Team 4 - Sprint 1 Retrospective

Alex Hardewig, Zach Johnson, Benjamin Kahlert, Eric Vondrak, David Wood, Ian Zanger

What Went Well

There were a lot of risks associated with this sprint, including the unfamiliarity of most of our team members with the languages and database we were using, our inexperience working in larger teams, and the juggling of everyone's schedules. Despite these hurdles, we were highly successful in our first sprint, for several reasons.

We completed all but one of the user stories that we promised. Every core user story was completed, ensuring we have a good foundation on which to build future features. Users can register with their Google accounts and create personal tasks and projects. Projects can be created and customized, users can be added to projects, and rudimentary project settings can be edited. Users receive notifications that they have been added to projects, and can accept or decline. Tasks can be created in group projects with different deadlines, priority levels, and estimated hours to complete. Tasks can be deleted, completed and moved to the archive, and moved back to the active task list from the archive. Most importantly, every one of these features is synced with our database in real-time, meaning that as users edit settings and create/complete/delete tasks, these changes are immediately seen by all other users on that project. Essentially we have a fully working product that could be enjoyed by many users already. In Sprint 2 we can build on this strong foundation to add additional features that will differentiate our product from other existing applications.

Another success of this sprint was the learning of multiple new languages, database systems, and packages. Few of us had ever worked with React, Re-base, or Firebase before, and some of us had no experience with HTML, CSS, or Javascript, either. Despite this, we spent our first week researching the structures of these tools and how we could use them to build a modern web application. Many of our hours that first week were simply spent getting up to speed so we could attempt to complete a user story, rather than diving right into user stories. We persevered and managed to get almost everything we promised completed after this setup period.

Not only do the features we implemented work, they work well. There are no major bugs with our implementations and everything is coded rather cleanly. If a user were presented with our application in its current state, they would have no major frustrations that would turn them away from using it. Our GUI is also quite polished for only the first sprint and looks simple yet professional. The GUI elements of the features that are currently working are in a near-final state, and will need minimal tweaks in later sprints.

A final thing that went well in our first sprint was our team collaboration. We meet twice a week to discuss our progress on the project, and our Sunday meetings often turned into group programming sessions. Members would bring up issues and roadblocks they encountered during the week, and other members often stepped up to help solve these bugs and clarify design decisions. As every member researched different topics throughout the sprint, there were often members that were “specialists” in one certain area, and they provided help in this area to others when needed.

What Didn’t Go Well

Although our first sprint went well overall, there of course were some aspects of it that leave room for improvement.

Our biggest failing was not accomplishing every user story we originally set out to complete. We moved one user story to a later sprint that involved the sorting of tasks based on priority level, deadline, and age. We decided to delay this user story over the others because it was not a core feature. It will be a useful feature to have, but everything else we worked on in sprint 1 was foundational - creating projects, adding users, creating tasks, syncing projects, etc. These foundational stories were essential to later sprints and were thus prioritized. We had to delay this user story simply due to a lack of time. There was quite a bit of startup time involved with this sprint, as we had to research how the languages we were using worked and how to integrate our database and packages of choice. This meant that even simple user stories took longer than expected. This was especially an issue with the syncing of project information with the database in real-time. This feature underlied nearly every user story as a necessary, core feature of our application, and was not simple to implement. We had to figure this out before we could even start working on some of our user stories, which didn’t happen until the second week, giving us a

later start than was ideal. We ended up having to work much more in later weeks to make sure we finished everything we could.

Another issue that cost us time was the onboarding of less-experienced members. Some members with less experience understandably needed more time to get up and running with our chosen technologies, and to implement the features they promised in the Sprint 1 Planning Document. This meant that they had a harder time achieving all of their promised objectives in the amount of time they had to work. More experienced members also had to spend time helping them get up to speed, which detracted from their own timelines.

Another issue was the inaccuracy of our time estimates. Simply put, they were way off. Some features took less time than expected, but a majority of features took longer or even much longer than predicted. This was due to the issues mentioned previously with onboarding and research. This meant some features assigned to certain members had to be completed by different team members who weren't as busy with their assigned tasks. Although we still accomplished most everything, this situation isn't ideal as it makes it more confusing to track who is working on what.

How We Will Improve

For Sprint 2 we will attempt to address some of the issues outlined above, and prevent new ones from arising.

We will attempt to address the "startup time" issue by keeping in mind how long this took us in the last sprint. It is inevitable that we will need to do research on how to implement certain features, but we will try to plan for that more explicitly and include it in the time estimates of tasks. If we discover something that we think will be helpful to the entire group, we will share it and teach it to the other members. We will also try to complete this research as soon as possible so we don't leave most of the actual programming to weeks 2 and 3.

We will aim to improve our time estimates of tasks by incorporating research time, applying our experiences from Sprint 1, and discussing more thoroughly the underlying systems that a user story requires. We will try to brainstorm the issues that could arise as we tackle different user stories, and discuss which user stories rely on others so that we can complete the most important ones first. Using our improved estimates, we will distribute the workload of our next sprint more evenly

and accurately, reducing the need for members to take on others' tasks. We will also avoid taking on too much work in one sprint as a team, so that we can complete all of the user stories we promised. This will inevitably be an ongoing issue due to the difficulty of estimating how long certain tasks will take, but we will aim to improve as much as we can as we gain more experience.

Finally, we will hold more group programming sessions, and plan to have this be the main point of our Sunday meetings. These sessions are highly productive because we can discuss design and implementation issues as we run into them, help each other with tricky bugs, and make sure no one is falling behind. There were also multiple occasions last sprint where one member knew an easy solution to something another member had been struggling with for a while. More of these sessions will save us all time in the long run and help us achieve our goals for the sprint.