Team 22 : Memoji

Sprint 1 Retrospective

Team Members

Scrum Master - Jonathan Poholarz (jpoholar@purdue.edu)
Maxwell Jones (jone1268@purdue.edu)
Manoj Polisetti (mpoliset@purdue.edu)
Andrew Ring (amring@purdue.edu)
Delun Shi (shi272@purdue.edu)

What Went Well

Our first sprint focused on setting up the Networking infrastructure and connection/disconnection of players from the hosted servers. Communication between our two Networking team members went well, and the design of the API on the client and server sides matched up easily and flexibly to account for potential needed changes. In addition, defining clear constants to be used in the Networking and message distribution helps to account for typos that may occur.

In this sprint, many of our team members were first learning to use the Godot Engine with its node structure and scripting language GDScript, and they were able to pick up the workflow and language fairly quickly which helped to increase momentum throughout the sprint.

In total, the user stories we wished to accomplish went well. We were able to implement all of them before the deadline, including fairly parallel work by our team members. Due to Godot's friendliness with source control systems, we were able to avoid many merge conflicts, only having to deal with a few of them throughout the entire sprint. Many of the merge conflicts we did have were often only a couple lines in a single file.

What Did Not Go Well

Despite the success in completing our user stories, we did run into a few issues that could be improved upon in our next sprint. First, the pacing of work was rather inconvenient with regards to testing. Although many tasks were started early, a lot of the necessary implementation code was not completed until late in the sprint. Because of this, testing much of the application was near impossible during the first few weeks. Therefore, very little testing was done until late in the sprint, which is another issue.

Second, our task assignment frequently resulted in one team member designing a screen and another team member implementing functionality in that screen. This created confusion as to which team member was responsible for what code as well as difficulty trying to understand the variable naming schemes and intended functionality. Inconsistent code styling and naming conventions

further confused members of our team throughout the sprint. To add to this, some of our core source files such as GameStateManager.gd and ScreenManager.gd were forgotten from the story task planning. This made implementing functionality, such as sending messages to the server, more confusing than it needed to be.

Lastly, we encountered some odd network bugs such as the lengths of messages being sent separately from the body of the message and the message protocol of Godot and of the Node.JS code being different. These led to further hiccups in determining the correct way to write our networking communication.

How To Improve

In order to improve our workflow for the next sprint, we have decided to make some improvements. First, we will be dividing up our stories and tasks into larger portions, mostly having a single person do almost everything in a given screen, in order to avoid ambiguity in who exactly is responsible for which features. This means we will be trying to assign the screen's design and implementation to the same person where possible.

Second, we will be starting our testing process much earlier, attempting to test completed tasks on a weekly basis and merge their code more frequently into master. In addition to refactoring messy code and agreeing on a more uniform coding style among team members, we believe that these changes will make our codebase less buggy by the end of the sprint and reduce the workload in the last week. Overall, this should help with our pacing issues.

Lastly, we have agreed to communicate in our group chat ahead of time when we are unable to make meetings. This will reduce the time spent waiting for teammates to arrive to meetings who aren't going to show up. We can also better plan our meetings based on the people who will be there (though we're hoping for overall better attendance to our planned meetings in Sprint 2).