CS 407 Fall 2024 | Team 4

Course Clash | Sprint 1 Retrospective

Team Members

Thomas Carsello <tcarsell@purdue.edu>
Jason Bodzy <jbodzy@purdue.edu>
Zachary Heskett <zheskett@purdue.edu>
Sooha Park <park1343@purdue.edu>
Celia Patricio <cpatrici@purdue.edu>
Sergio Alvarez <alvar166@purdue.edu>

What went well?

Something that went particularly well during our first sprint was our use of version control and branching within our git repository. There was never an issue of corrupting our master branch due to misuse of the version control system. Everyone was able to integrate their contributions seamlessly through the use of feature branches and pull requests. This was particularly helpful later on in the sprint when there started to become a larger surface area on which several user stories could be worked on simultaneously, as opposed to the early parts of the sprint where many user stories were prerequisite to others.

Another aspect of our sprint that was effective was our communication within our WhatsApp group chat. Team members were easily accessible and highly effective communicators over this medium, which made it easy to coordinate task delegation and meeting times. We also found that using When2Meet was a great way for us to find meeting times that work for everyone in our group, given that we are all on largely different schedules.

Finally, while we did not do this on purpose, it was quite effective that the team members developed their own specialties within the project, and that made task delegation and feature delivery quite efficient and straightforward. For example, Sooha was a UI/UX guru who we could always depend on to make the application more user friendly and more of an attractive experience. Similarly, Tom was focused mainly on the backend and data models of the application, which made it efficient to fix issues we were facing.

What did not go well?

Fortunately, we got all of our user stories implemented, so we do not have any specific user stories to mention. However, some of them came down to the wire a little bit, as they were not closely associated with some of the other user stories which had logical connections between them. It was easy for someone who implemented user account creation to implement the log out feature, for example, because that person was already familiar with how that part of the system functioned. When it came to user stories that were more disconnected, like the color-scheme user story, communication on those slipped through the cracks as there was not an immediately obvious person who would be responsible for it.

Additionally, we found it difficult to have regularly occurring meeting times each week. This was largely a function of us each having different schedules that do not align very well. At the same time, there could have been some mechanism used to have regular status updates even if that came in an asynchronous medium rather than live meetings. More could have been done on that front.

How should we improve?

The first way that we believe our processes could be improved for Sprint 2 is the establishment of some sort of recurring meeting/informational mechanism to share periodic status updates. While we were good at communicating directly through our group chat, it would have been nice to have a centralized location to see exactly what the status of each person was at the time rather than having to individually message and get a response. We know that there is not a good time during the week where we can all meet together, but it would be possible to set up two meetings during the week where different subsets of the team are available. This way we could have a more standardized setting for administrative communication. People who were not available for one of the meetings that week could be looped into the previous discussion during the next meeting for which they are available. Overall, this would expedite the process of getting every member of the team on the same page.

A second way that our group could improve for Sprint 2 is the potential to assign user stories by technical expertise as opposed to functional similarity. For sprint 1, we assigned user stories in groups that seemed functionally “close” to each other, but implementing those stories often required an extremely vertical effort on the database, API backend, and frontend UI. Instead for sprint 2, we could assign user stories such that team members are more localized into their specific technologies. For instance, a frontend-focused team member would not have to create database migrations, and a backend developer would not have to be writing CSS. Of course, there will always be cross-functional responsibilities, but we can do a better job of minimizing these in Sprint 2, as they seemed to be one of the major slowdowns for Sprint 1.