James Gossling

Shivam Vashi

Andrew Pester

Cody Tomkins

Through implementing slapjack and dividing up the tasks, we all set out to work on our tasks and decided to regroup later to combine everything we had. When we met up to combine everything, we ran into multiple merge conflicts with code that had overlapping methods that we implemented. Because some of our tasks depended on functions that were not pushed to the git, we had multiple people working on an apply method for SlapJackInit for example. While we fixed this in the group meeting, in the future we would probably want to divide not just the tasks, but make sure to communicate any dependencies on methods that multiple people would need, and how each person would modify them so our merge process becomes more efficient. Another fix to this would be to have the same people work on tasks that depend on each other to increase cohesion. Our main design worked effectively though, as we decided to base our code's structure similar to the structure of the already working Pickup 52 code, only changing necessities such as creating two piles, and consistent card locations and orientations.

Our team ended up sticking to our plan pretty much exactly in regards to the classes we decided to make. We didn't have to make any new methods or classes on the fly. It was super helpful being able to get on Discord together and help each other debug the issues we were having. One thing we could change about our design is when the game is about to end, have the end move happen once a player plays their last

card. As of right now, the winning player has to click on their deck before the game can end.