Relatório Final

Luís Felipe Teixeira Dias Brescia
Victor Reis Carlota

Análise da Popularidade de Projetos de Código Aberto

(i) Introdução com Hipóteses Informais

Este documento tem como objetivo investigar os fatores que contribuem para a popularidade de projetos de software de código aberto. A popularidade de um projeto, frequentemente mensurada pelo número de estrelas em plataformas como o GitHub, pode ser influenciada por uma série de características intrínsecas ao projeto e ao seu desenvolvimento. Acreditamos que projetos de sucesso tendem a exibir certas tendências em sua trajetória.

Hipóteses Informais:

- H1 (Idade e Popularidade): Nossa primeira suposição é que sistemas mais populares são, em geral, mais maduros e antigos. A lógica por trás disso é que leva tempo para um projeto ganhar reconhecimento, atrair uma comunidade e acumular um número significativo de estrelas.
- H2 (Contribuição Externa e Popularidade): Em segundo lugar, hipotetizamos que sistemas populares recebem muitas contribuições externas. Um projeto de destaque naturalmente atrairia mais desenvolvedores interessados em colaborar, resultando em um alto volume de *pull requests* aceitas.
- H3 (Frequência de Releases e Popularidade): Por fim, esperamos que sistemas populares lancem *releases* com frequência. A atualização constante e o lançamento de novas versões indicam um projeto ativo e bem mantido, o que pode ser um atrativo para usuários e contribuidores, impulsionando sua popularidade.
- H4 (Atualização e Popularidade): É provável que projetos populares sejam atualizados frequentemente, garantindo que o projeto esteja sempre relevante e funcional.
- H5 (Linguagem e Popularidade): Hipotetizamos que projetos populares são desenvolvidos em linguagens de programação amplamente utilizadas e reconhecidas, devido à maior base de desenvolvedores e recursos disponíveis.
- H6 (Fechamento de Issues e Popularidade): Acreditamos que sistemas populares mantêm uma alta taxa de fechamento de *issues*, demonstrando responsividade e boa gestão do projeto.

(ii) Metodologia Utilizada

Para investigar as hipóteses acima, a análise baseou-se em um conjunto de dados de repositórios, onde a popularidade foi definida pelo número de estrelas de cada projeto. As questões de pesquisa (RQs) foram formuladas da seguinte forma:

- RQ 01: Sistemas populares são maduros/antigos?
    - Métrica: Idade do repositório (calculada a partir da data de sua criação).
- RQ 02: Sistemas populares recebem muita contribuição externa?
    - Métrica: Total de *pull requests* aceitas.
- RQ 03: Sistemas populares lançam *releases* com frequência?

    - Métrica: Total de *releases* lançadas.
- RQ 04: Sistemas populares são atualizados frequentemente?
    - Métrica: Dias desde a última atualização.
- RQ 05: Sistemas populares estão concentrados em linguagens de programação populares?
    - Métrica: Média de estrelas por linguagem e Total de estrelas por linguagem.
- RQ 06: Sistemas populares apresentam uma taxa elevada de fechamento de *issues*?
    - Métrica: Taxa de fechamento de *issues*.

A metodologia consistiu na construção de gráficos de dispersão para as RQs 01, 02, 03, 04 e 06, onde o eixo Y representava o número de estrelas (popularidade) e o eixo X representava a métrica correspondente a cada RQ. Para a RQ 05, foram utilizados gráficos de barras ou linhas para visualizar a relação entre linguagens e estrelas. Essa visualização permitiu observar a existência ou ausência de padrões e correlações entre as variáveis.

(iii) Resultados Obtidos para Cada Questão de Pesquisa

A análise visual dos gráficos revelou os seguintes resultados:

- RQ 01 (Idade do Repositório x Estrelas): O gráfico "Idade (dias) x Estrelas" indicou uma dispersão significativa dos pontos. Embora haja uma concentração de repositórios mais jovens, não foi possível identificar uma tendência clara de que repositórios mais antigos possuam um número consistentemente maior de estrelas. Muitos projetos, mesmo com poucos dias de existência, já exibem alta popularidade.
- RQ 02 (PRs Aceitos x Estrelas): No gráfico "PRs Aceitos x Estrelas", os pontos também se mostraram amplamente dispersos. Não se observou uma correlação direta onde um maior número de *pull requests* aceitas necessariamente corresponderia a um maior número de estrelas, ou vice-versa. Projetos com muitas contribuições podem ter poucas estrelas, e projetos com poucas contribuições podem ser altamente populares.
- RQ 03 (Releases x Estrelas): O gráfico "Releases x Estrelas" apresentou um padrão semelhante de dispersão. A frequência de lançamentos de novas versões (total de *releases*) não demonstrou ter uma relação linear ou evidente com a popularidade do repositório. Há casos de projetos com poucas *releases* e muitas estrelas, e o contrário também foi observado.
- RQ 04 (Dias desde a última atualização x Estrelas): O gráfico "Dias desde a última atualização x Estrelas" mostra que a maioria dos sistemas populares apresenta um número reduzido de dias desde a última atualização, concentrando-se próximos de zero. Isso indica que projetos populares, em sua maioria, são mantidos ativamente. Entretanto, observamos que alguns repositórios continuam muito populares mesmo sem atualizações recentes, evidenciando que a popularidade pode ser mantida também por fatores históricos (como pioneirismo ou utilidade consolidada), independentemente de atualizações frequentes.
- RQ 05 (Linguagem e Popularidade): A análise mostra que a maioria dos sistemas populares está concentrada em linguagens amplamente utilizadas e reconhecidas globalmente, como Python, JavaScript e TypeScript, que lideram tanto em número total quanto em média de estrelas por repositório. Esse resultado sugere que há uma correlação entre a popularidade das linguagens na comunidade de desenvolvedores e a popularidade dos projetos construídos com elas. No entanto, observam-se exceções como Markdown e Dockerfile, que, apesar de não figurarem entre as linguagens de programação mais comuns, aparecem com elevado número de estrelas devido à

existência de repositórios de grande impacto que utilizam essas linguagens como primárias.
- RQ 06 (Fechamento de Issues e Popularidade): O gráfico de dispersão "Fechamento de Issues x Estrelas" mostra que, em geral, sistemas populares apresentam uma taxa elevada de fechamento de *issues*. O gráfico de dispersão evidencia que os projetos com maior número de estrelas concentram-se entre 70% e 100% de fechamento, indicando uma comunidade ativa na resolução de problemas. No entanto, também existem casos de repositórios bastante populares com taxas de fechamento muito baixas, sugerindo que a popularidade não implica, necessariamente, em uma gestão eficiente de *issues*.

(iv) Discussão sobre Hipóteses e Resultados

Os resultados obtidos através da análise dos gráficos de dispersão apresentaram um contraste interessante em relação às nossas hipóteses iniciais.

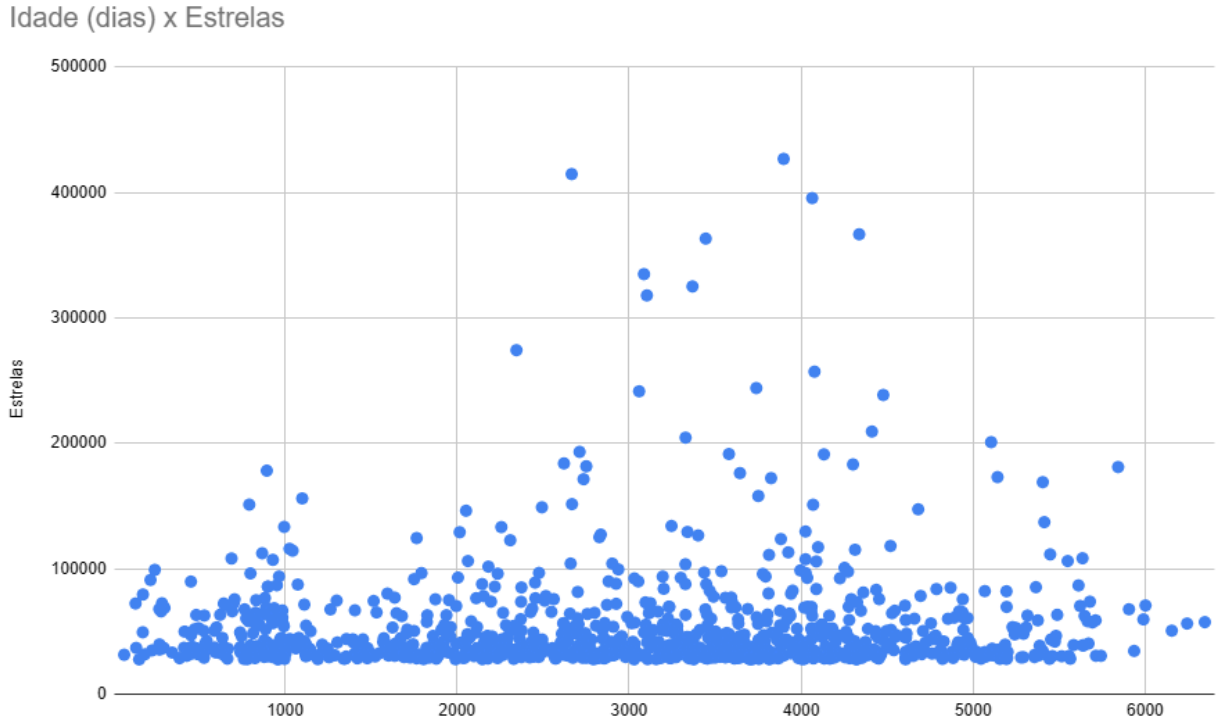
- H1 (Idade e Popularidade) vs. Resultados: Nossa hipótese informal sugeria que projetos mais populares seriam mais antigos. No entanto, os dados não apoiam essa ideia de forma conclusiva. A observação de projetos jovens com alto número de estrelas sugere que a popularidade pode ser alcançada rapidamente, desafiando a noção de que a maturidade é um pré-requisito estrito para o sucesso. A idade parece não ser um fator preditivo forte para a popularidade.
- H2 (Contribuição Externa e Popularidade) vs. Resultados: Esperávamos que projetos populares tivessem um alto volume de *pull requests* aceitas, indicando uma comunidade ativa e engajada. Contudo, a ausência de correlação direta nos gráficos sugere que a popularidade (estrelas) e o volume de contribuições externas não andam necessariamente de mãos dadas. É possível que alguns projetos populares sejam tão bem estabelecidos que exigem menos contribuições externas em termos de novas funcionalidades, ou que a natureza da popularidade venha de outros fatores, como utilidade ou marketing.
- H3 (Frequência de Releases e Popularidade) vs. Resultados: A expectativa era de que projetos populares mantivessem um ritmo constante de *releases*. Os resultados, no entanto, não indicaram uma ligação forte entre a frequência de *releases* e o número de estrelas. Isso pode significar que o valor de um projeto pode residir em sua estabilidade e funcionalidade já existente, e não necessariamente em um ciclo de lançamento rápido.
- H4 (Atualização e Popularidade) vs. Resultados: A hipótese de que projetos populares seriam atualizados frequentemente foi parcialmente confirmada. A maioria dos projetos populares é atualizada regularmente. Contudo, a existência de projetos populares que não são frequentemente atualizados sugere que a utilidade e o impacto histórico podem, em alguns casos, sustentar a popularidade mesmo na ausência de atualizações constantes.
- H5 (Linguagem e Popularidade) vs. Resultados: Esta hipótese foi fortemente corroborada. Há uma clara correlação entre a popularidade das linguagens (Python, JavaScript, TypeScript) e a popularidade dos projetos construídos com elas. As exceções, como Markdown e Dockerfile, que são linguagens de marcação e configuração, destacam que projetos fundamentais para o ecossistema de desenvolvimento podem alcançar alta popularidade, independentemente de serem linguagens de programação no sentido tradicional.
- H6 (Fechamento de Issues e Popularidade) vs. Resultados: A hipótese de que sistemas populares manteriam uma alta taxa de fechamento de *issues* foi confirmada em grande

parte. Projetos com muitas estrelas tendem a ter taxas elevadas de fechamento de *issues*, indicando uma boa gestão. No entanto, a presença de alguns projetos populares com baixas taxas de fechamento sugere que a popularidade pode ser mantida mesmo com uma gestão menos eficiente de *issues*, talvez devido à sua relevância intrínseca ou à capacidade da comunidade de encontrar soluções independentemente.

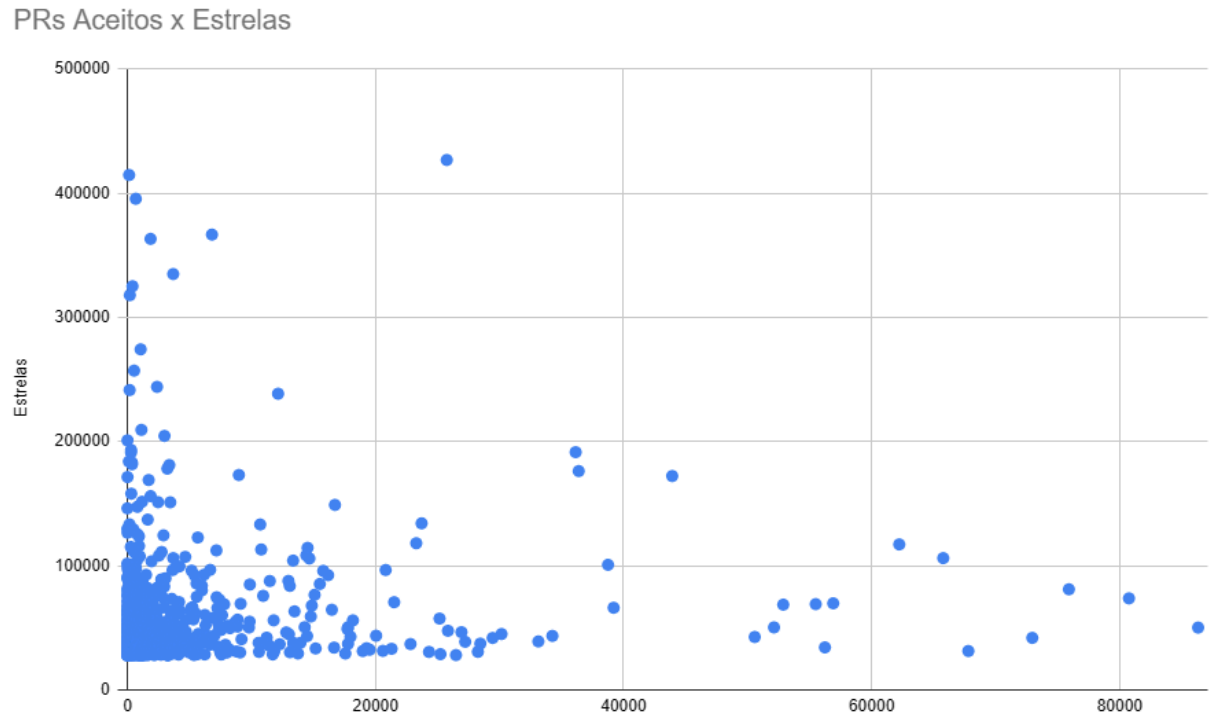
Conclusão da Discussão:

De modo geral, as hipóteses informais levantadas foram desafiadas ou parcialmente confirmadas pelos resultados. A popularidade de um projeto de código aberto, conforme medida pelo número de estrelas, parece ser um fenômeno mais complexo e multifacetado do que a mera correlação com a idade, o volume de contribuições externas ou a frequência de *releases*. Outros fatores, como a relevância do problema que o projeto resolve, a qualidade da documentação, a acessibilidade para novos usuários, ou até mesmo tendências de mercado e adoção por grandes empresas, podem desempenhar um papel mais significativo na ascensão de um projeto ao estrelato. A linguagem de programação utilizada e a taxa de fechamento de *issues* (com algumas ressalvas) mostram-se mais alinhadas com a popularidade, indicando a importância da base tecnológica e da gestão do projeto. Esta análise serve como um ponto de partida para investigações mais profundas sobre os drivers de popularidade no ecossistema de código aberto.

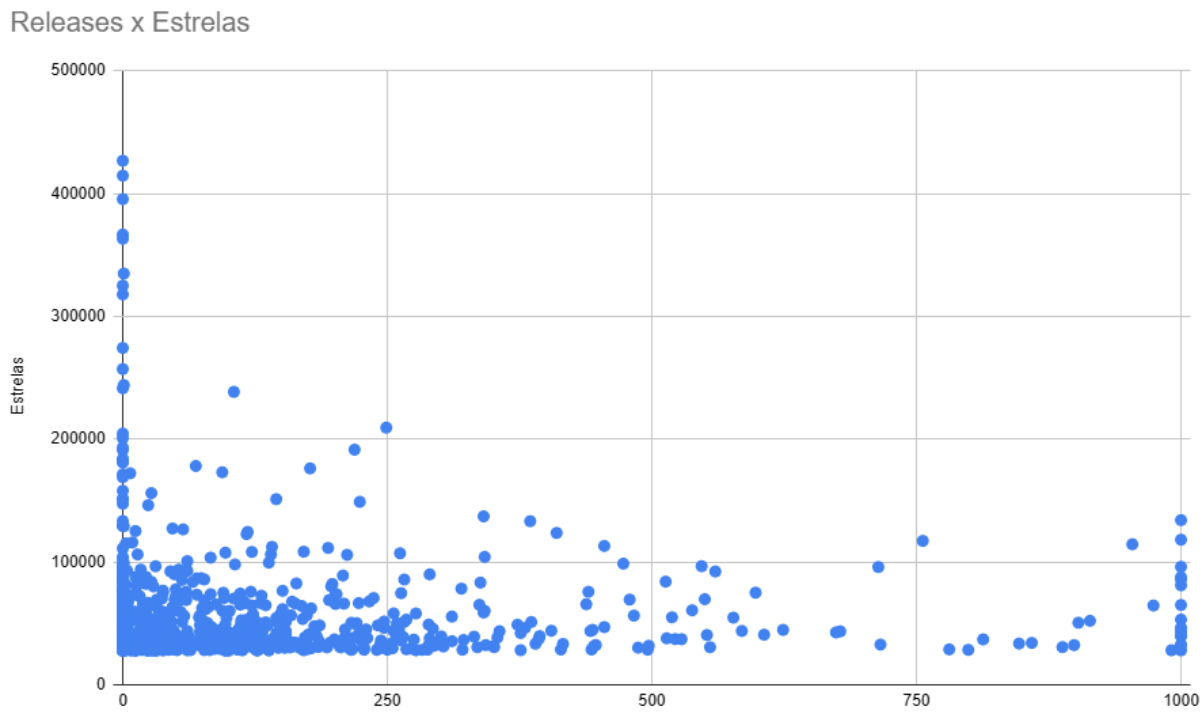
RQ 01.
O gráfico "Idade (dias) x Estrelas" não mostra uma correlação clara. A maioria dos repositórios, populares ou não, está concentrada nas idades mais baixas, o que sugere que muitos projetos podem alcançar um alto número de estrelas em pouco tempo. Não há uma tendência de que os projetos mais antigos sejam, necessariamente, os mais populares.



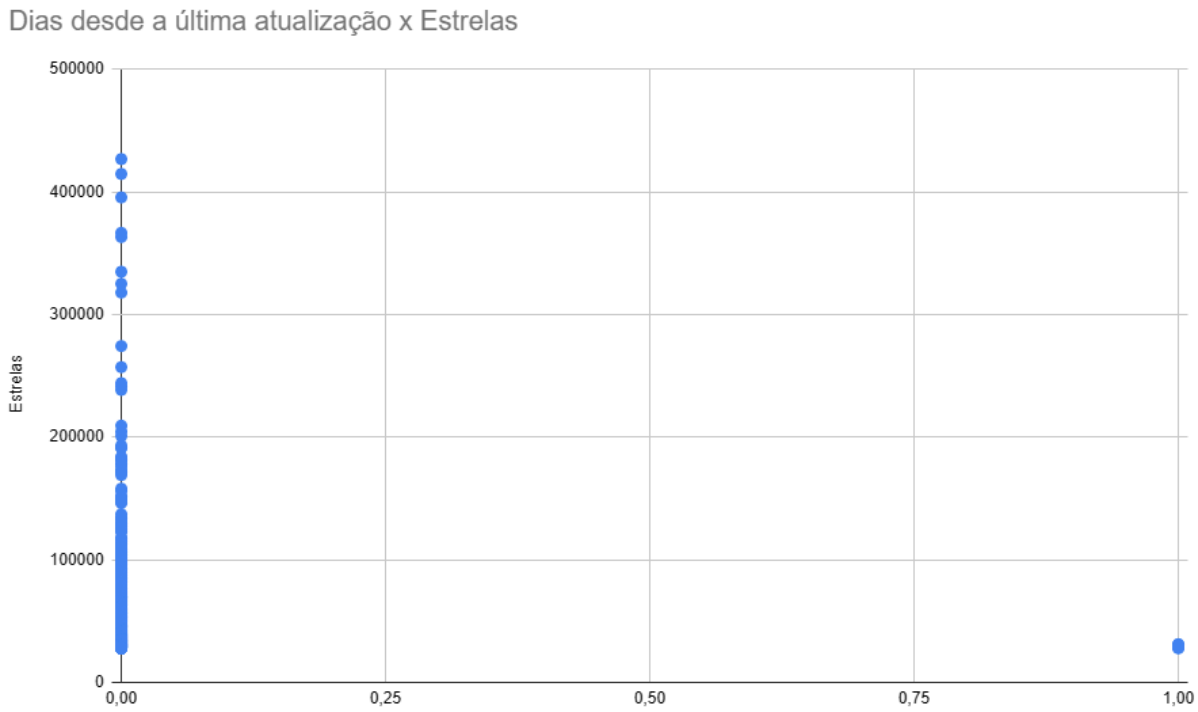
RQ 02.
O gráfico "PRs Aceitos x Estrelas" também não apresenta uma correlação evidente. Embora haja alguns repositórios com um número muito alto de pull requests aceitas, o número de estrelas é bastante disperso, sem uma relação direta com as contribuições externas. Isso indica que a popularidade de um projeto não está diretamente ligada à quantidade de contribuições aceitas.



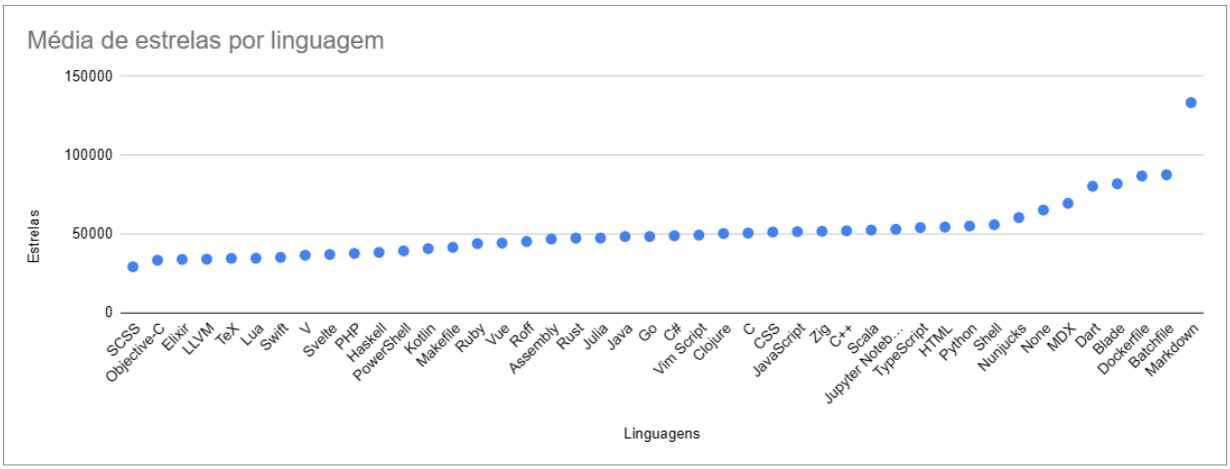
RQ 03.
O gráfico "Releases x Estrelas" mostra que, assim como nos outros casos, não há uma correlação forte. Existem projetos com muitas releases e poucas estrelas, e vice-versa. A dispersão dos dados indica que o número de releases não é um fator determinante para a popularidade de um repositório.

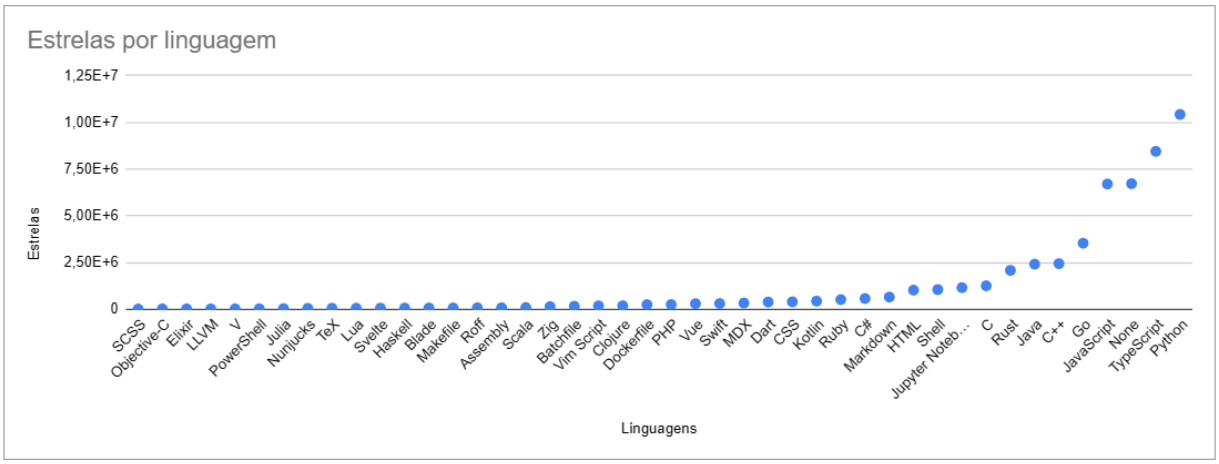


RQ 04.
A análise mostra que a maioria dos sistemas populares apresenta um número reduzido de dias desde a última atualização, concentrando-se próximos de zero. Isso indica que projetos populares, em sua maioria, são mantidos ativamente.
Entretanto, observa-se que alguns repositórios continuam muito populares mesmo sem atualizações recentes, evidenciando que popularidade pode ser mantida também por fatores históricos (como pioneirismo ou utilidade consolidada), independentemente de atualizações frequentes.



RQ 05.
A análise mostra que a maioria dos sistemas populares está concentrada em linguagens amplamente utilizadas e reconhecidas globalmente, como Python, JavaScript e TypeScript, que lideram tanto em número total quanto em média de estrelas por repositório.
Esse resultado sugere que há uma correlação entre a popularidade das linguagens na comunidade de desenvolvedores e a popularidade dos projetos construídos com elas.
No entanto, observam-se exceções como Markdown e Dockerfile, que, apesar de não figurarem entre as linguagens de programação mais comuns, aparecem com elevado número de estrelas devido à existência de repositórios de grande impacto que utilizam essas linguagens como primárias.





RQ 06.

A análise mostra que, em geral, sistemas populares apresentam uma taxa elevada de fechamento de issues. O gráfico de dispersão evidencia que os projetos com maior número de estrelas concentram-se entre 70% e 100% de fechamento, indicando uma comunidade ativa na resolução de problemas.

No entanto, também existem casos de repositórios bastante populares com taxas de fechamento muito baixas, sugerindo que a popularidade não implica, necessariamente, em uma gestão eficiente de issues.