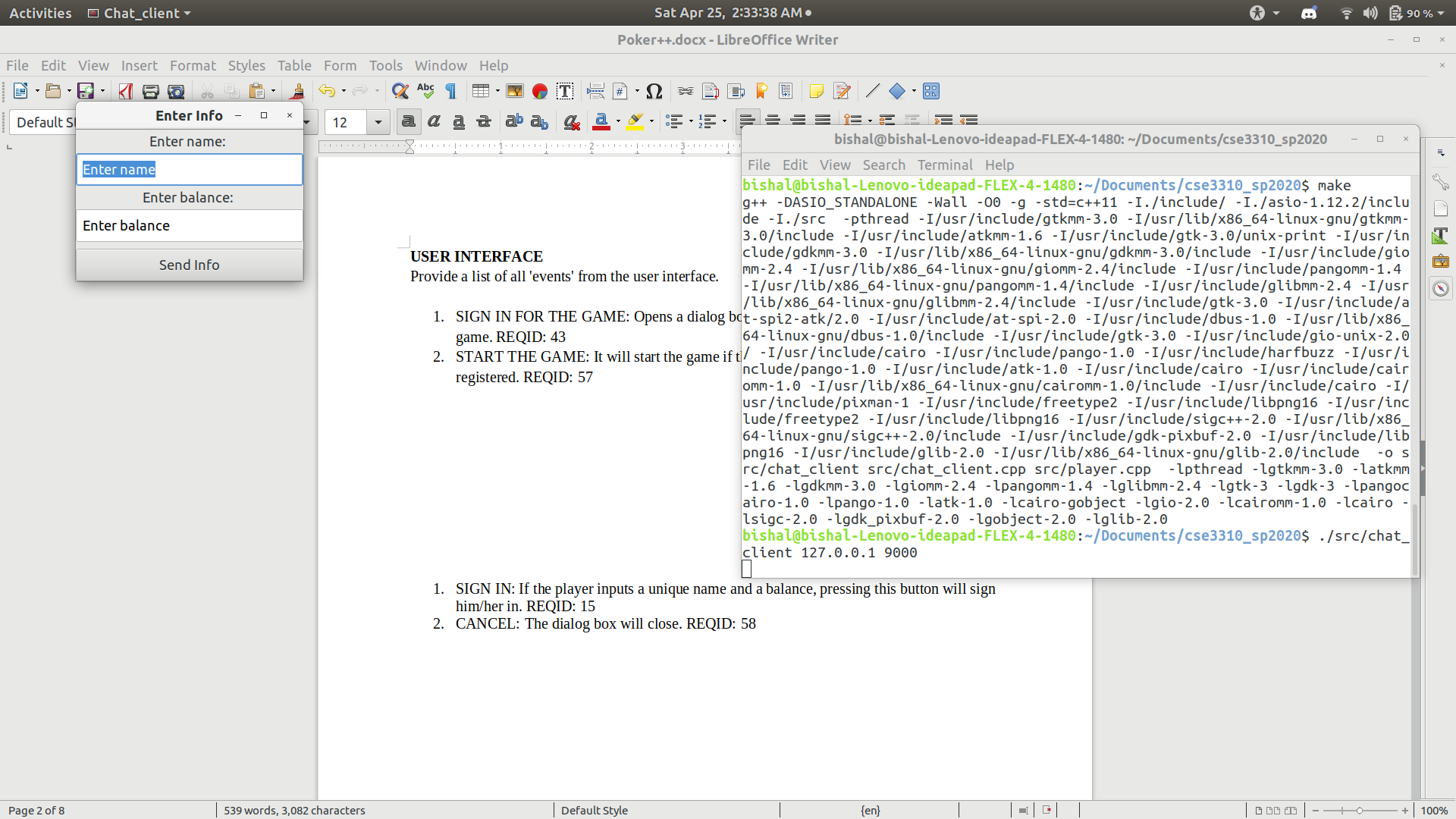
**INTRODUCTION**

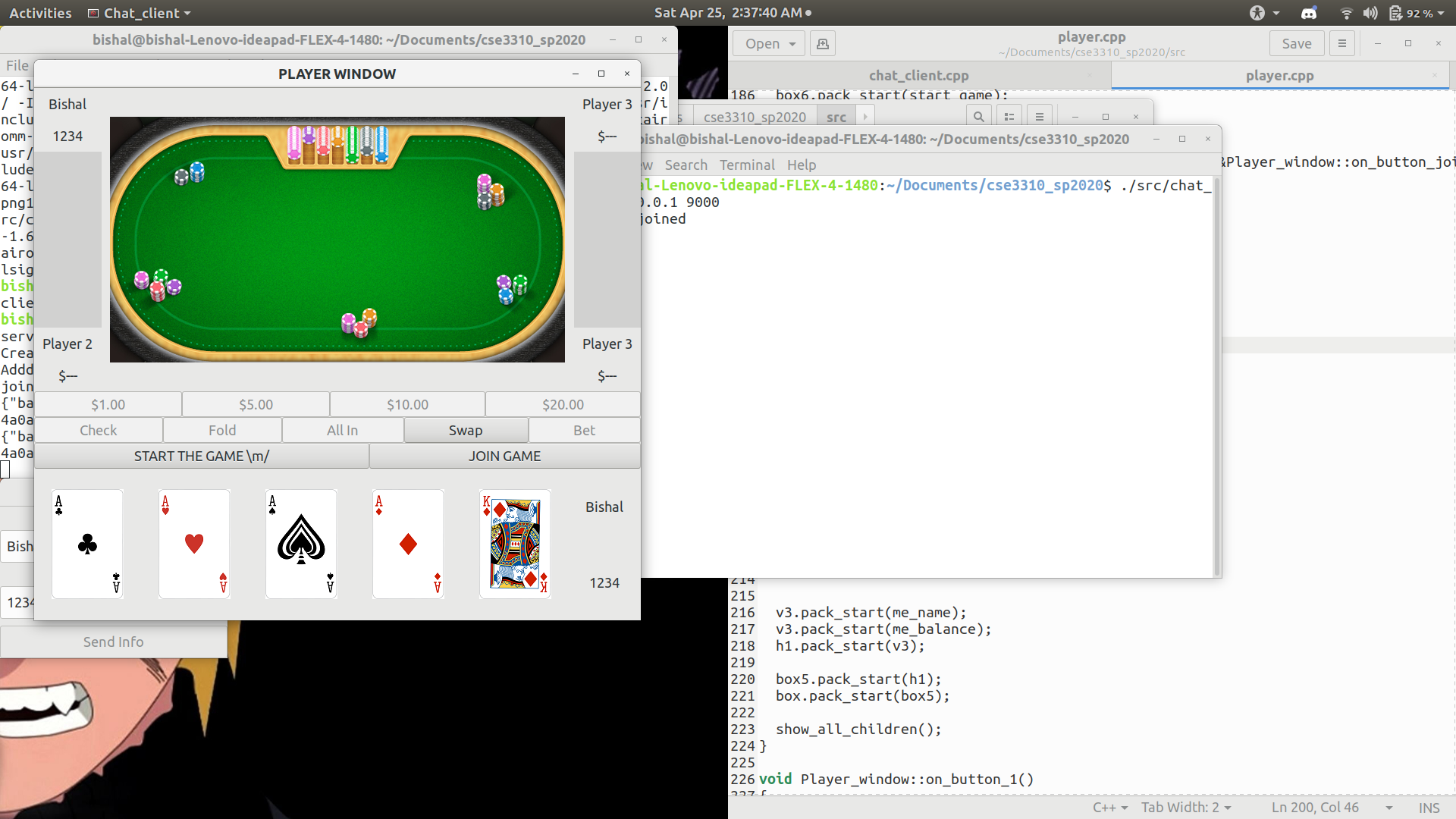
In our group we have Bishal Adhikari, Kenneth Garza, Jenny Nguyen, and Armaan Zirakchi. For this project we are creating Poker++ which is a virtual game that simulates Five Card Draw Poker. This virtual game will have a server program that runs the dealer system, and the dealer system will be responsible for the majority of the functionality of the game. There will be a minimum of at least 2 players at a time and a maximum of 4 players at a time. Each player will have 5 cards in their hand at each point of the game.

Players are allowed to switch cards once during each round if they do not like their cards. The hands will be compared and will be ranked in order according to conventional poker rankings. The player with the highest ranking hand will be the winner for that round and will receive the pot. The pot is created from all of the players' bets. If a player has no money left they are out of the game and become a spectator. The winner of the overall game is decided when only one player is left standing.

**USER INTERFACE**

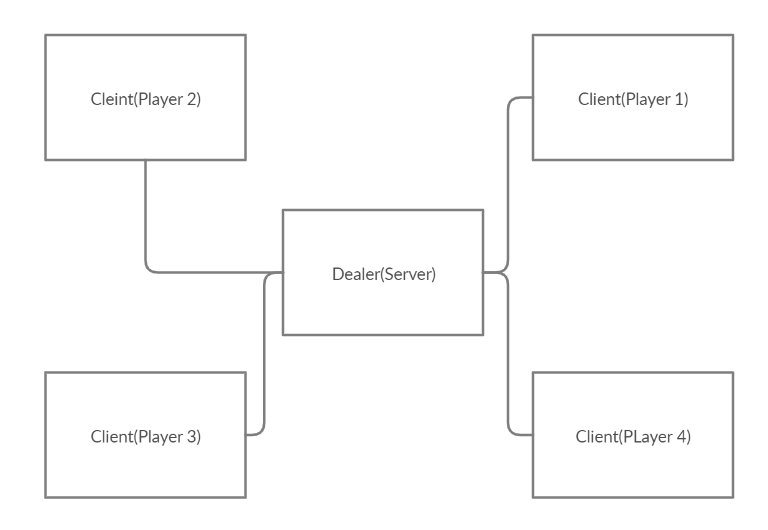
Provide a list of all 'events' from the user interface.

1. The Entry Info window:- The player enters their names and the balance they want to put in the game.
2. The Player\_Window:- This is where all the game is being played and each players sees their different cards.

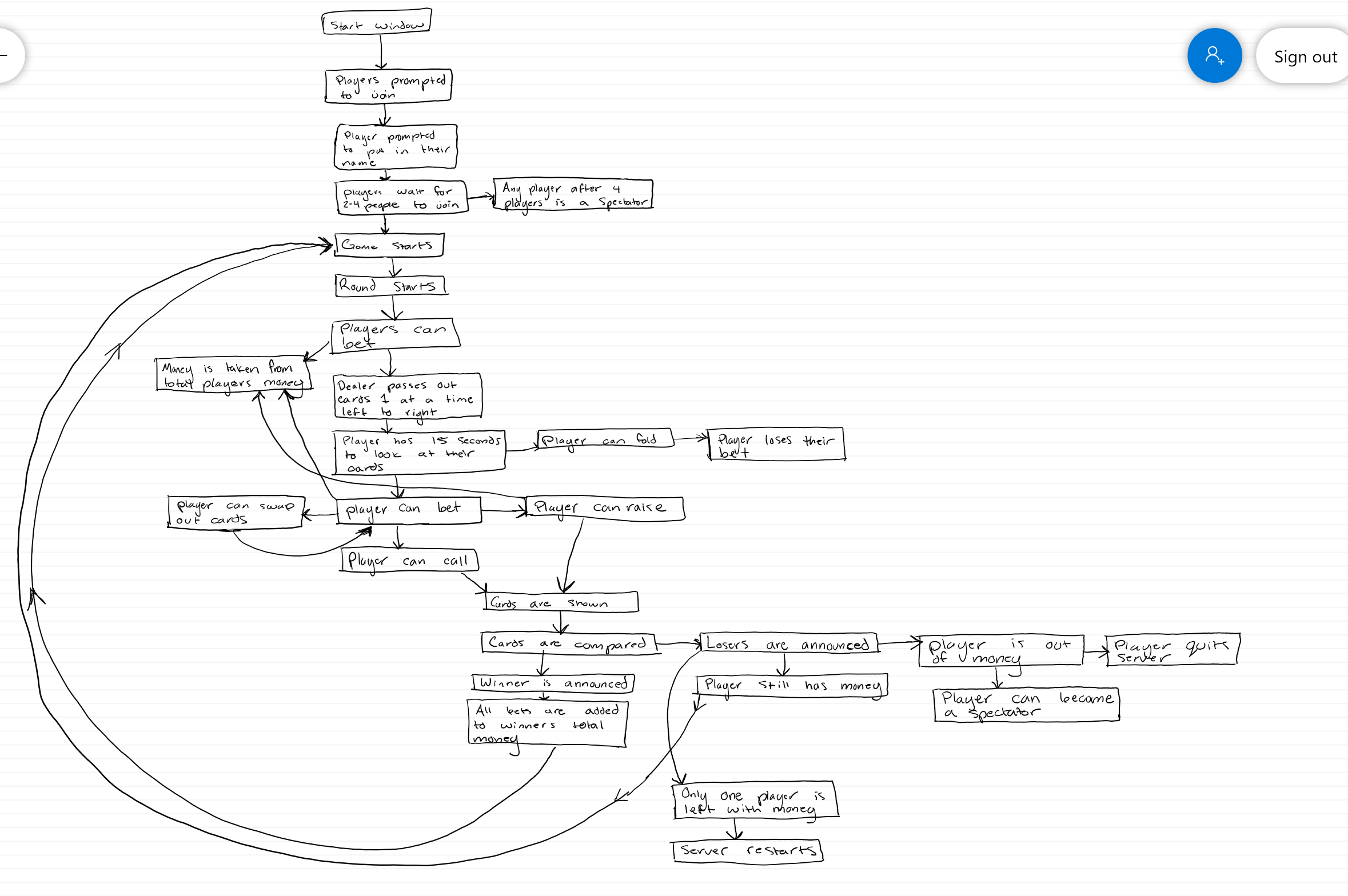
****

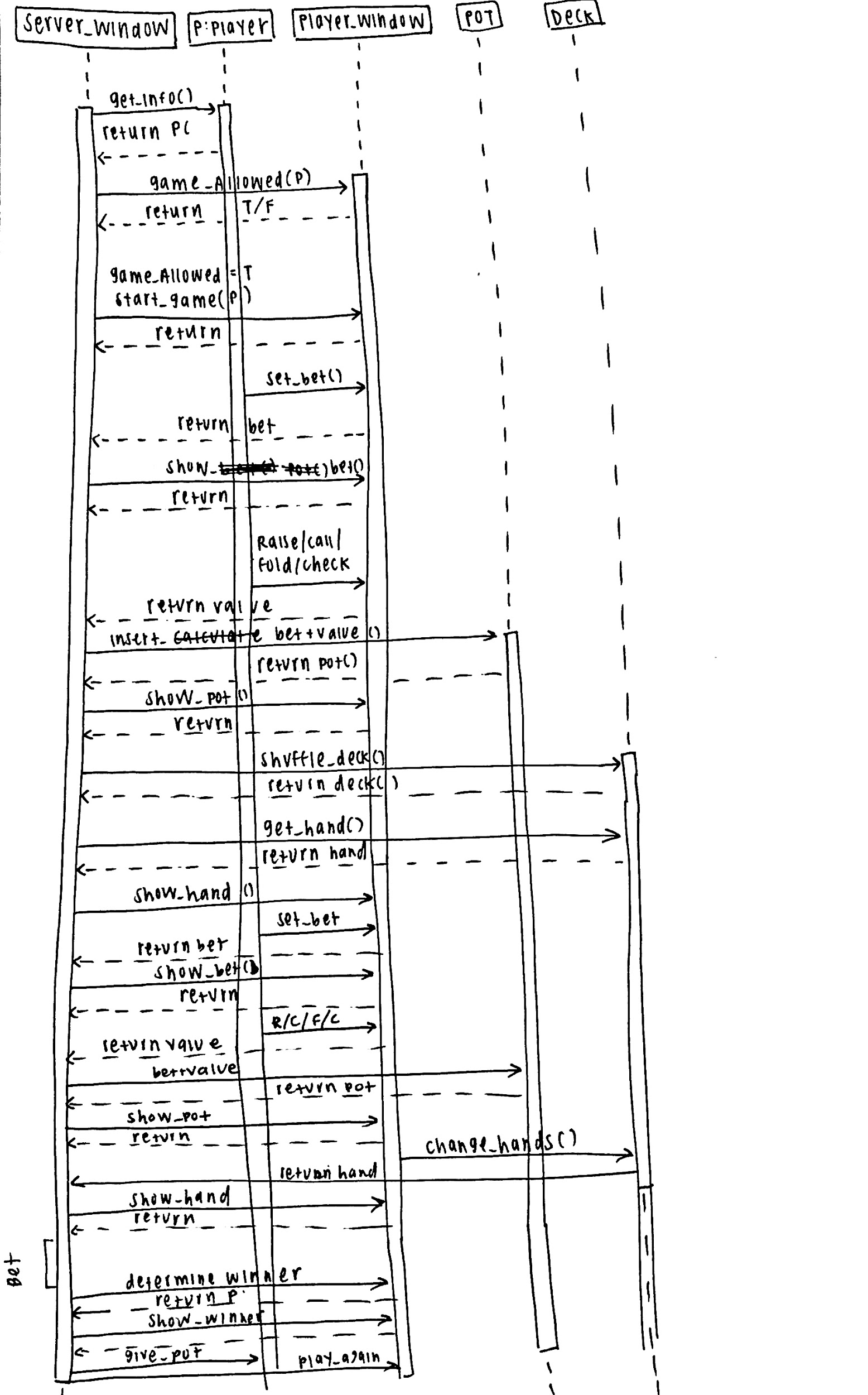
**SYSTEM UML DIAGRAMS**

CONTEXT DIAGRAM



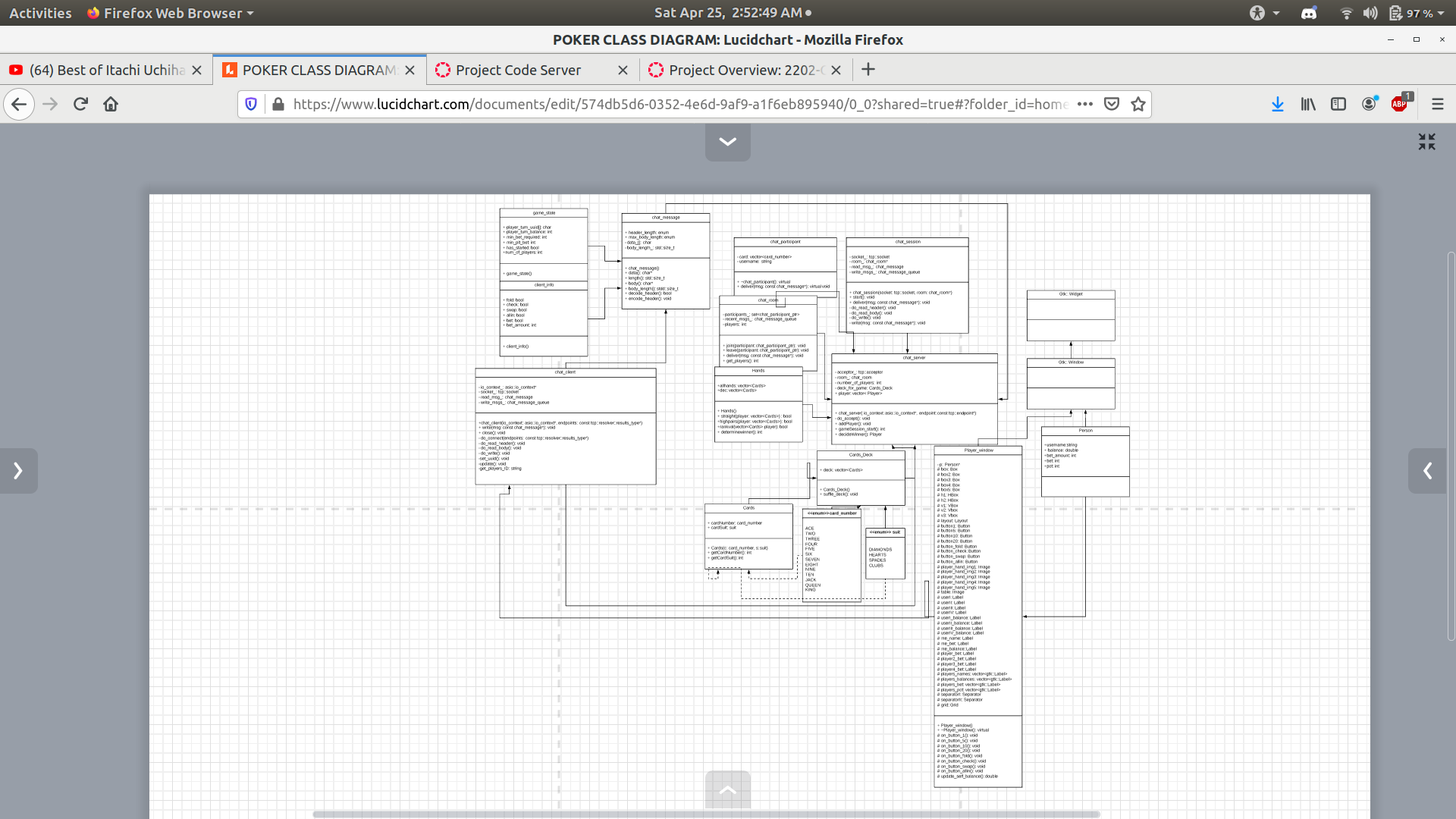
STATE DIAGRAM OF THE OVERALL GAME (WILL TURN INTO CODE)



EVENT DIAGRAM SHOWING INTERACTION WITH THE USER

**CLASS DIAGRAM**

**UPDATE: for a clearer version https://www.lucidchart.com/invitations/accept/cf4cc9d3-da22-43a7-a2aa-f847840edc64**



**DIFFICULT MECHANIZATION**

Some paragraphs describing some of the more complex problems encountered during the software design activity. In most instances, an event diagram will also be required. For this project, some candidates would be:

· Maintaining continuous communication with the server.

The software stablishes a complete connection with the server and each other. However, after the connection is stablished, the connection halts. We are still working on ways to fix this problem.

· Updating the cards in the code

Since the communication between the clients and the server is not working properly, the cards to be displayed are not working properly.

* Shuffling the cards to be truly random
  + Using srand to generate random numbers results in sometimes generating the same shuffle that will then give the players the same cards even though it is technically new deck that has been generated.
* Determining the winner and dealing out the cards
  + To make it simpler the class that determines the winner of the round also deals out the cards technically. But this also caused problems for allowing the other classes access to this. The way the code determines the winner is assigning an integer portion to each of the ten different hand ranks and then based on the high card within that hand adds a decimal value to that overall ranking value. The player with the highest-ranking value will win the round. In the case that two players have the same value the player that entered the game first will win.
* Swapping the cards
  + We were unsure of how to make it so the player could choose the cards that they want to swap, but instead of selecting the images of the cards that are displayed they must choose the checkboxes that pop up when the swap feature is pressed. We had some difficulty keeping the swap function random and changing within the given 52 deck of cards, and then reupdating the players display.