Taskboard Manager

Authors: Nishesh Saikrishna (nisheshsai9@vt.edu), Peter Murphy (pmurphy26@vt.edu), Benjamin Hurt (benjaminhurt@vt.edu), Joel Buba (joelb1@vt.edu)

ABSTRACT

Our problem is that there are many software engineers that are working in teams and do not have a proper way to organize their tasks and keep track of their deadlines. This can apply to either software engineers working in the workplace or software engineering students working together on software engineering projects. There are many instances where software engineers are working on multiple projects at the same time. In these cases, it can become very difficult to remember the details of each individual project they are working on. It also can be difficult to keep track of all of the deadlines for all of the different products.

Our proposed solution is to create a Taskboard Manager that will help software engineers working in teams to better manage their team's goals, deadlines, and division of work. It will help them prioritize tasks, set deadlines for tasks, and assign certain tasks to certain team members. We did not receive any feedback from the abstract section from our proposal.

INTRODUCTION

Almost every software company has their employees working in teams. Similarly, in many computer science projects, students often do projects and assignments in teams. Sometimes, these teams may have difficulty in terms of managing their workload. This can be problems with time management, division of roles, or remembering the details of their projects.

Our solution, which is the Taskboard manager will help software engineers keep track of all of the work they need to do on an individual level, but also help them remember what the rest of their team needs to do and the deadlines for all of their work. The Taskboard manager will allow team members to divide up their tasks, set deadlines for the tasks for all group members, and add notes to tasks to allow them to remember the key details for each task.

MOTIVATING EXAMPLE

One motivating example for our product is a software engineering firm that has two projects, Project A and Project B due in a few weeks. Project A is due in one week and Project B is due in two weeks. Person A and Person B are both working on Project A and Project B, but they both have different roles in the project. Our Taskboard app will let both users know which tasks they have in each project and when the project deadline is. It will also let them view certain details about each project, so they know what each project is about.

BACKGROUND

In the context of our application, a task is anything that needs to be completed by a software engineer. In most workplaces, this refers to a project that they are working on. The deadline refers to when the task needs to be completed by. We refer to task groups as the group of members assigned to collectively work on a group of tasks.

RELATED WORK

One example of related work to our project is Google TasksBoard [1]. TasksBoard is an application that allows users to add assignments or notes from any window and any device. The application allows for the ability to add descriptions and create subtasks with Tasks, which is a feature that we also wanted to implement in our own Taskboard. Another feature that the Google TasksBoard has similar to our product is the ability to set due dates to tasks. However, there were features such as assigning people to tasks that Google's TasksBoard does not have that we wanted to have in our taskboard.

DESCRIPTION OF IMPLEMENTATION

We used swift UI primarily for the GUI of our taskboard. We used it to create a super basic login/home screen and then our actual taskboard as well, which is made up of a panel of buttons to add users, tasks, and task groups, along with a button to remove task groups. The middle of the board is where all the task groups and tasks are, and the right side contains the currently selected task. We chose this implementation because it visually breaks up the board into different sections dedicated to different actions. The left side deals primarily with the creation of new objects for the taskboard. The right side of the board is where the finer details such as the due date and other notes about the task are displayed and the features of the task can be edited. The center is the main point of our taskboard and contains all of

the tasks that have been added to the board. Because we couldn't find other taskboards that allowed the user to sort the tasks by the group assigned to them we chose to sort the tasks this way.

The high level design pattern that we used was an event based pattern because this fit the use of our GUI. Users can interact with the taskboard by clicking on the various buttons found across the GUI, which in turn triggers an event on the backend such as adding and removing users, adding different groups working on different tasks together to the center panel, creating a task, and more. Even the act of creating a task can be broken down into different events, such as assigning the task name, assigning different users to the task, and adding any notes to the task. Since the act of creating, selecting, or editing a task would be treated as an event we would be able to easily produce new events, which is the main purpose of the taskboard. With this design pattern adding new features in the future as we look to maintain our application would be easy because the new feature would simply be a new event that is triggered by some button being presson on the screen. In this program, the GUI acts as the dispatcher which receives the user actions and other events, and then calls the appropriate functions to accordingly update itself and the taskboard object. In addition to our implementation design easily following the event based architecture design, there is no need for any of our components to have control over the order of execution so we do not have any drawbacks to consider when following the event based architecture pattern.

In terms of testing we used both white box and black box while developing our application. While developing new features, we often relied on white box testing to test the functionality of these new features and spot any bugs, especially in edge cases. We tested things such as system requirements and use cases during black box testing and made a lot of changes to our taskboard as we interacted with it. Some of these changes included adding the all tasks panel automatically to the taskboard, moving completed tasks to the bottom of the group, and selecting group members and setting the task status for a task from a list instead of having to type it in manually.

DEPLOYMENT PLAN

We would use a canary deployment strategy for our project. We believe that this would be the most efficient way to deploy the product because we would get good feedback from the users of the product. By starting off by just deploying the product to a few users, we think that we can get valuable feedback which will help us realize the strengths and weaknesses of the product. We would maintain the product after deployment by adding fixes to any issues that users encounter.

DISCUSSION

There are certainly more features and changes that we can add to our project to improve it. One such example of a feature is allowing different users to communicate with each other directly on the taskboard by leaving messages for their team members. This can help make communication more efficient, as they can see all of their tasks and the communication about each task all in the same place. Another feature that can be useful would be to allow users to make requests to change tasks. Sometimes, a user may not be satisfied with a task that they are assigned, so a feature to request their boss or manager to change their task can be useful for them.

CONCLUSION

We believe that our product will help many software engineers that are working on teams stay on track with their tasks. So far, we have a prototype with most of the basic functionality implemented. The things that we have to complete to make this a finished product would be creating a user login system, an online database so users can save their taskboards and work on them from any location or device, and to touch up on some of the graphics on the GUI. In the future we would primarily focus on implementing these features and then maintaining our software. We think this product will improve the efficiency of many software engineers in the workplace in terms of getting their work completed as well as improve collaboration between teammates.

REFERENCES

[1] Google. https://tasksboard.com/