An Overview of Project Manager

By: Dominic Bembry, Zachary Sublett, Sid Govindasamy Ramalingam

Abstract

The field of software development has become extremely fast-paced. Because of this good project management and organization has become a vital part of success. This paper proposes a novel solution to efficiently organize and manger projects. This solution incorporates a project manager into VS Code one of the most popular integrated developer environments (IDE). It includes several features such as: project switching, file information viewer, GitHub integration, time tracker, and a few other features that have not been fully fleshed out yet. With these features developers should have a fast and efficient way to view, access, organize, and keep track of all of their current project. Which in the long run will aid in speeding up the development cycle while also causing less stress for developers.

ACM Reference Format:
By: Dominic Bembry, Zachary Sublett, Sid Govindasamy Ramalingam. 2024. An Overview of Project Manager. In . ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 Introduction

Today the landscape of software development features many ever changing technological advancements and evolving methodologies. This has led to project management becoming a pertinent part of a projects success. But, as the complexity of projects grow, and the timeline afforded to developers shrinks, proper management of projects can become difficult to employ. This paper attempts to address those issues through the integration of a project manager directly into a developers integrated development environment (IDE). Specifically VS Code which is the IDE that will be utilized in this paper.

By consolidating tools that are essential for project management within VS Code this will help developers streamline their workflows, improve organization, and efficiently track a projects progress. This solution should not only enhance efficiency, but also reduce the stress that comes with working on multiple development projects at the same time. Ultimately, the goal of this project is to speed up the development process while still allowing developers to produce high-quality products.

Conference'17, July 2017, Washington, DC, USA

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM
https://doi.org/10.1145/nnnnnnn.nnnnnnn

2 Features

We were able to add four features to our VScode extension: File Information Viewer, GitHub Repositories Viewer, Project Switching, and TimeTracker. The file information viewer feature shows the user information about their files, the GitHub repositories viewer shows a list of all of the user's repositories, the project switcher allows someone to create and switch between projects seamlessly, and TimeTracker tracks the amount of time a user spends in an opened file. Each of the following subsections will go into more detail about each of the different features we implemented. It is important to note that we encountered some library issues and merge conflicts that rendered some of these features to not work, and we were unable to determine what caused the issue in time.

2.1 File Information Viewer

The file information viewer feature of our VScode extension is meant to show the tree structure of

your file system starting at the directory you opened to begin working in VScode. It is supposed to display the absolute path of each file in a sorted fashion. Clicking on each of these nodes would display all of the data that can be found in the file stat of a file. This is one of the features that unfortunately broke after a merge issue with the project switching feature, and we did not have time to resolve the issue.

2.1.1 Motivation. The motivation behind this feature is to provide the user with a convenient way to view file information without having to run the stat commands either on the command line or by making a program that displays the information for you. It is not meant to be the main focus of the project, but rather an additional bonus that the user can choose to use if they need or want to.

2.1.2 Issues Encountered. The file information system feature supposed to be a tree view representation of the entire subdirectory starting at the VScode workspace root, which is just whichever directory you initially open up to begin working in VScode. In order to get the data we needed to do this, we had to recursively read the file system from that workspace root directory. We ran into our first roadblock when attempting to read through the file system, which was asynchrony. Initially, we used the VScode API for handling file I/O, but we quickly ran into the issue of nested promises from these read functions were not behaving as we had expected them to. We would be able to catch one promise, but then when we tried to use that caught value for the next promise it could not find the value. We attempted different ways of waiting for promises and using its value, but ultimately we decided to move on from using the VScode API for this and tried the built-in synchronous file reading provided in the 'fs' library of JavaScript. After successfully reading from the files, we wished to store them into a local JSON file that way they could easily be read into the tree that stored and displayed the file information, but this is where came across the issue of the Enhanced Read-Only File System (EROFS) of Linux as well as the issue of cross-compatibility with other operating systems. The EROFS was a change to the Linux kernel in November of 2019 that was aimed at making the file system read-only in various cases that called for it [1]. We were never able to directly solve this issue, instead we had to use a workaround that ended up generating another problem. However, I believe that this system triggered due to a security oversight with our extension. Since VScode extensions are free to download for any user, people who download our extension would have to hope that it is a well-behaved program, but being able to write directly to a person's file system is dangerous if our extension was malicious. The workaround we developed for this issue was to write to the user's /tmp directory, which the file system allowed us to do. This solution works well for Linux and Mac users, but unfortunately Windows users do not have a /tmp directory. Instead, Windows users have a /temp directory so the extension would call an exception when it tries to write a JSON file in the missing directory. We unfortunately did not have time to find a solution to this issue, so the extension is currently not compatible with Windows users.

2.2 GitHub Repositories Viewer

The GitHub repositories viewer feature is meant to display all of the repositories that are associated with a given GitHub personal access token. In its current state, the repositories would be viewable from the primary sidebar in our extension, and would have their full name as well as the URL associated with that repository. This feature was made functional using the GitHub REST API and the Octokit library. This was the last feature made, and as such it did not receive all of the functionality that we wanted it to. This feature also broke prior to our demonstration due to an issue we will discuss in section 2.2.2.

2.2.1 Motivation. The motivation behind this feature was to provide users with a way to manage their GitHub repositories from their workspace, without needing to go to github.com for everything. Some of the functionality we had in mind was creating and deleting repositories, generating new tokens, creating and viewing issues in a repository, and any other small features that GitHub

offers. The idea was that it would be more convenient for the user to do as much work as they can from their workspace without needing to tab back and forth between their VScode workspace and their browser.

2.2.2 Issues. The first issue we had was storing and using the personal access tokens (PAT), from the user. Whenever we received the PAT from the user, it would be stored in plain text, which is a very serious security issue. We investigated various techniques for securing the token, like using environment variables as the token instead, but we did not have enough time to fully flesh out that security so it is currently still stored in plain text. The larger issue we had, however, was that near the date for our project demonstration two of our group members were experiencing issues with the Octokit library that we are using to fetch the GitHub repositories. One of the node modules in the library was no longer supported in the version of nodeJS we were working in, and uninstalling and reinstalling the library proved to be ineffective. This occurred whenever we encountered one of our merge conflicts, so we believed that perhaps that is what caused the issue, but even rolling back the troublesome commit did not fix the issue. We are still unsure if the issue stemmed from something that we did or something that Octokit changed, but in the end we were never able to resolve this issue.

2.3 Project Switching

The project switching feature enables the user to seamlessly switch between each of their projects in vs code. To use this feature the user must first add their desired project paths to a specific config.json file. This will then display the users projects in a tree view that is inside of vs code. The user can click on each one of the projects to seamlessly switch between them.

2.3.1 Motivation. The motivation of the project switching feature was to have one centralized way for a developers to efficiently view, and access each of their projects that they are currently working on.

2.3.2 Issues. The main issue with the project switching feature is that it relies on user to manually enter the path to their projects into a config.json file. This can be annoying and counterintuitive to the motivation behind this feature. To remedy this problem the feature was extended to include a command palette command that would streamline the configuration process by providing a file explorer pop-up that would let users select the path of their project much easier. However, this implementation was OS specific so, it has not been included in the version of the project that is discussed in this paper. This is a topic that will be revisited in the future works section.

2.4 Project Time Monitor

In the rapidly evolving field of software development, effective project management is crucial to the successful delivery of projects. Developers and project managers often grapple with the challenge of optimally allocating time and resources to various tasks and modules within a project. The Project Manager extension for Visual Studio Code is designed to enhance productivity and project oversight by providing robust tools tailored for managing software development projects directly within the IDE. One of the critical features of this extension is the ability to track and display the amount of time spent on each file or module within a project. This functionality is motivated by the need for greater transparency and accountability in software development processes. By understanding where time is being invested, both developers and managers can make more informed decisions regarding project priorities, resource allocation, and deadlines.

2.4.1 Motivation. The time tracking feature of the Project Manager extension automatically records the time a developer spends on each file or module when working within the Visual Studio Code environment . This data is then aggregated and can be displayed in various formats, such as charts or tables, providing a clear breakdown of time expenditure on different project components.

2.4.2 Technical Details. The feature integrates seamlessly with the VS Code workspace, using the

editor's events and APIs to monitor active file engagement. Time spent is only logged when the file is actively in focus, ensuring that the statistics represent genuine work periods. The data is stored locally or optionally synchronized with a central project management server to support remote teams and multi-developer projects. It consists of the following components:

Event Listener: This component hooks into the VS Code API to listen for editor events such as file open, file close, and focus change. This ensures that the time tracking starts and stops accurately as the developer switches contexts within the IDE.

Time Logging Engine: When an editor event indicates that a file has been opened or is in focus, this engine begins a timer. The timer pauses when the file loses focus or is closed. The timestamps are recorded with precision to ensure accurate time accounting.

Data Storage: Time data for each file is stored in a structured format, either locally in the user's workspace or on a remote server configured for team environments. The storage mechanism is designed to handle data efficiently, ensuring quick reads and writes without disrupting the user's workflow.

Synchronization Module: For teams, a synchronization feature ensures that time data from different team members can be aggregated. This module handles data conflicts and merges seamlessly, providing managers a unified view of the team's efforts.

2.4.3 Utility for Developers and Managers.

For developers, this feature in the Project Manager VS Code Extension serves as an invaluable tool for personal productivity and time management. By providing a detailed breakdown of time spent on each file or module, developers gain a clearer understanding of their work patterns and areas where they might be investing disproportionate amounts of time. This self-awareness allows

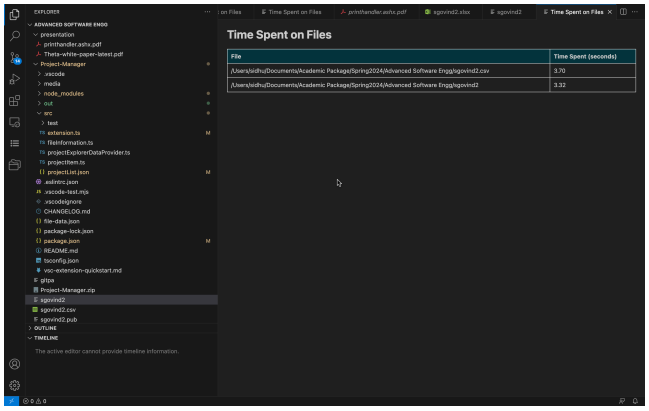

Figure 1. Project Time Monitor dashboard

developers to optimize their workflows, focus on prioritizing tasks that are crucial to the project's success, and reduce time spent on less impactful activities. Additionally, this feature can help in personal development by highlighting potential skills gaps or areas requiring further learning, as developers can see which tasks take them longer than expected. Over time, this not only enhances individual efficiency but also contributes to career growth by encouraging more balanced and strategic work habits.

For project managers, the utility of this feature is twofold: it enhances project oversight and improves resource allocation. By accessing aggregated data on the time spent by their team across different modules and files, managers can better understand workflow patterns, identify bottlenecks, and detect under or over-utilized resources. This information is crucial for making informed decisions regarding task assignments based on individual strengths and project demands, potentially leading to more balanced workloads and increased team productivity. Furthermore, the data-driven insights provided by the time tracking feature allow managers to refine project timelines and budget forecasts with greater accuracy. Managers can also use this data to justify resource requests and allocations in project reviews, ensuring that projects are not only managed more efficiently but are also aligned with broader organizational goals and strategies.

3 Conclusion

By embedding a project management tool directly inside of VS Code developers can streamline the development cycle and increase their productivity. As shown by this paper many of the features like, project switching, file information viewer, GitHub integration and time tracker are very intuitive but could still use some work to fully flesh them out.

4 Future Works

There are still several things that can be added or improved upon for this project. One of the major improvements that needs to be made is fixing the operating system compatibility issue with the project switching feature. This will allow users to easily add their projects to the manager regardless of their operating system. Another area of improvement would be to further extend the GitHub integration feature so it can create project repositories directly from the project manager extension within VS Code.

5 Contributions

5.1 Dominic Bembry

- Created File Information Viewer
- Created GitHub Repositories Viewer
- Helped trouble shoot GitHub Repository issues we had
- Made our GitHub Repository's README

5.2 Zachary Sublett

- Created the GitHub Repo
- Created project switching feature
- Helped with GitHub integration

5.3 Sid Govindasamy Ramalingam

- Developed a dashboard to monitor time spent on file
- Contributed to brainstorming the project manager features concept.
- Explored various methods for displaying data within the VS Code IDE.

6 References

[1] "EROFS - Enhanced Read-Only File System," The Linux Kernel documentation, https://docs.kernel.org/filesystems/erofs.html (accessed May 13, 2024).

[2] Microsoft, "Visual Studio Code Extensions API," Microsoft Docs, 2022. [Online]. Available: https://code.visualstudio.com/api. [Accessed: May 13, 2024].

[3] Gartner, "Magic Quadrant for Project and Portfolio Management," Gartner, Stamford, CT, 2020. [Online]. Available: https://www.gartner.com/en/documents/3 quadrant-for-project-and-portfolio-management. [Accessed: May 10, 2024].