Improving The Basic Kanban Tool

CS 3704, Virginia Tech, “Kanban Pioneers”

Jiale Song Ludwig Olaru Maddie Gonzalez

Shivani Vaddepalli Xinchen Liao Ziad Elgataa

ABSTRACT

Within a rapidly evolving environment of project management tools, kanban boards have emerged as a key tool for increasing organizational efficiency and task transparency. However, we believe traditional kanban board designs frequently fall short of meeting the flexibility and customization needs of varied teams and projects. In this term project for CS 3704 at Virginia Tech, our team, Kanban Pioneers, seek to create an improved tool to allow software engineering teams who use kanban boards to organize their work more productively as well as allow for increased customizability. The problem we wanted to address is the lack of useful productivity tools currently offered by companies who design kanban boards for consumers. Most kanban tools offer considerably basic functionality for their boards and lack customizability. We identified the fact that projects vary largely in scope, size, and complexity, necessitating the need for a tool that is both highly customizable and scalable to accommodate the unique needs of each project. Our product will integrate adjustable columns, a shared to-do list feature, dynamic task highlighting, and an effortless way to organize tasks by priority. Through a user-centered design, this project is constructed to accommodate swift changes, assure high performance, and preserve reliability. This results in a tailored project management solution that meets the various needs of today's teams. Moreover, our product is designed to integrate with agile methodologies that currently dominate the software engineering processes by allowing tasks to be organized into sprints or specific timeframes.

INTRODUCTION

After assessing current kanban tools such as Jira and Microsoft Azure, our team has identified several problems and areas for improvement. One problem in current kanban boards is that there is no function for every teammate to view their own tasks clearly. They can see all the tasks assigned to the team but are unable to see which tasks are assigned to themselves. We plan to design a new function called “My Tasks” which will allow users to view their assigned tasks directly from a side panel of the web page. A second problem is that current tools do not allow users to set a deadline or timeline for tasks. Having an option to set dates on tasks allows for several more features to be added to improve software development team organization. Such features that we wish to incorporate include time boxes for every task, an interactive calendar, and timed-task notifications. Allowing the option for time boxes will allow the manager/project lead to set the expected time duration for the whole project and each task, and teammates can organize their time better. For more usability, we will add a calendar function. There are two main improvements this will allow: a calendar view of a “to-do” list where all deadlines for tasks will be listed on the calendar, and a notification feature that will notify the user of start time and deadlines of tasks. Users can choose to use the original view or the calendar view as their preference and they will get a notification for starting day of project, meetings, and tasks. Users will also get notifications about the deadline as many times as they want. They can select how many days before deadlines they wish to receive a reminder, and how many reminders per day, week, or month. At the core of our project, we plan to have a fully functional kanban board that encapsulates all the good features of existing tools. The simplicity of the kanban tool is why so many software development teams choose to use it, and it is our goal to improve the tool rather than over-complicating it. We want to emphasize this by having the kanban board at the center of our web page, with all additional features within side panels to stress the fact that they are optional productivity tools. Our kanban board will also feature customizable columns to give teams the ability to modify titles to best accompany their work, as well as a feature to change the color of tasks that are blocked to draw attention to items that need to be resolved quickly.

RELATED WORKS

Jira issue tracker: Jira is an issue tracking website by Atlassian that offers many features, one of which is a Kanban board. The main draw of Jira is easy integration with code repository sites such as Bitbucket, GitHub, and GitLab, as well as customizable automated actions linked events such as creating an issue.

Microsoft Azure: Microsoft offers a version of a kanban board in their Azure tool. They have many features other than kanban, including a product backlog, scrum sprints, retrospectives, dashboards with customizable widgets, and even code repositories. Azure offers a great variety of tools for managing software projects, but they can be difficult to navigate and the kanban board specifically is not very user-friendly.

MONDAY work management: A software to manage the work. The visual design is a prominent feature of http://monday.com. The designers of this tool use color to make it easier for users to use it. As a user, you can easily judge the status of tasks and projects through color and status. Also, users can easily add new files, labels, links, etc. By clicking on 'Add more columns', and those will be displayed in different colors as well.

SOFTWARE ENGINEERING PROCESS

For this project, our team used an Agile framework with multiple sprints to complete work and keep track of deliverables that need to be turned in. This seemed like a logical choice because there is a clear structure to the schedule/deadlines of the project assignments, it allowed us to keep track of everyone's progress, and we could easily identify any blockages that occur during our project. We held stand-up meetings at the beginning of each group session and tracked our work using a kanban board in GitHub.

HIGH-LEVEL DESIGN DECISIONS

For the software architecture of our project, we decided to use Model-View-Controller architecture. There are some points that we need to consider. First, our improved Kanban board will have a web application and desktop application. We need a server end software to handle client request and changes. And the data of application will be hosted on our server, which allow user to access in any different devices that connect to internet.

The Model-View-Controller architecture perfectly fits our requirements. It has client tier, application tier, and data tier. Our server, which is in application tier, will handle all user's access and change request. The database, which is in data tier, will only connect to server software. This will prevent errors caused by multiple users accessing and modifying the same data at the same time. The web and desktop application, which is in client tier, can handle our UI design and create request and change to server. What's more, Model-View-Controller architecture is widely used in modern web applications, which proves that it is an excellent software architecture.

IMPLEMENTATION PROCESSES

While our project unfortunately did not reach the development stage, we still utilized many software engineering ideas throughout the design process. First, we used requirements elicitation to dictate the need for our project and how certain features may target different users of our system. Next, we used requirements analysis and user stories to streamline our ideas and narrow them down to dedicated requirements within our proposed solution. Our user stories strongly influenced the design of our final wireframe and prototype by capturing the importance of usability throughout the system. As we reached the end of the time constraints for developing our system, we reflected on how we would deploy our app if we had more time, as well as developing a black box test plan to test for bugs and ensure the system is well-maintained.

TESTING APPROACH

To test our system, we plan to deploy a black box test plan (See Black Box Test Plan in GitHub). This test plan will set expected results for different features of our application and evaluate the accuracy of the interface by observing the actual results. The features our test plan will analyze are creating a new task, dragging/dropping tasks between columns, editing tasks, navigating between tools, column functionality, task details, search functionality, multiple user support, response time, and authentication.

DEPLOYMENT & MAINTENANCE

If we were to continue with developing and then deploying and maintaining this project, we would choose to use Jenkins for continuous integration/deployment. We would have biweekly deployment meetings where we talk over the status of all the changes we originally planned to include in that deployment and create a new release branch in our code repository. On Jenkins, we would set up an automatic pipeline to trigger a build/production deploy on any new release branch.

We would also have a main development branch with similar automatic Jenkins build/deployments, except that for the dev branch, any commit will trigger a Jenkins pipeline, deploying to a private development version of our application that we can use for end-to-end testing.

For maintenance, we would also have a public changelog/wiki with information on the changes in each release. Somewhere on that page would be the option for users to report bugs, which we would add to a bug tracker and eventually convert to tasks. These tasks, in addition to tasks of our own creation for planned features, would eventually go into one of our future sprints for completion, and this whole cycle would repeat for the duration of the project's lifetime.

CONCLUSION

In conclusion, our improved kanban board offers a fresh rework of the traditional kanban tool that our team hopes will drive software teams to significantly improve their productivity. While we believe our solution is sound, there are some limitations and future features we would have liked to include.

Limitations

Due to the time constraints of our project, there were a few features that were not included in our final design. We chose to prioritize the features that are heavily tied to the main board functionality, so our solution does not include the task notification feature, the calendar view of task due dates, or the ability to highlight blocked tasks in a different color. Furthermore, we chose to keep our user interface very simple, so the lack of visual appeal is another limitation of our system.

Future Work

In terms of future improvements, we would first turn our design into a full-stack web application. Once development and testing are complete, we would consider adding the following additional features: More agile support (daily scrum standups, sprint retrospective tool, etc.), additional productivity tools (pomodoro timer, customized productivity analytics), and support for multiple user types with differing permissions (scrum master, development team, admin, etc.).

REFERENCES

[1] Atlassian. "Welcome to Jira Software." Atlassian, www.atlassian.com/software/jira/guides/gettingstarted/introduction. Accessed 18 Feb. 2024.

[2] Chcomley. "Use Your Kanban Board - Azure Boards." Azure Boards Microsoft Learn, learn.microsoft.com/enus/azure/devops/boards/boards/kanbanquickstart?view=azure-devops. Accessed 18 Feb. 2024.

[3] "Project Management." UTMDigital, digital.utm.my/project-management/. Accessed 18 Feb. 2024.