia, permitindo a colaboração entre desenvolvedores e empresas ao redor do mundo. Neste relatório, analisamos os 1.000 repositórios mais populares do *GitHub*, com base no número de estrelas, para entender melhor suas características e padrões de desenvolvimento.

A pesquisa busca responder perguntas como: repositórios populares tendem a ser mais antigos e maduros? Eles recebem muitas contribuições externas? São frequentemente atualizados e lançam novas versões com regularidade? Além disso, investigamos se esses projetos utilizam as linguagens de programação mais populares e qual a taxa de fechamento de *issues*.

Para responder a essas questões, realizamos a coleta de dados via *GraphQL* e aplicamos métricas específicas para cada aspecto analisado. Os resultados permitem compreender melhor como esses projetos evoluem e quais fatores podem influenciar sua popularidade e manutenção ao longo do tempo.

1.1 Objetivos

1.1.1 Objetivo geral

Analisar as características dos repositórios *open-source* mais populares do *GitHub* para identificar padrões relacionados à sua maturidade, contribuição externa, frequência de atualização e uso de linguagens de programação.

1.1.2 Objetivos específicos

- Determinar a idade média dos repositórios populares e avaliar se projetos mais antigos tendem a ser mais bem avaliados.
- Analisar a quantidade de contribuições externas recebidas por esses repositórios, considerando o número de *pull requests* aceitas.
- Verificar a frequência com que esses repositórios lançam novas versões (*releases*).
- Avaliar a regularidade das atualizações, analisando o tempo decorrido desde a última modificação.
- Identificar as linguagens de programação mais utilizadas nos repositórios populares.

- Examinar a taxa de fechamento de *issues* para entender a manutenção e gerenciamento dos projetos.
- Comparar como essas características variam de acordo com a linguagem de programação utilizada nos repositórios.

2 METODOLOGIA

2.1 Coleta de Dados

A coleta de dados foi realizada por meio da **API do GitHub**, utilizando consultas *GraphQL* para obter informações detalhadas sobre repositórios populares. A query *GraphQL* foi construída para buscar os 10 repositórios mais populares por vez, com paginação para coletar dados de 1000 repositórios. Os campos incluídos na query foram:

- **name**: Nome do repositório.
- **createdAt**: Data de criação.
- **pullRequests**: Número de pull requests aceitas.
- **releases**: Número de releases.
- **updatedAt**: Data da última atualização.
- **primaryLanguage**: Linguagem primária.
- **issues**: Número de issues abertas e fechadas.

A paginação foi implementada utilizando o cursor (**after_cursor**) para buscar os próximos conjuntos de repositórios até atingir o total de 1000 repositórios. A automatização da coleta foi realizada por meio de scripts em **Python**, utilizando a biblioteca *requests* para fazer chamadas à API. Os dados brutos foram armazenados em um arquivo CSV (resultados.csv).

2.2 Processamento de Dados

Os dados coletados foram processados utilizando a biblioteca **Pandas** em Python. As etapas de processamento incluíram:

- **Conversão de Datas**: As colunas de datas (*createdAt* e *updatedAt*) foram convertidas para o formato *datetime* para facilitar cálculos.
- **Cálculo de Métricas**:
    - **Idade do Repositório**: Calculada como a diferença entre a data atual e a data de criação, em anos.
    - **Dias desde a Última Atualização**: Calculada como a diferença entre a data atual e a data da última atualização, em dias.
    - **Taxa de Fechamento de *Issues***: Calculada como a razão entre o número de *issues* fechadas e o total de *issues* (fechadas + abertas).

Os dados processados foram armazenados em um novo arquivo CSV (resultados_processados.csv), contendo as métricas calculadas.

2.3 Análise de Dados

Antes da análise, os dados passaram por uma etapa de pré-processamento para garantir a qualidade e a consistência. As seguintes ações foram realizadas:

- **Limpeza de dados:** Remoção de registros incompletos ou duplicados.
- **Transformação de dados:** Conversão de colunas para tipos de dados adequados (e.g., datas para o formato *datetime*).
- **Normalização:** Padronização de nomes de colunas e valores para facilitar a análise.

A análise descritiva foi realizada para resumir e descrever as características principais dos dados. Foram utilizadas as seguintes métricas e técnicas:

- **Medidas de tendência central:** Cálculo da mediana para variáveis numéricas, como idade do repositório, número de *pull requests* aceitas, total de *releases*, dias desde a última atualização e taxa de fechamento de *issues*.
- **Contagem de categorias:** Frequência de linguagens de programação primárias nos repositórios analisados.
- **Visualização de dados:** Geração de tabelas para facilitar a interpretação dos resultados.

As seguintes ferramentas e tecnologias foram utilizadas para a análise de dados:

- **Linguagem de programação:** Python.
- Bibliotecas: Pandas (para manipulação de dados), NumPy (para cálculos numéricos)

Para cada uma das questões de pesquisa (RQs), foram definidas métricas específicas:

1. **RQ 01 - Idade dos repositórios:** Calculada a partir da diferença entre a data atual e a data de criação do repositório.
2. **RQ 02 - Contribuição externa:** Quantificada pelo número total de *pull requests* aceitas.
3. **RQ 03 - Frequência de *releases*:** Calculada pelo número total de *releases* publicadas.
4. **RQ 04 - Atualização recente:** Determinada pelo número de dias desde a última atualização no repositório.
5. **RQ 05 - Linguagens de programação:** Contagem da linguagem primária de cada repositório.

6. **RQ 06 - Taxa de fechamento de *issues*:** Calculada pela razão entre o número de *issues* fechadas e o total de *issues*.

Os dados processados foram armazenados em um novo arquivo CSV (resultados_analisados.csv), contendo as medianas calculadas.

3 APRESENTAÇÃO E DISCUSSÃO DOS RESULTADOS

3.1 Apresentações e Hipótese dos resultados

RQ 01. Sistemas populares são maduros/antigos?

Métrica: Idade do repositório (calculado a partir da data de sua criação)

Hipótese: Espera-se que sistemas populares sejam maduros, ou seja, tenham uma idade considerável, pois sistemas mais antigos têm mais tempo para ganhar popularidade e contribuições.

- **Mediana da idade dos repositórios:** 8.69 anos

RQ 02. Sistemas populares recebem muita contribuição externa?

Métrica: Total de *pull requests* aceitas

Hipótese: Espera-se que sistemas populares recebam muitos contribuições externas, refletindo uma comunidade ativa e engajada.

Resultado:

- **Mediana de *pull requests* aceitas:** 415

RQ 03. Sistemas populares lançam releases com frequência?

Métrica: Total de *releases*

Hipótese: Espera-se que sistemas populares lancem releases com frequência, indicando uma manutenção ativa e evolução contínua do projeto.

Resultado:

- **Mediana de *releases*:** 42

RQ 04. Sistemas populares são atualizados com frequência?

Métrica: Tempo até a última atualização (calculado a partir da data de última atualização)

Hipótese: Espera-se que sistemas populares sejam atualizados com frequência, refletindo uma manutenção contínua e resposta rápida a problemas e novas funcionalidades.

Resultado:

- **Mediana de dias desde a última atualização:** 0 dias

RQ 05. Sistemas populares são escritos nas linguagens mais populares?

Métrica: Linguagem primária de cada um desses repositórios

Hipótese: Espera-se que sistemas populares sejam escritos em linguagens de programação populares, como *JavaScript*, *Python e Java.*

Resultado:

- **Contagem por linguagem primária:**
  - **JavaScript:** 78 repositórios
  - **Python:** 68 repositórios
  - **TypeScript:** 62 repositórios
  - **Go:** 32 repositórios
  - **Java:** 28 repositórios
  - **C++:** 20 repositórios
  - **Outras linguagens:** 52 repositórios

RQ 06. Sistemas populares possuem um alto percentual de *issues* fechadas?

Métrica: Razão entre número de *issues* fechadas pelo total de *issues*

Hipótese: Espera-se que sistemas populares tenham um alto percentual de *issues* fechadas, indicando uma boa gestão de problemas e uma comunidade ativa na resolução de *issues*.

Resultado:

- **Mediana da taxa de fechamento de issues:** 1.0 (100%)