SYST 17796 DELIVERABLE 2
DESIGN DOCUMENT TEMPLATE
Overview

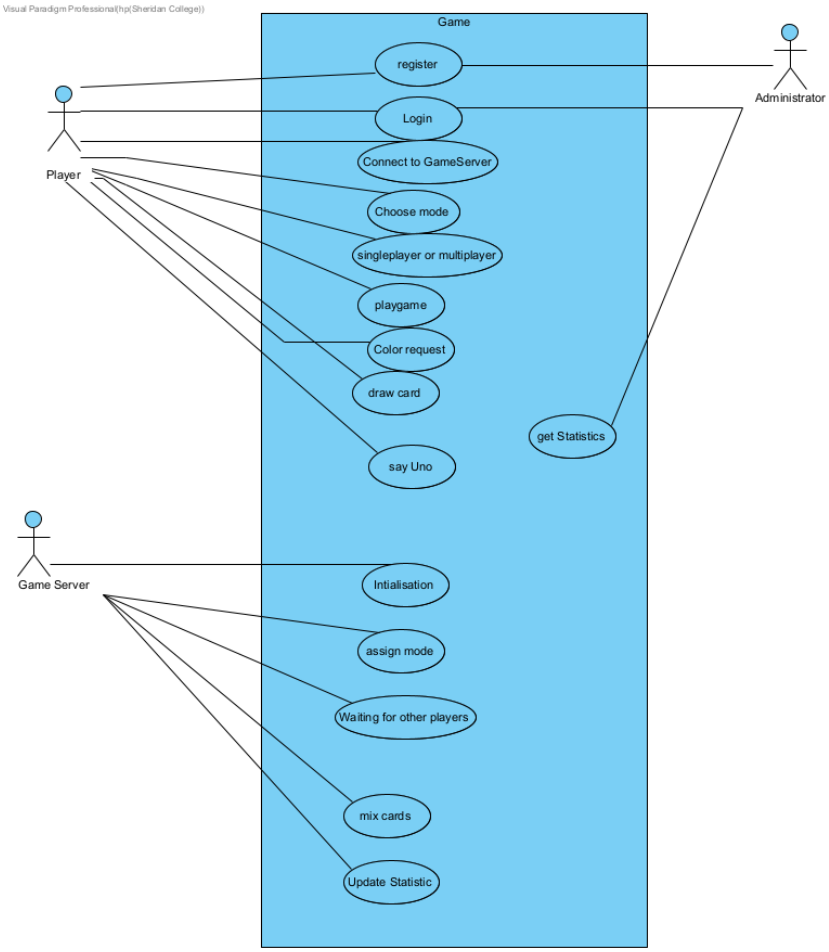
1. Project Background and Description

Initially both the player and the game server will be given 7 cards. The number of cards left with each player will be also be shown. The game will only have several round until either player is able to score 300 points. After the score limit is reached , the player would be given the choice of continuing the game or not. The points awarded at the end of a round will be decided based upon the number of cards left with the opposite player and the value they hold. If the player wants to leave the game early he can declare the current round as final round and whoever wins it will win the game. The game server will keep records of the number of games the player has played ,won and loss.
There would be a help section where new players could get to know how the game is played and what does each card mean

and what value they hold. There would be an exit button which will allow the user to leave the game but this will mean that the user will have to consider that he loss that game. There will be an uno button which the player will press to indicate that he is left with only one card, but he should press it before the countdown is over.

2. Design Considerations

3. Use case Diagram



4.Use Case Narration

I. Use case: Register

Use case name:	Register.
Scenario:	Needs to register for playing game.
Triggering event:	New User wants to register for playing online game.
Brief description:	Player needs to register online by entering name and password to play the game.

Actors:	Player	
Related Use Cases:	No related use cases.	
Stakeholders:	Draw card, play game and get Statistics.	
Preconditions:	Register System must be available for the Players.	
Postconditions:	Player must register the name and password. Multiple registration is not allowed. Player can add member and can add their teammates.	
Flow of activities:	Actor	System
	1.Each player indicates to register the basic information and password. 2.Player can add members and teammates.	1.System creates a new player. 2.System prompts for player's Password. 2.1 System prompts to add member or teammates. 2.2 System generate a message about registration details.
Exception conditions:	1.Password is not valid. 2.Basic information is incomplete about player.	

II. Use Case: Get Statistics

Use case name:	Get Statistics

Scenario:	Get Statistics is used to show score.	
Triggering event:	Statistics is used to calculate the score of players.	
Brief descriptions:	Statistics must record each player' s score, record the score in his account and display his position in the game.	
Actors:	Administrator	
Related Use Cases:	Register, Login	
Stakeholders:	Server,Update Statistics	
Preconditions:	Player and account must exist. Administrator must exist.	
Postconditions:	Statistics is created and associated with player. After every game the statistics is update on player's account.	
Flow of activities:	Actors	System
	1 Administrator must check in the player id. 2Adminsitrator must update the status about the score.	1.1 System looks for the player's id and information. 2.1 System must look for the player's account and update the score of the game.
Exception conditions:	1.If player leaves the game in middle, his score will not be updated. 2.Player must register for game if they want to see the score of game.	

5.Design class Diagram

The First class is Deck , all other classes will inherit this class. Deck contains a no argument constructor deck which will initialize variable like cardsInDeck(gives the number of cards in the

deck),and also declare constructors like pause game ,reset, drawcard. The hand class will inherit the deck class. It contains method to start the game, shuffle the cards, and also getting statistics of player and the game . For the shuffle method we would use random numbers to generate different values between a fixed range. The second class inheriting the Deck class is Card class. This class is mainly responsible for storing the cards and getting their colors and values. It contains two subclasses which are enumeration. The color class would specify the different color cards and also the wildcard. The value class will give the value of various cards.

Relationship between value and card class-

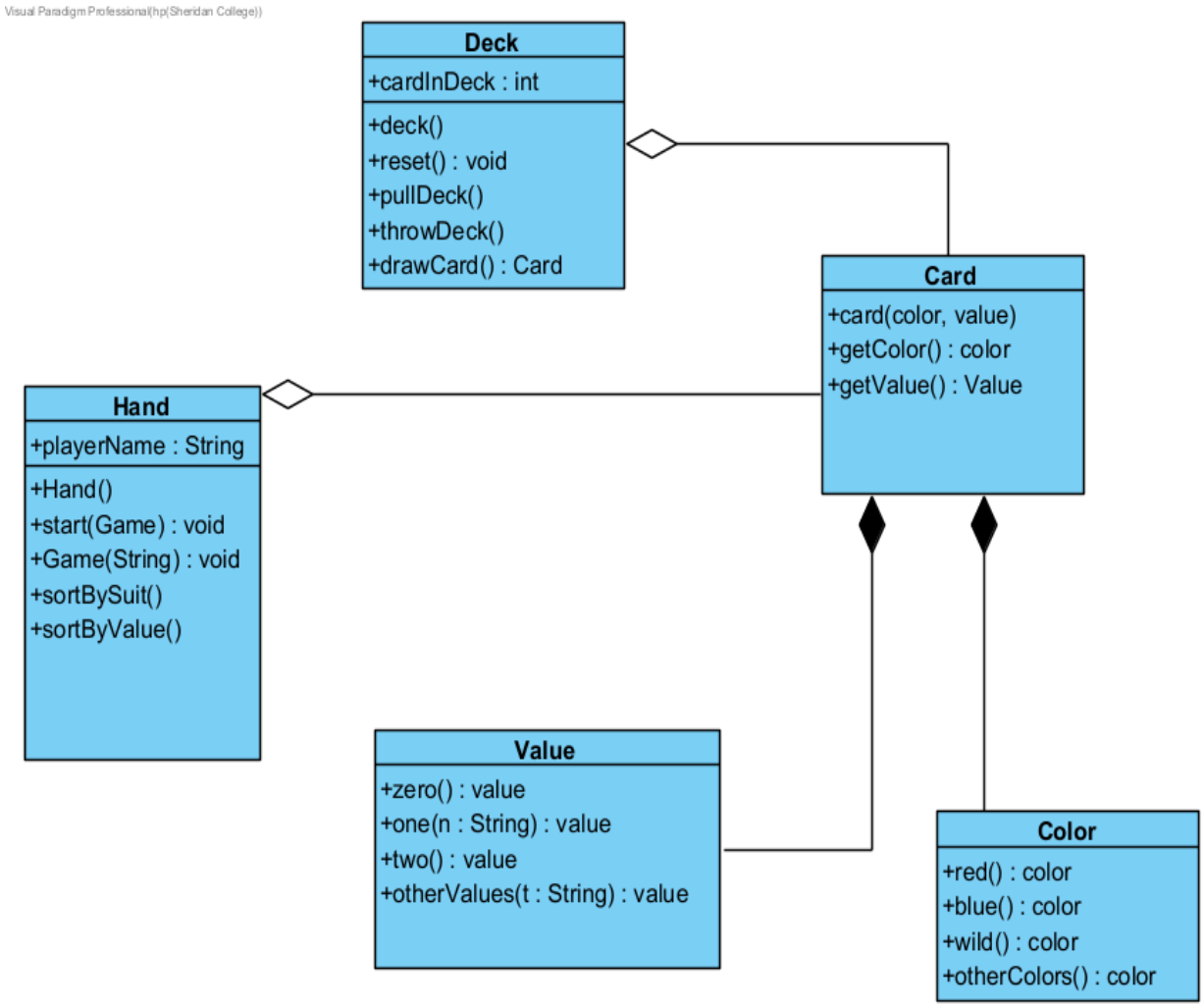
Many cards can have the same value ,and there will always be a value associated with them(multiplicity =1*),however, one value is associated with one card(multiplicity=1) .

Relationship between card class and color class-

There can be cards of various colors(multiplicity 1*) ,but for each color there will only be one card corresponding to a value.

Relationship between hand and card class-

A player should have at least one card(hence multiplicity=1*) ,but a particular card can only belong to a single player (hence multiplicity =1).


Visual Paradigm Professional(hp(Sheridan College))
Deck
+cardInDeck : int
+deck()
+reset() : void
+pullDeck()
+throwDeck()
+drawCard() : Card
Card
+card(color, value)
+getColor() : color
+getValue() : Value
Hand
+playerName : String
+Hand()
+start(Game) : void
+Game(String) : void
+sortBySuit()
+sortByValue()
Value
+zero() : value
+one(n : String) : value
+two() : value
+otherValues(t : String) : value
Color
+red() : color
+blue() : color
+wild() : color
+otherColors() : color