Final Project Reflection:

Our project idea began during a poker tournament we attended a few weeks before the first final project deadline. We both enjoyed the poker tournament and realized that studying up on poker theory to program a poker evaluator would teach us about poker and programming at the same time. On the poker side, it would allow us to more thoroughly appreciate poker by showing us how to optimally play the game. And on the programming side, it would teach us how to build a way to create a live renderer and create an evaluator. To start, we read up on poker theory to create a good evaluator.

Poker consists of five rounds of betting with differing amounts of information at each round. We began by studying pre-flop theory, or how to correctly bet before the flop based on your position relative to the dealer, the cards in your hand, and how other players bet. We then looked into various evaluator libraries on GitHub for inspiration on how to build an evaluator.

Currently, our project consists of a program that stores information about a given state displays a rendered version of the state, and makes rudimentary evaluations of the state to assist the user. The system can accurately display a game of poker, with a few minor bugs in complicated turns (eg: player 1 bets, player 2 bets higher, player 1 bets even higher, player 2 folds doesn't correctly display on the screen). We are currently using an evaluator library from Github called Treys, which takes in a given hand and outputs the percentage of all possible hands that your best hand beats.

Given more time, we would do two things. First, we would implement a better evaluator that can provide precise advice based on the current hand and previous hands. This would involve storing key information from each hand in a new class to pass to the evaluator, as well as finding a new evaluator with more capabilities. Second, we would improve the renderer's

capabilities. This would include improving resolution, displaying more information (eg: the evaluator's evaluation, player distinction, player stack size), and fixing the minor bugs with complicated betting.

Looking back, we would spend more time designing our evaluator and less time trying to store superfluous information in our state classes. We ended up using a fairly rudimentary evaluating technique, which is why we also would primarily work on improving this part of our project given more time. As for our state classes, we were too optimistic in how much we could get done and spent too much time implementing ways to store useless information. For instance, we currently store player stack size but never use it in our hand evaluations. Since we had to finish the state classes before working on the evaluator, we left insufficient time to complete the evaluator. Despite this, we are very satisfied with our project and hope to continue it on our own time in the future.