Team 22 : Memoji

Sprint 2 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 second sprint consisted mainly of gameplay implementation including systems for responding to prompts, voting on prompt answers, and tallying up votes in order to score players. Overall, we think we were able to implement all of these systems fairly effectively in our second sprint, completing all of our user stories we had planned. Because work was better divided among team members in this sprint compared to the first sprint, we were able to reduce the overlap and the merge conflicts that would cause us development headaches. It also helped that our team has become fairly familiar with the Godot engine, it's scripting language, and its workflow. This reduced the number of problems and questions we ran into when developing. By the end of the sprint, we could tell that the project was finally taking shape and coming together as a cohesive application.

Our demo for the second sprint went more smoothly than the first sprint. We were easily able to show off all of our user stories in a reasonable time frame, and we did not experience any unanticipated crashes or problems. Because we arranged our Sprint Planning document in the order of presentation, we were able to announce which stories we were presenting when which helps to keep people unfamiliar with our presentation on track and able to follow our documentation in addition to the visual aide. Although we did not present an in-depth overview of our project in this second sprint, we were able to briefly summarize our project and goals to be met. In total, we believe the presentation went well.

For our final presentation, we plan to incorporate many of these same techniques used in the second presentation such as ordering our documentation to follow our script. In addition, we will prepare a useful powerpoint presentation highlighting the engine we have been using to construct our project and the design decisions we made in our architecture. We believe this information will be informative for our viewers as Godot is a fairly unknown engine. We also plan to assign team members more specific roles for who will be talking when so that everything we want to be said can

be said without interrupting each other, something we struggled with slightly in our first couple presentations.

What Did Not Go Well

Although many things went well, we still ran into a few issues this sprint, particularly with testing our stories. First, the exact implementation details of our prompts/answers and how they would be stored were not really planned or documented ahead of time. When we had multiple people working with these unimplemented data structures, many issues and questions arose, slowing down our development as no one really knew what they needed to code.

Second, testing became quite a hurdle this sprint. Overall, testing was difficult and became a team process, especially late in the sprint, as testing multiple players concurrently is slow to do alone. In order to speed it up, utilizing all team members when testing was almost necessary. Mostly, test runs were long because we cannot save the state of the test part way through and must re-enter usernames, avatars, responses to prompts, etc. for every test run. Similarly, input to different screens was not simulated, so nearly all testing was forced to be done manually (though, we're not sure how much time simulating the input would have really saved us in the end due to bugs arising across multiple screens).

In addition, because testing one team member's code was almost always gated behind the success of another team member's code, many screens and situations were left mostly untested until the preceding screens were working. This produced unnecessary slowdown which we should intend to avoid in the future. Although we had planned in the last retrospective to test progress weekly, because of Spring Break being in the middle of the sprint, teammates unable to make meetings due to travel, and progress not being made until late in the sprint, weekly testing that we had hoped to do did not really happen. To add to this, because teammates did not simulate input to their screens, much of the bugs gated behind success of other screens would have made our weekly testing next to impossible in the state the code was in.

Lastly, since not all of the code was pushed with it needed to be at a couple points, we were unable to continue with our testing process until the missing teammate got around to committing their code. This put us a bit behind schedule a couple times and required a longer night of testing before the presentation.

Although we were able to improve our proposed user story assignment changes and meeting communication improvements suggested in the first sprint retrospective, we fell far short of implementing our proposed testing changes. Code was still not merged as frequently as we would have liked, which we plan to combat in Sprint 3.

How To Improve

Going forward, we will be assigning Manoj and Delun to be in charge of following up on our proposed changes, specifically to testing, as their code will depend more heavily on other team members finishing their sections and fixing bugs that may prevent access to the later screens Manoj

and Delun will be working on. They will have the responsibility to highlight issues in our meetings related to our development process if any arise. This will include failing to meet expectations set out in this document for Sprint 3. If or when this happens, they will raise the issue in the meeting so it can be discussed by all team members involved. Of course, non-designated team members may also voice such complaints or notices, if needed, at these same meetings.

In addition, we will be testing our code more frequently this sprint in order to catch any bugs introduced by the significant refactoring Jonathan will be primarily working on throughout the sprint. We will also be pushing code more frequently throughout the first couple weeks compared to before where often code was not committed into the main testing branch until late week 3 if not week 4. The refactor will aim to simplify messy code and screen structures as well as organize some of our files into folders and sub-folders to reduce the clutter in the high level directories.