My card game is made up of 4 major objects which are Card, Deck, Game Manager, and Player. The Card object is made up of 2 enums which are suit and class.. The enum was a design choice that allowed the cards to have a small memory size but also allow the ability to add more suits and ranks to scale the game as needed. In addition the Card class overloads the comparison and ostream operations to allow easy printing of each card to the console and easy comparing of each card according to the game rules.

The deck object is a vector of cards. The deck contains operations from drawing cards from the vector, shuffling the vector, and querying information about the size of the vector. A vector was chosen due to the ability of it to have varying sizes so we can just pop a card off the vector instead of worrying about what index of an array we were at to draw cards.

The player object represents a player that has a card and a boolean variable to check if they drew a card. The player object is small but allows for the ability to scale different features if needed to associate with the player.

The final object is the game manager. The game manager holds all the logic and management for the game. The data the game manager has is the deck it will use during the game, a vector of players, and an atomic boolean that allows for functions to signal when a round is over. The game manager handles rounds through the atomic boolean. If the timer function ends the atomic boolean is set to true and the round is over, or if all players are done drawing the player input sets the atomic boolean to true and the game is over.

BUGS: The player object will accept one extra input before calculating the winner if the timer is over. This is due to getline halting the program until input is given to the getline. This could be fixed through a redesign of accepting player input through a library like ncurses where we can choose to get input in a non-halting manner.