Kanban Issue Tracking Application

Automated Issue Management

Domenic Martin
Computer Science
Virginia Tech
Blacksburg VA USA
dmartin80@vt.edu

Samantha Austin
Computer Science
Virginia Tech
Blacksburg VA USA
samanthaaustin@vt.edu

Ayda Haydarpour
Computer Science
Virginia Tech
Blacksburg VA USA
ayda@vt.edu

Peyton Ludwig
Computer Science
Virginia Tech
Blacksburg VA USA
peymae6@vt.edu

ABSTRACT

In recent years the adoption of collaborative tools has increased from 55% in 2019 to 75% in 2021 according to Tajammul Pangarkar in his article Collaboration Software Statistics 2024 By Workplace, Workers, Technology [1]. The foundation of any collaboration tool is to increase productivity in hopes of becoming more efficient. As software engineering becomes more embedded into society the need to continue to grow becomes more relevant. Due to the growth of the software engineering field, the lack of efficient collaborative tools has a noticeable negative effect on communication and general productivity; where money, company time, and general product functionality experience adverse effects.

Our Kanban Issue Tracking application is a bot created with the intent of addressing all losses of productivity software engineers and companies experience. The purpose of the application is to create a bridge between teams that considers each developer's workload and equally distributes work based on effort value, required skills, and team hierarchy. By automating and streamlining the Kanban workflow to increase developer productivity and tackling the problem of decreasing productivity within the software engineering field, the Kanban Issue Bot serves to push software engineers to their full potential.

CCS CONCEPTS

• Software development process management • Software development methods • Software configuration management and version control systems

KEYWORDS

Issue Assigning Automation, Software Tools, Software Development Management, Kanban, Issue Tracking, Project Management, Software Development Processes, Agile Methodology

1 Introduction

In modern software development, issue tracking plays a crucial role in managing the progress and efficiency of teams working on complex projects. Especially as companies scale, the challenges of effectively assigning tasks and balancing workloads among developers become more prevalent. Existing issue-tracking systems help with these problems, but they often lack features that automate key aspects of task assignment and workload distribution. Efficiently managing a software development team involves not only tracking progress, but also ensuring tasks are appropriately distributed based on the skills, experience, and availability of team members. This is especially important in Agile environments, where flexibility and adaptability are very important to the project's success. Automating these task assignments would save time, reduce bottlenecks, and improve overall productivity. Our solution is the Kanban Issue Bot, which automates the assignment of issues in the development workflow. It would consider factors such as team hierarchy, required skill sets, and workload distribution, and by doing this would enhance the productivity of software developers, streamline project

management, and reduce the administrative burdens of team leads.

2 Motivation Example

Imagine a software engineering team working on a large-scale application with a tight deadline. The team consists of developers with varying levels of experience, specialized skills, and different workloads. The project manager manually assigns tasks based on their understanding of the team's capacity, but this often results in inefficiencies. For instance, a senior developer might be overwhelmed with numerous high-priority tasks, while a junior developer has little to do, or a team member might get assigned a task requiring expertise outside their skill set.

These misalignments lead to delays, frustration, and lower overall performance. The lack of an efficient task assignment system not only hampers the progress of the project but also increases the administrative burden on team leads, leaving less time for strategic decision-making.

The KabBan issue Bot addresses this problem by automating task assignments using a combination of team hierarchy, skill sets, and workload distribution. For example, if a new feature request is added to the backlog, the bot evaluates the complexity of the task, matches it with developers who have the appropriate skills, and considers their current workloads. It then assigns the tasks to the most suitable team member, ensuring an even distribution of effort and preventing bottlenecks. This automation enhances team productivity, reduces delays, and fosters a more collaborative environment. By eliminating the manual effort in task allocation, the bot allows project managers to focus on higher-level planning, making it a valuable tool for software engineering teams navigating the challenges of modern development workflows.

3 Background

The KanBan System, originating from Toyota's manufacturing processes in the 1940s, is a workflow management methodology designed to visualize tasks and improve efficiency [2]. Taking a visual approach, it represents tasks as cards and their progress in labeled columns, providing a clear overview of work status.

Issue tracking refers to the process of identifying, recording, and managing tasks, bugs, or feature requests within a project.

Agile methodology is a popular approach to software development that places emphasis on flexibility, collaboration, and iterative progress.

4 Related Work

Task assignment software has been a complex issue to tackle for many years now. There have been several implementations and methods developed to address the challenge.

In recent years Kanban tools, such as Trello and Jira, have been used to create a visual task management software that allows teams to assign tacks and track progress of team projects. Both these software tools require a large amount of manual effort for tasks to be assigned in the appropriate fashion. In contrast, The KanBan Issue Bot introduces the idea of automating the KanBan system to optimize workload distribution.

The task assignment software, Jira, was created in 2002 with the original purpose of being a issue tracking software for software engineers [3]. Jira expanded over the years into a task management system. Jira has programmable automation features but it lacks advanced AI automation. Our KanBan issue bot takes the approach of AI automation to the extreme by completely automating the task system and evaluates in real time ensuring the team remains agile and productive.

Research done in the AI field has shown a 40% increase in skilled workers' performance with the

integration of AI in a study done by Harvard Business[4]. Existing works combined with our implementation of AI boosts productivity in the work environment and pushes past the issue of task management software systems.

5 Implementation Design and Progresses

5.1 High Level design Decisions

Our product will be designed using the client-server architecture. This will be used as the backbone of our system design because it easily allows for the system to synchronize user boards across different machines which will allow teams to collaborate on the same or linked boards.

We also used wireframing to design and finetune our UI. Wireframing helped us to ensure that the layout and functionality of the design makes sense to users and is simple to use while still providing advanced and effective features.

5.2 Implementation Processes

We used the KanBan software engineering process to manage the development of our product. It allowed us to easily track and visualize our team's tasks while limiting the amount of work that had to go into coordinating and assigning tasks.

The implementation of our product will follow the test driven development(TDD) approach. TDD will improve code quality and lead to better code design decisions while also simplifying the debugging process. Code quality will be improved under this approach as writing tests first allow developers to ensure that the code they have written meets the requirements of the new feature. TDD will also lead to better code design as writing tests first will force developers to better understand the requirements of new features before they begin coding.

5.3 Testing approaches

Our testing policy will focus first on TDD where developers write a failing test for a new feature, and then write the code to pass that test. This process is repeated until that feature is complete.. After a feature is fully implemented integration tests will be written to check that the behavior of the new feature works as intended. The developer will then modify the code until these tests pass. At this point our CI process described below will start as the new feature is pushed to the GitHub repository and tested against the previous integration tests. At the same time the developer will select the most important tests from this feature to add to the automated testing suite. This process will repeat for each new feature added to the system.

6 Deployment and Maintenance Plan

6.1 Deployment

Our deployment will be automated with a GitHub actions CI/CD pipeline. We are using GitHub actions because of its integration with our repository manager GitHub and its ability to quickly set up small scale pipelines and later build them out further. Since GitHub actions is fully integrated with GitHub it is simple to set up pipelines that are triggered by any interaction with the GitHub repository. In our case this would most commonly be pull requests. GitHub actions also allows developers to quickly set up a simple small scale pipeline which can later be scaled up. This would allow us to quickly get a pipeline up and running and expand it as needed to fit the growing needs of our project [5]. We are using CI/CD pipelines because they allow developers to ensure that new code doesn't break prior functionality by automating testing using prior integration tests. CI/CD also results in significantly faster deployments when compared to the prior model of long integration and deployment periods [6]. One common issue with CI/CD is test case bloat which can slow down integration and development. We will combat this by using test case prioritization where only the most important test cases are added to the automated testing suite.

We will also combine our CI/CD pipeline with canary deployments. Since our product will likely be used for large software development projects any downtime or errors will result in large productivity losses for our customers. To mitigate the risk to customers we will

use canary deployments to slowly release new builds to small groups of users in small phases. This will allow us to catch any missed errors or bugs in the program while only affecting a small portion of our user base.

6.2 Maintenance

Refactoring is an essential component of maintaining code. We will focus primarily on floss refactoring and particularly the red-green method. Red-green refactoring follows the test driven deployment cycle and consists of 3 steps. The first stage is "red" where you write a failing test case for the next piece of functionality. After that is the "green" stage where you write just enough code to pass the red stage test. The final stage is the refactoring stage where you optimize and clean up the code you just wrote without adding any new functionality. This cycle repeats until all of the functionality is added for the next deployment [7]. By using this refactoring method code is made better and simpler by the developer who is most familiar with how it works and while it is fresh in their memory. This speeds up the refactoring process as developers do not need to first work to understand the code. Developers will also be encouraged to refactor the code whenever they see a useful change that could be made. This will help our code adapt to future requirements and will simplify the code so that future developers can more easily understand and change it.

7 Discussion

The KanBan issue bot serves the purpose of addressing workload management issues, team hierarchy, and automates task management using AI to ensure a modern solution in software engineering task management applications. Yet, There are still several avenues the future of our application can go.

The future work for the KanBan issue bot is applying the design into code and creating a functional prototype to be tested. A working prototype would require a global and local kanban board for users, working methods that ensure the AI properly allocates tasks, and additionally having quality assurance testing where real users can interact with the prototype to guarantee the needs of the user are being met.

Despite the advantages the Issue bot addresses there have been limitations. The biggest being time. Due to the short nature of the semester there was not enough time to properly develop a prototype of the bot. Moreover, since time was limited edge cases were not properly diagnosed. Another challenge was the universal issue KanBan boards experience; an overwhelming amount of tasks in the "in-progress" section. This is often a case where KanBan boards will experience bottlenecking in the project. The problem can also lead to extreme stress in software engineers when there is an overwhelming amount to do on their GUI interface. In an attempt to address the issue we designed a notification system about upcoming due dates to help mitigate the opportunity for an overwhelming board.

Nonetheless the future for the KanBan Issue Bot is bright as with more time the limitations and issues can be addressed.

8 Conclusion

The KanBan Issue Bot would provide a significant advancement in task management for software engineering teams. By automating task assignment and leveraging AI to consider workload, skill sets, and experience, it addresses the gaps in efficiency in commonly used manual task assignment systems. This design would effectively streamline project workflows, reduce bottlenecks, and allow teams to work to their best potential. Although time constraints limited the development of a working prototype for this project, the foundation laid provides a view of the potential of AI-driven automation in software development processes. With implementation, testing, and refinement, the KanBan Issue Bot could become a major tool for Agile teams to ensure balanced workloads, better collaboration, and improved productivity.

REFERENCES

[1]Tajammul Pangarkar. 2024. Collaboration Software Statistics 2024 By Workplace, Workers, Technology. Market.us Scoop. https://scoop.market.us/collaboration-software-statistics/

[2]Kanban tool. Kanban Tool. (2024, August 6). https://kanbantool.com/kanban-guide/kanban-history

[3]Atlassian. (n.d.). Jira for teams. Jira. https://www.atlassian.com/software/jira/guides/getting-started/who-uses-jira

[4]Somers, M. (2023, October 19). How generative AI can boost highly skilled workers' productivity. MIT Sloan. https://mitsloan.mit.edu/ideas-made-to-matter/how-generative-ai-can-boost-highly-skilled-workers-productivity

[5]Douglas, B. (2024, July 23). How to build a CI/CD pipeline with github actions in four simple steps. The GitHub Blog. https://github.blog/enterprise-software/ci-cd/build-ci-cd-pipeline-github-actions-four-steps/

[6]Folwer, M. (2024, January 18). Continuous integration. martinfowler.com. https://martinfowler.com/articles/continuousIntegration.html

[7]GeeksforGeeks. (2022, January 21). 7 code refactoring techniques in software engineering. GeeksforGeeks. https://www.geeksforgeeks.org/7-code-refactoring-techniques-in-software-engineering/