The MR Scraper: Turning Data to Knowledge

A Technical Report submitted to the Department of Computer Science

Presented to the Faculty of the School of Engineering and Applied Science

University of Virginia • Charlottesville, Virginia

In Partial Fulfillment of the Requirements for the Degree

Bachelor of Science, School of Engineering

Kevin Cooper

Spring, 2023

On my honor as a University Student, I have neither given nor received unauthorized aid on this assignment as defined by the Honor Guidelines for Thesis-Related Assignments

ADVISOR

Briana Morrison, Department of Computer Science

ABSTRACT
The productivity tracker at General Atomics Commonwealth Computer Research, Inc., commonly known as the merge request scraper, or MR Scraper, had flaws and was incomplete. As an intern there, my solution utilized GitLab's Application Programming Interface (API) to gather the company's productivity information. That data was then parsed and transferred into a sensible format and saved into a PostgreSQL database. It was finally exported and displayed as graphs and charts on a Grafana dashboard. The company's progress data was shown during the next company-wide meeting. In the future, the scraper can be improved by fixing the other bugs I found while coding in my solutions.

1. INTRODUCTION

Lasting success is driven by productivity. For large website-focused companies, this takes driven workers who collaborate well. It is difficult to write with someone else using pen and paper. Programs like Google Docs and Microsoft Office Online allow multiple users to type on the same document simultaneously. This is helpful until someone writes directly on top of someone else's work or deletes everything off the page. Git and GitLab, one of its interfaces, are powerful tools to combat this collaboration issue.

Git is a local version control system (VCS) that allows for multiple users to work on the same files on their own and then merge them together, with a Merge Request (MR), when the authors are ready. General Atomics Commonwealth Computer Research, Inc. (GA-CCRi) uses GitLab because of its code version and integration control. GA-CCRi is an industry leader in geospatial storage, visualization, and analysis. They are a defense contractor that specializes in real-time situational awareness of aerial and nautical vehicles around the globe. They collect data from a variety of sources, including vehicular and weather sensors, with predictive analysis and data science techniques to achieve precise locations for all tracked vehicles with a low latency. Their clients are various government agencies around the world and commercial companies that have a use for global data or the company's global visualization program.

I spent my time at GA-CCRi as a back-end developer working on the company's MR Scraper. It is a program which gathers data from the company's GitLab API and displays it in a digestible manner. The scraper collects MR and project data like who collaborated on a merge request or project and how long it was worked on. That data is then displayed on a Grafana dashboard. This helps the company immensely because it allows for easy comprehension of the productivity at GA-CCRi.

2. RELATED WORKS

Git is a local VCS that stores information about a filesystem by collecting snapshots of the files in a repository. Git remembers the current image with a reference of the repository whenever the project is saved or committed (Git-scm, 2022). Different versions of the code, or branches, can be edited and have their own commits. Once a developer is content with their branch, they can merge it back to the main branch with a merge request (MR). An example workflow is shown below in Figure 1. An MR is open from when the original developer claims that they are done with the code and it is closed when their

branch is merged onto main after getting it reviewed and accepted by a colleague. If an employee merge's their own branch without review, then it is self-merged.

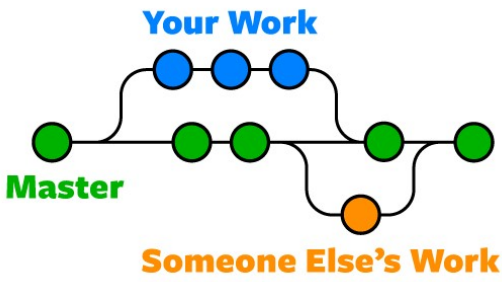

Figure 1. A sample Git repository with branches which are merged together.
© 2018 by F. Igor.

Merge requests are one of the last lines of defense for buggy code to be deployed to the complete project. Developers draft them in such a way that their colleagues can decipher what they changed on their branch and why it should be changed for everyone. This can be done in a few ways but comments are one of the most helpful because it does not clutter the code itself with comments, but can still have a plaintext way to explain someone's reasoning (Fayock, 2019).

GitLab is an online interface built on Git (Pipinellis**,** et al, 2022). It stores a version of the code online so that anyone who has access can create a clone of the repository on their own machine. Developers write code and can push what they wrote, or pull new information from a colleague from the combined repository. Another benefit is GitLab's built-in integration tool (Shipton, 2019). The integration tool allows for a company's website to be tested and built straight from GitLab.

The two end points of the MR Scraper are that it reaches an external website's API and then is eventually put in a Grafana dashboard. An API is a type of software that communicates with the specified website and gives document reports of the information on the page (IBM, 2022). Grafana is a web application that provides charts, graphs, and text for viewers to understand the data that has been collected (Grafana, 2022). A sample Grafana dashboard is shown below in Figure 2. The Scraper does all of the work in between and is the meat of the program.

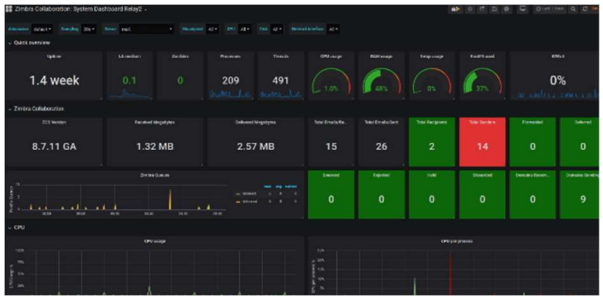

Figure 2. A sample Grafana dashboard.
© 2019 by Capterra.

3. PROCESS DESIGN

The six steps when data scraping are to identify a source, determine how that data is represented, write a program, harvest the data, traverse the webpages that contain data, run test cases, execute the scraping algorithm, and build the final dataset (Landers, 2017). My time at GA-CCRi presented each step because the MR Scraper uses this process to gather data from GitLab's API. The company uses GitLab for its project repositories so it makes sense to gather data from the GitLab API. It first asked for information from the API and processed the data. This data was then sent to a PostgreSQL database for mass storage. This database collected the new data while also holding all of the historical information. Whenever the database is updated, the Grafana table is updated, as well.

The company had an incomplete implementation of the scraper before I started on it because it contained bugs and had missing information. I worked to fix most of the known bugs in the code while also adding some missing features. I contributed by adding more information tables like a Git commit, a Git changes, and a project statistics table. The language Scala was used for the API scraping and for the functions that allow it to be run repeatedly. SQL is how the data was gathered and sent to the database (Koleoso, 2021) . Grafana was connected separately and I made new visuals respectively to the added information and added information to the current tables. Most notably, I added the graphs for the average time that an MR was open, and how many were self-merged.

Directly after an MR is initiated and after each commits to that branch thereafter, GitLab would run automated tests to see if the code would compile and build with the rest of the program. GitLab's API updates almost instantly so when the scraper is run, it contains all up-to-date information. The MR scraper is run automatically every hour during work days to ensure consistently accurate Grafana displays.

Like most coding projects, the process was not smooth. It consists of writing a feature or fixing a bug and then testing it to find a new error. This was frequent because I was new to Scala. One of the more difficult problems was understanding how the program converted the API's JavaScript data into something that Scala would understand and be able to send to SQL. I only worked on one problem at a time. I added a new function, tested it, and then only after my own MR was merged, I could move to the next task. It was entertaining to see the Grafana graphs update with the merge requests that I made for this project.

4. RESULTS

GA-CCRi previously rarely presented the gathered information from the scraper because there was not much information to display. Now they can use the scraper to get the necessary information to make informed decisions about how to be more productive as a company. I added plenty of important statistical tables that can now be viewed by those in managerial positions to share with their employees if they feel the need. The company can create more effective merge requests which leads to fewer bugs in the long run and therefore saves time and money.

The MR Scraper was integrated seamlessly into one of GA-CCRi's repositories in GitLab. It scrapes all MRs and project information for GA-CCRi's various repositories on GitLab including its own, all with the same accuracy standards. It automatically runs and updates the new information without needing to be changed for each use. At the full company meetings, my Grafana graphs will be shown to inform the company of their work and productivity levels. The MR and project data can be compared with the recommended productivity levels at GA-CCRi. Some statistics to be evaluated are how long an MR should be open, how many comments are on each, and if it is being merged by the person who requested the MR.

Presenting supervisors with the scraper's information can help them know what everyone is working on and show the supervisors how to best redistribute tasks if necessary to balance everyone's workload. Presenting the information to employees can highlight those that are doing well and incentivize others to work as hard and use their time wisely at work (Patnaik, 2022). Productivity trackers are beneficial for transparency on how the company works;

however there is a limit to how much companies should track their employees.

5. CONCLUSION

The MR Scraper is a program designed to collect data from General Atomics Commonwealth Computer Research, Inc.'s (GA-CCRi) GitLab API and display it on a Grafana dashboard. The Scraper successfully collects MR and project data, which allows for easy comprehension of GA-CCRi's productivity. The MR Scraper has demonstrated that it can be used to convert data into knowledge. The success of this project has demonstrated the value of using APIs and data visualization tools in managing a company's productivity.

Throughout my experiences at UVA and during my internship, I have honed a range of skills that will serve me well in my future career. These include coding proficiency, effective communication, and a deep understanding of the value of tracking and analyzing productivity. The MR Scraper project has reinforced the importance of reflection and continuous improvement, and I am confident that its results will benefit GA-CCRi's work culture moving forward. Overall, my internship has been an invaluable learning experience, and I am excited to apply my newfound skills and knowledge to future projects.

6. FUTURE WORK

I have accepted the work offer from GA-CCRi and will be working there next year full time. I am unsure if I will be continuing to work on the MR Scraper, but there will always be something to change or add to the project. Future improvements can be made to the program like better error handling, improved data visualization capabilities, or more efficient data storage and retrieval. There is always an option to fix any bugs that I found or others found after me and make it more efficient. As long as the scraper is regularly maintained it will remain as a valuable tool for the company.

REFERENCES

Fayock, C. (2019, May 2). Why you should write merge requests like you're posting to Instagram. Retrieved October 26, 2022, from FreeCodeCamp.org website: https://www.freecodecamp.org/news/why-you-should-write-merge-requests-like-youre-posting-to-instagram-765e32a3ec9c/

Git-scm. (2022). Git—What is Git? Retrieved October 26, 2022, from https://www.git-scm.com/book/en/v2/Getting-Started-What-is-Git%3F

Grafana. (2022). Grafana: The open observability platform. Retrieved October 26, 2022, from Grafana Labs website: https://grafana.com/

IBM, C. E. (2022, September 28). What is an Application Programming Interface (API)? Retrieved October 26, 2022, from https://www.ibm.com/cloud/learn/api

Koleoso, T. & O'Reilly Online Learning: Academic/Public Library Edition. (2021). *Beginning jOOQ: Learn to Write Efficient and Effective Java-Based SQL Database Operations*. S.l.: Apress. (Internet materials). Retrieved from http://RE5QY4SB7X.search.serialssolutions.com/?V=1.0&L=RE5QY4SB7X&S=JCs&C=TC_046418529&T=marc

Landers, R. N. (2017). Crash Course in I-O Technology: A Crash Course in Web Scraping and APIs. *TIP: The Industrial-Organizational Psychologist*, *55*(2), 5–11.

Patnaik, K. (2022, March 7). 05 Excellent Ways Productivity Tracker Can Boost Employees Productivity. Retrieved October 26, 2022, from https://empmonitor.com/blog/productivity-tracker-improving-work-performance/

Pipinellis, A., Read, E., Sedlak-Jakubowski, M., Qualls, A., & Selhorn, S. (2022). Git on the command line | GitLab. Retrieved October 26, 2022, from https://docs.gitlab.com/ee/gitlab-basics/start-using-git.html

Shipton, L. (2019, October 14). GitHub vs GitLab: Which Platform should I choose? Retrieved October 26, 2022, from Venture Lessons website: https://www.venturelessons.com/github-vs-gitlab/