1. INTRODUCTION

WHAT YOU WILL FIND HERE

This document consists of a series of web pages that attempts to give a comprehensive solution in object-oriented analysis and design to a Card Game System that has two popular card games, Switch and Basic Rummy.

Design of the Card Game System starts with the requirements statement, requirement analysis, overall design and detailed design.

Requirement analysis is done by use case diagrams and detailing the flow each use case in two different card games. We have listed down a set of functional test cases to confirm the initial test cases. A analysis class diagram also given in this section for identified analysis classes.

Overall design starts with CRC cards and then class diagram, state charts and interaction diagrams.

In detailed design, detailed about classes and package diagram can be found.

BACKGROUND

This documented is created for an assignment given by Dr. Neil Hurley, School of Computer Science and Informatics, University College Dublin for Object Oriented Design module.

USING THESE PAGES

The best way to start referring these pages is to first go through the requirements statement; then move to analysis, overall design and detailed design accordingly. We have used ArgoUML and Visual Paradigm Professional to draw the diagrams.

DOWNLOADABLE VERSION

You can find a downloadable version of the complete set of pages from here.

AUTHOR AND COPYRIGHT INFORMATION

Author information are as follows:

Team: *NoNameYet*

15209132	W.A.H.M. Dinuka T Jayaweera
15209372	Fernando W. Dilshan Nilendra
15209379	Ranatunga I.M.R.
15209147	Dushan Chamara Galappaththi
14208891(15209397)	Pasindu De Silva
15209136	Piumal Migara Mahawasala K.R.T.M.D.

2. REQUIREMENTS

REQUIREMENT STATEMENT FOR CARD GAME SYSTEM

The software to be designed is a software system for playing card games and this system consists of two popular card games; Switch and Basic Rummy.

Image 2.1, Source: http://www.rgbstock.com/bigphoto/2dASAN6/Card+Pack

A card deck consists of 52 individual cards and each card has a suit and a value as given in the following sets.
Suit = {Diamonds, Hearts, Spades, Clubs}
Value = {A, 2, 3, 4, 5, 6, 7, 8, 9, 10, J, K, Q}
A, J, K, Q has special names as Ace, Jack, King and Queen respectively.
There can be one or more players and each player is given a hand of cards assigned to the player where hand is the set of cards assigned to a player. Number of cards given for a hand is differ from one game to another. Players should perform actions such as lay one or more cards on the table, discard one or more cards, put one or more cards back in the deck in each turn.Each game has a set of rules for moving cards and winning strategy.

Below you can find the requirements descriptively explained on Switch and Basic Rummy card games.

REQUIREMENT STATEMENT FOR SWITCH CARD GAME

Switch card game, also known as Two Four Jacks, Black Jack and Irish Switch is a shedding type card game.
Switch card game is played with a regular, single deck of playing cards or if there are large number of players with two standard decks shuffled into one. Each player at his turn may play any card from his hand that matches the suit or the rank of the card previously played.

Game play is as following;

- Players are initially dealt a similar sized hand of cards, often seven per person, but the exact number may vary depending on how many players are present.
- The remainder of the deck is placed face down and serve as a "pool" or drawing stack.
- At the beginning of the game the topmost card from the "pool" is revealed and, so long as this card is not an Ace, play begins. Switch may not start with an Ace, and so if the "starting card" is an Ace, cards shall continue to be selected from the pool until a non-Ace card is revealed.
- The first to play, generally, the player on the dealer's left should select from his or her hand a card that matches either, the suit or the rank of the open card, the card that is "top"; for example, on a 9 of spades, only a spade card or a 9 may be played. If a player is not able to place a card and the player does not have an Ace, he draws a single card from the stack.
- If the drawing stack is run down and becomes empty, the playing stack or discard pile (except for the topmost card) is shuffled, and placed face down to become the new "pool."
- Ace: can be played regardless of the suit or value of the topmost card on the playing deck—that is, the Ace may be played at any time in the game. When playing an Ace, the player can decide freely the suit that has to be played next; from then on, play continues as normal, but on the suit selected by the player of the Ace.

In Switch some cards are known as "power" or "trick" cards, because they are being played directly affects the gameplay:

- 2: if a player places a two (of any suit) down, the next player is required to pick up two cards. Should that player have a two himself, however, he may place it down, requiring the next player to pick up four; if he has a two, he may place it, requiring the next player to pick up six; this may continue until the flow reaches a player who does not have a two in his hand, at which point he is required to pick up the required number of cards. A player that draws cards after a two has been played is usually not permitted to put any more cards down.

- 8: the next player misses their turn. There is not usually the option for the next player to play an 8 if she has any, as there is with the 2; however, if this rule is included, then 8s will continue to be played, until the flow reaches a player without an 8, in which case he will miss a number of turns equivalent to the number of 8s played immediately previously.
- King: king reverses the play direction
- Black Jack: When the Black Jack is played, the following player must pick up the same amount of cards dealt or play another Black Jack and the following player must then pick up double that. If you have both Black Jacks then you can play both of them at the same time, to then cause the next person to pick up.
- Red Jack: Is best played when a Black Jack is played as this will cancel the pick up black jack rule. One red Jack cancels one black jack.
- Ace: can be played regardless of the suit or value of the topmost card on the playing deck—that is, the Ace may be played at any time in the game. When playing an Ace, the player can decide freely the suit that has to be played next; from then on, play continues as normal, but on the suit selected by the player of the Ace.

If a player has a 10, they can place any card of the same suit down, but from then it must carry on in order. For example, 10 of hearts is placed, then you can put down a 7 of hearts. However, after this you have to put down an eight if hearts, or a six of hearts, or a 7 of a different suit, or move on to the next player.

When a player has only one remaining card they must remember to call last card (by saying *last card* aloud) before their turn has ended, to inform the other players that they are about to win. Should a player who has graduated to last card fail to call before the end of the turn in which they reach last card (that is, once the next player has started her turn after the last-card player has put down his or her second last card), he may be penalised, often to the cost of picking up one card immediately (over and above any picking up as a matter of routine course in the game).

As soon as a player plays their last card they win the game. If the last card is a power card they must draw another card as a game can not end with an power card. The game can continue until all the players get rid of their cards.

REQUIREMENT STATEMENT FOR BASIC RUMMY CARD GAME

(Image 2.2, Source:https://en.wikipedia.org/wiki/Rummy#/media/File:A_Game_of_Rummy.JPG)

The game is best played with two to four players, but up to six can take part. One standard deck of 52 cards is used. Cards in each suit rank, from low to high: Ace 2 3 4 5 6 7 8 9 10 Jack Queen King.
The dealer is chosen randomly and the cards are dealt one at a time. In a two player game, each player is dealt a hand of ten cards. Seven cards each are dealt if there are three or four players, and when five or six play each player gets six cards.
After the deal, the next card is placed face up on the table to start the discard pile, and the remainder of the deck is placed face down beside it to form the stock.
The players look at and sort their cards.

The object of the game is to dispose of all the cards in your hand. There are three ways to get rid of cards: melding, laying off, and discarding.

- *Melding* is taking a combination of cards from your hand, and placing it face up in front of you on the table, where it stays. There are two kinds of combination which

can be melded: sequences (also known as runs) and groups (also known as sets or books).
    - a sequence or run consists of three or more cards of the same suit in consecutive order.
    - a group, set or book is three or four cards of the same rank
- *Laying off* is adding a card or cards from your hand to a meld already on the table. The cards added to a meld must make another valid meld.
- *Discarding* is playing a card from your hand on top of the discard pile. You get rid of one card this way at the end of each turn.

Game rules

If there are two players, they take alternate turns starting with the non dealer. If there are more than two players, they take turns in clockwise rotation, beginning with the player to dealer's left. Each turn consists of the following parts.

1. The Draw. You must begin by taking one card from either the top of the Stockpile or the top card on the discard pile, and adding it to your hand. The discard pile is face up, so you can see in advance what you are getting. The stock is face down, so if you choose to draw from the stock you do not see the card until after you have committed yourself to take it. If you draw from the stock, you add the card to your hand without showing it to the other players.
2. Melding. If you have a valid group or sequence in your hand, you may lay one such combination face up on the table in front of you. You cannot meld more than one combination in a turn. Melding is optional; you are not obliged to meld just because you can.
3. Laying off. This is also optional. If you wish, you may add cards to groups or sequences previously melded by yourself or others. There is no limit to the number of cards a player may lay off in one turn.
4. The Discard At the end of your turn, one card must be discarded from your hand and placed on top of the discard pile face up. If you began your turn by picking up the top card of the discard pile you are not allowed to end that turn by discarding the same card, leaving the pile unchanged - you must discard a different card. You may however pick up the discard on one turn and discard that same card at a later turn. If you draw a card from the stock, it can be discarded on the same turn if you wish.

If the stockpile has run out and the next player does not want to take the discard, the discard pile is turned over, without shuffling, to form a new stock, and play continues. A player wins by either melding, laying off, or discarding all of his or her cards. Getting rid of your last card in one of these ways is called going out. As soon as someone goes out, play ceases and that player wins the game.

REFERENCES:

I. Switch (card game) - Wikipedia, the free encyclopedia (2016), https://en.wikipedia.org/wiki/Switch_(card_game) [Accessed on 03 January 2017].

II. Rummy - Wikipedia, the free encyclopedia (2016), https://en.wikipedia.org/wiki/Rummy [Accessed on 03 January 2017].

3. ANALYSIS

3.1.USE CASES FOR CARD GAME SYSTEM

General use cases of the card game system are identified as follows:

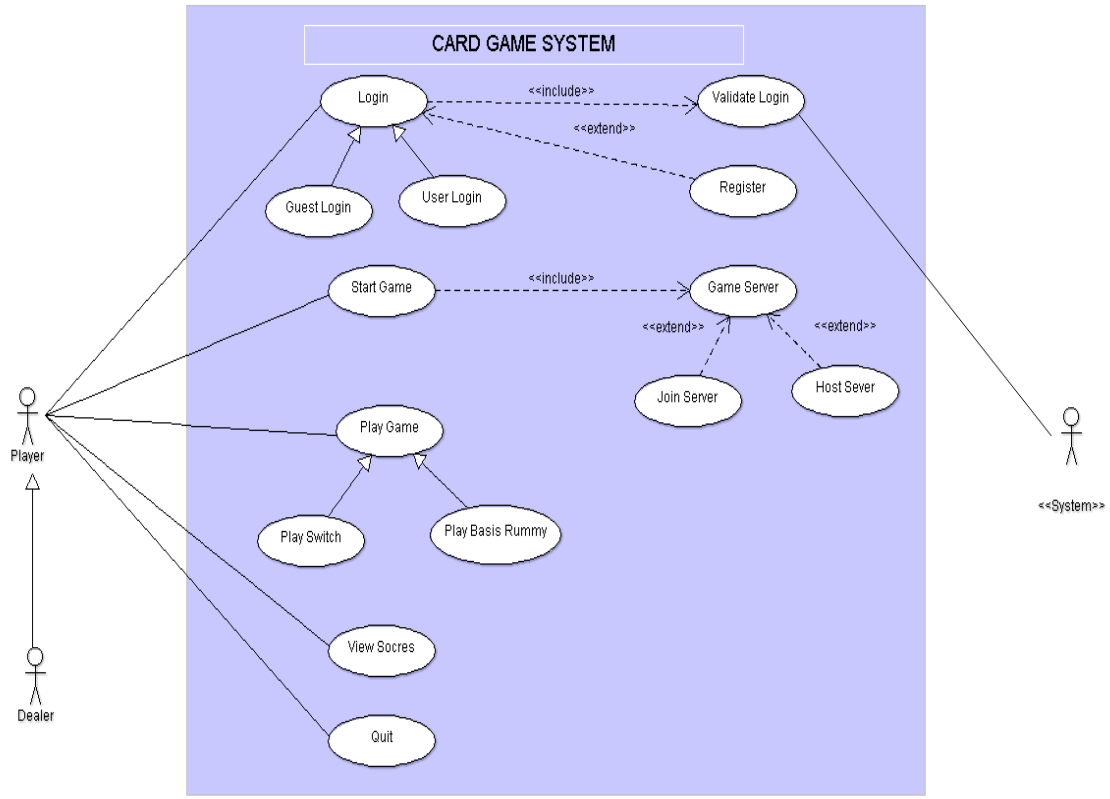

Image 3.1. 1.

View full size image from here.

Login use case

As a player, I should be able to log in to the game system, so that I can play the game.
There should be a GUI with the options,

- User Login
- Guest Login
- Register

for user to select an option to login to the game system.

User Login use case

As a registered user, I should be able to login to the game system, so that I can play the game.
There should be a GUI with the fields,

- Username
- Password

For user to login to the game system. After a successful login, a player should be directed to the Main menu page.
Main menu page will have two options,

1. Start Game
2. View Scores
3. Quit

Guest Login use case

As a player, I should be able to login to the game system without been registered, so that I can play the game.
There should be a GUI with the field to enter the 'Username' for user to enter the game system and after a successful login, a player should be directed to the Main menu page.

Validate Login use case

As a system operator, I want login details to be validated, so that I can keep user information accurately.

- Password should be equal to the password given for the registered username and username should be a valid one.
- Username given at a guest checkout must be unique and shouldn't be conflict with already existing registered username. There can be guest users with same username, but we do not capture their scores or information in our system.

Register use case

As a player, I should be able to register myself as a player, so that I can play the game while maintaining my records.
There should be a GUI to enter,

- Username
- Password
- Confirm password
- Email

for a user to register himself. An confirmation email should be sent to the user's provided email address upon a registration. When the registration is success a pop-up message should show up and should be redirected to login page.
Acceptance criteria:

- Username should be unique.
- Email should be a valid email address.

Start Game use case

As a player, I should be able to start the game by connecting with server, so that I can play the game.
There should be a GUI to enter,

- Server name (Dropdown selection)
- Username

and options to select,

- Join server

- Host server
- Start game

for a player to start the game.

Host Server use case

As a player, I should be able to host a game, so that I can host a new game by myself to play.
(Refer Start Game use case)

Join Server use case

As a player, I should be able to join a game, so that I can join an already hosted game to play.
(Refer Start Game use case)

Play Game use case

As a player, I should be able to play a card game of my preference, so that I can play cards on my computer.
There should be a GUI to select options,

- Play Switch
- Play Basic Rummy

for a player to select a game of his choice to play. By clicking on the option should be directed to the game screen.

Play Switch use case

As a player, I should be able to play Switch card game, so that I can play Switch card game on my computer.
(Refer Play Game use case)

Play Basic Rummy use case

As a player, I should be able to play Basic Rummy card game, so that I can play Basic Rummy on my computer.
(Refer Play Game use case)

View Scores use case

As a player, I should be able to view my past scores, so that I can play well to increase my scores.

Quit Game use case

As a player, I should be able to exit the game by disconnecting the connection with the server, so that I can exit the game.
Exit button should show up,

- At the end of a game
- In the main menu

USE CASE DIAGRAM FOR SWITCH CARD GAME

Use case diagram for Switch card game is as follows:

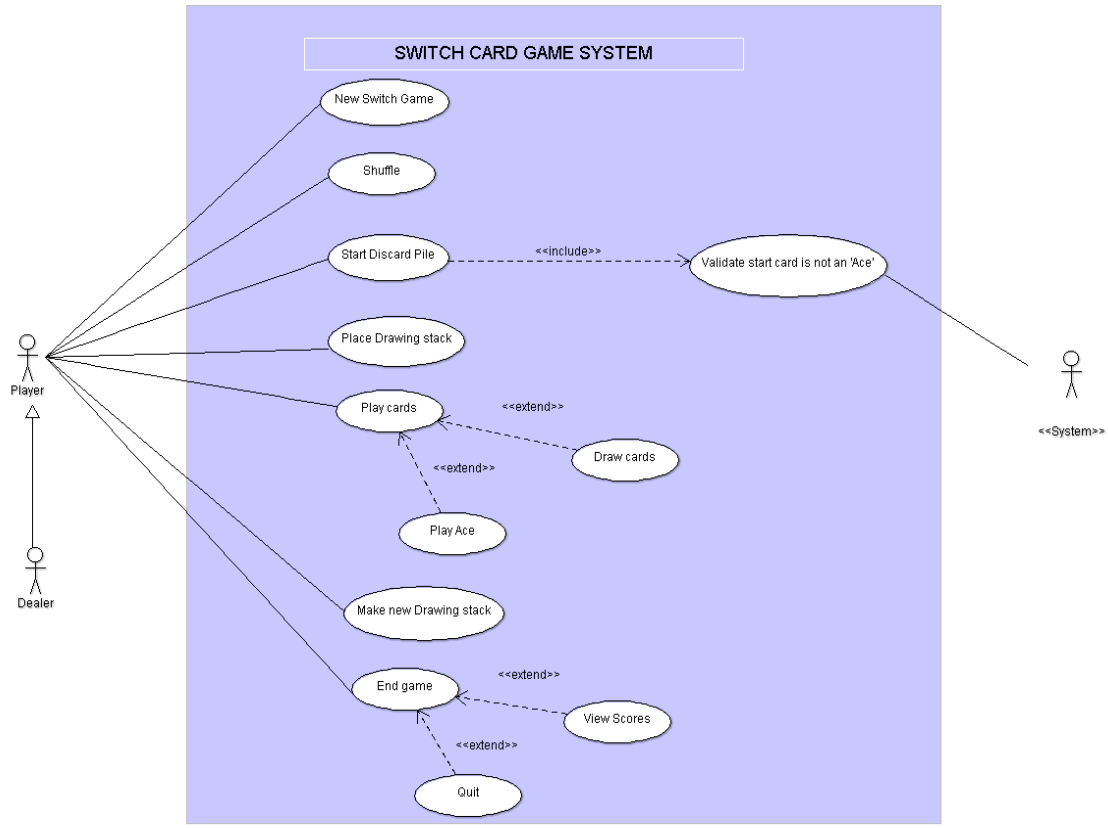

Image 3.1.2.

View full size image from here.

New Switch Game use case

As a player, I should be able to play a new Switch game, so that I can play a new Switch card game.

Shuffle cards use case

As a dealer-player, I should be able to shuffle the cards so that I can make sure that cards are mixed.

Deal cards use case

As a dealer-player, I should be able to deal similar sized hand of cards to players, so that I can ensure that everyone has got similar number of cards.

- Players should initially deal a similar sized hand of cards and by standard it should be 7 per person, but it can vary upon the number of players.

Place Drawing Stack use case

As the game system, I should be able to place the remainder of the deck face down to serve as drawing stack, so that I can place the remainder of the deck to serve as a ‘pool’ or drawing stack.

Start the discard pile use case

As a player, I should be able to reveal the top most card from the drawing stack, so that I can start the discard pile and begin the play.
Acceptance Criteria:

- Starting card should not be an Ace, if the "starting card" is an Ace, cards shall continue to be selected from the pool until a non-Ace card is revealed.

Validate Start Card is Not an ‘Ace’ use case

As the game system, I should be able to validate, “starting card” is not an Ace, so that I can maintain the game rules.

Play Cards use case

As the first player - player on the dealer's left, I should be able to select a card from my hand that matches either, the suit or the rank of the open card, the card that is "top" or if I am not able to place a card and I do not have an Ace, then I should be able to draw a single card from the deck, so that I can play my turn.

Draw Cards use case

As a player, if I do not have a card in my deck that matches either, the suit or the rank of the open card, then I should be able to draw a single card from the deck, so that I can play my turn.

Play Ace use case

As a player, I should be able to play Ace regardless of the suit or value of the topmost card on the playing deck, so that I can play Ace anytime in the game.
Acceptance criteria:

- Player can decide freely the suit that has to be played next and from then on, play continues as normal, but on the suit selected by the player of the Ace.

Make New Drawing Stack use case

As a player, if the drawing stack is run down and becomes empty, the playing stack or discard pile except for the topmost card is shuffled, then I should be able to place down to form a new ‘pool’, so that I can form a new drawing stack.

End Game use case

As a player, when all the players get rid of their cards I should be able to end the game, so that I can end the game.

Winning the game scenario:

As a player, I should be able to win the game as soon as I play my last card, so that I can win the game.

Acceptance Criteria:

- If the last card is a power card they must draw another card as a game can not end with an power card.

Power cards/ Trick cards: 2, 8, King, Black Jack, Red Jack, Ace (Refer Requirements Statement for Switch Card Game)

View Score use case

As a player, once the game ended I should be able to view the scores of players, so that I can improve playing switch card game.
Once the game is ended two options should pop-up,

- Quit
- View scores

USE CASE DIAGRAM FOR BASIC RUMMY CARD GAME

Use case diagram for Basic Rummy card game is as follows:

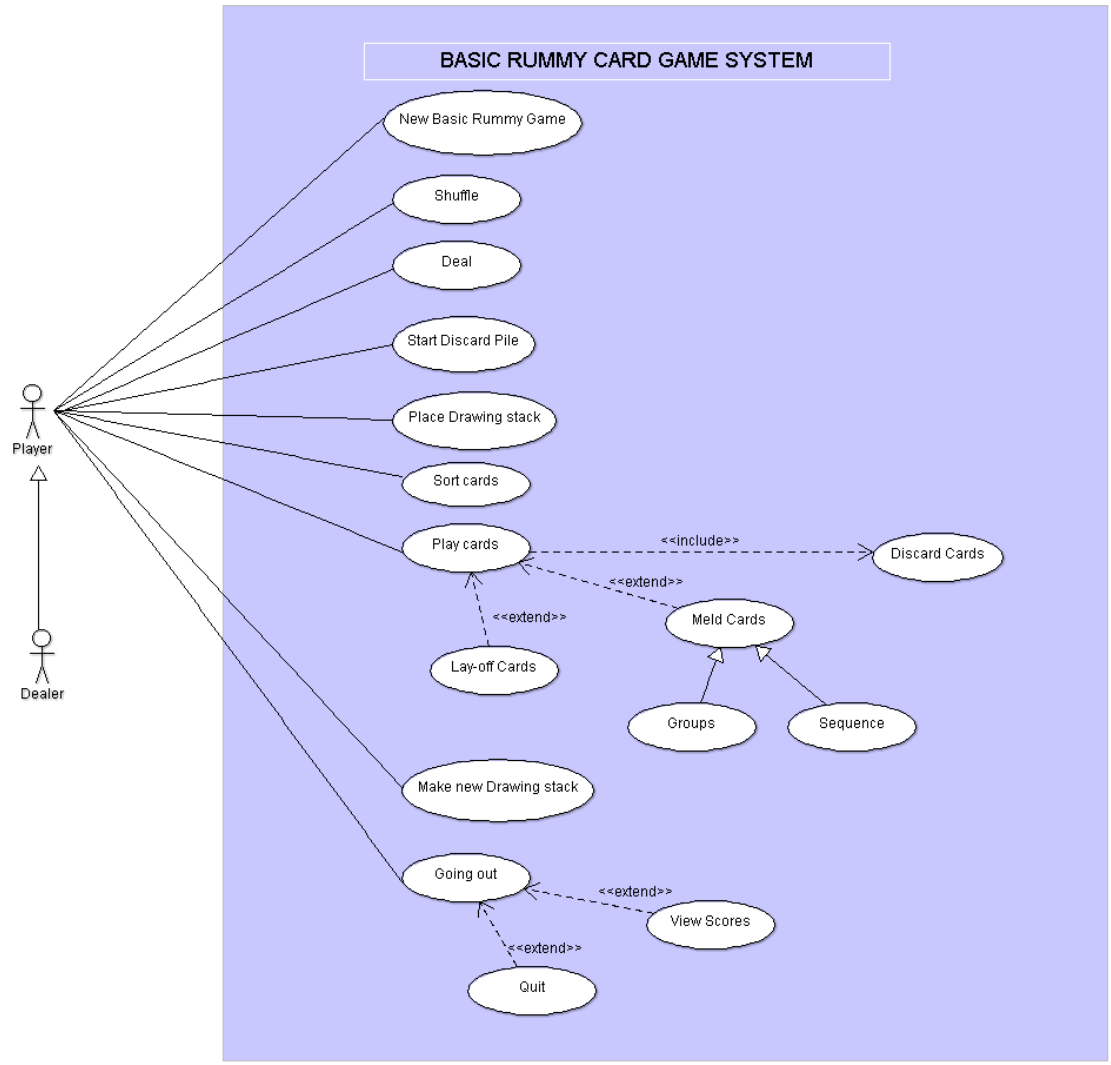

Image 3.1.3.

View full size image from here.

New Basic Rummy Game use case

As a player, I should be able to play a new Basic Rummy game, so that I can play a new Basic Rummy card game.

Shuffle Cards use case

As a dealer-player, I should be able to shuffle the cards so that I can make sure that cards are mixed.

Deal Cards use case

As a dealer-player, I should be able to deal similar sized hand of cards to players, so that I can ensure that everyone has got similar number of cards.
Notes:

- In a two player game, each player is dealt a hand of ten cards. Seven cards each are dealt if there are three or four players, and when five or six play each player gets six cards.
- The cards are dealt one at a time.

Start the Discard Pile use case

As a player, after the deal I should be able to place the next card face up on the table to start the discard pile, so that I can start the discard pile.

Place Drawing Stack use case

As a player, I should be able to place the remainder of the deck face down, so that I can place the remainder of the deck to serve as a 'pool' or drawing stack.

Sort the Cards use case

As a player, I should be able to sort the cards, so that I can arrange my cards in a order.

Play Cards use case

As a player, I should be able to play the game to dispose of all the cards in my hand, so that I can play Basic Rummy.
Notes:
There are three ways to play the cards as follows:

I. Melding
II. Laying off
III. Discarding

Meld Cards use case

As a player, I should be able to meld the cards, so that I can get rid of cards.
Notes:
Melding is taking a combination of cards from hand, and placing it face up on the table, where it stays. There are two kinds of combination which can be melded:

I. Sequences (also known as runs)
II. Groups (also known as sets or books)

Meld Cards - Groups use case

As a player, I should be able to meld the cards- Group wise, so that I can get rid of cards using melding.
Notes:
A group, set or book is three or four cards of the same rank.

Meld Cards - Sequences use case

As a player, I should be able to meld the cards- Sequences wise, so that I can get rid of

cards using melding.
Notes:
A sequence or run consists of three or more cards of the same suit in consecutive order.

Lay off Cards use case

As a player, I should be able to lay-off the cards, so that I can get rid of cards.
Notes:
Laying off is adding a card or cards from hand to a meld already on the table.
Acceptance Criteria:
The cards added to a meld must make another valid meld.

Discard Cards use case

As a player, I should be able to discard cards, so that I can get rid of one card at the end of each turn.
Notes:
Discarding is playing a card from your hand on top of the discard pile.

Make a New Pool use case

As a player, if the stockpile has run out and the next player does not want to take the discard, I should be able to turn over the discard pile without shuffling to form a new stock, so that I can continue the play.

Going-out use case

As a player, if I get rid of the last card, I should be able to go out (win the game), so that I can be the winner.

View Scores use case

As a player, once the game ended I should be able to view the scores of players, so that I can improve playing Basic Rummy card game.
Once the game is ended two options should pop-up,

- Quit
- View scores

3.2.INITIAL FUNCTIONAL TEST CASES FOR CARD GAME SYSTEM

GENERAL TEST CASES

Use Case	Function Being Tested	Initial System State	Input	Expected Output
Login	A user should be able to select an option to login to the game system. Options are: User Login, Guest Login and Register.	System off	Click on buttons:User Login, Guest Login and Register.	Should go to User Login, Guest Login and Register screens on the relevant button clicks.
User Login	A registered user should be able to login to the game system.	User haven't logged in	Registered username and password	After a successful login, a player should be directed to the Main menu page.
Guest Login	A non-registered user should be able to log in to the system.	User haven't logged in	Username	After a successful login, a player should be directed to the Main menu page.
Validate Login - Guest Login	Username given must be unique and shouldn't be conflict with already existing registered username.	User haven't logged in	Username	After a successful login, a player should be directed to the Main menu page.
Validate Login - User Login	Password should be equal to the password given for the registered	User haven't logged in	Registered username and password	After a successful login, a player should be directed to the

	username and username should be a valid one.			Main menu page.
Register	A user should be able to register himself/herself.	User haven't registered	Username, Password, Confirm password, Email	A success pop-up message should show up and should be redirected to login page.
	A user should receive a confirmation email for a successful registration.	User haven't registered	Username, Password, Confirm password, Email	A confirmation email to users provided email address.
	Provided username should be unique.	User haven't registered	Username	A tick in-front of the field after validating
	Email should be a valid email address.	User haven't registered	Email	A tick in-front of the field after validating
Start Game	A user should be able to start the game by connecting with server.	User logged in	Server name , Username	Should go to select game screen.
Host Server	A user should be able to host a game.	User logged in	Server name , Username	Should go to select game screen.
Join Server	A should be able to join a game.	User logged in	Server name , Username	Should go to select game screen.
Play Game	A user should be able to select an option to play to a card game. Options are: Play Switch,	User joined a server	Click on buttons: Play Switch, Play Basic Rummy	Should go to Play Switch or Play Basic Rummy screens on the relevant button clicks.

	Play Basic Rummy			
Play Switch	A user should be able to play Switch card game.	User on the select game screen	Click on button: Play Switch	Should be directed to the Switch game screen.
Play Basic Rummy	A user should be able to play Basic Rummy card game.	User on the select game screen	Click on button: Play Basic Rummy	Should be directed to the Basic Rummy game screen.
Exit Game	Should be able to exit the game.	User logged in	Click on Exit button: At the end of a game, In the main menu	Should be disconnected from the server and system should go down.

Table 3.2.1.

TEST CASES FOR SWITCH CARD GAME PLAY

Use Case	Function Being Tested	Initial System State	Input	Expected Output
New Switch Game use case	A user should be able to play a new Switch game.	User on the Switch game screen	Click on button: New Switch Game	Go to Switch Game play screen
Deal cards use case	A dealer - player should be able to deal similar sized hand of cards to other users.	User on the Switch Game play screen	Number of players Click on button: Deal	Everyone should get similar number of cards
Place Drawing Stack use case	Game system should be able to create a drawing stack just after the dealt.	Dealt cards	Click on button: Create Drawing Stack	Creation of drawing stack.
Start the discard pile use case	Should be able to reveal the top	Created drawing stack	Click on button: Discard Pile	Revealed the top most card

	most card from the drawing stack			from drawing stack.
Validate Start Card is Not an ‘Ace’ use case	Game system should be able to validate, “starting card” is not an Ace.	Revealed the top most card from drawing stack.	Click on button: Discard Pile	If the "starting card" is an Ace, cards shall continue to be selected from the pool until a non-Ace card is revealed.
Play Cards use case	A user should be able to play cards.	Started discard pile	Select a card from the hand by Click on buttons: Draw card, Play Ace	Can play the turn
Draw Cards use case	A user should be able to draw a single card from the deck	Started discard pile	Click on button: Draw card	Can play the turn
Play Ace use case	A user should be able to play Ace regardless of the suit or value of the topmost card on the playing deck.	Started discard pile	Click on button: Play Ace	Can play Ace
	After playing ‘Ace’, play should continue as normal, but on the suit selected by the player of the Ace.	Played Ace	Click on button: Draw card	Can play the turn
Make a New Pool use case	A user should be able to make	Game playing	Click on button: Create Drawing Stack	Form a new drawing stack.

	a new drawing stack.			
End Game use case	A user should be able to end the game	Game playing	Click on button: End game	Should redirect back to the main menu page.
	If the last card is a power card they must draw another card.	Game playing	Click on button: End game	Should show a Error message -Game can not end with an power card. I.e. Power cards/ Trick cards: 2, 8, King, Black Jack, Red Jack, Ace
View Score use case	Once the game ended should be able to view the scores of players.	Game end	Click on button: View Scores	Scores of users should show in a pop-up screen.

Table 3.2.2.

TEST CASES FOR BASIC RUMMY CARD GAME PLAY

Use Case	Function Being Tested	Initial System State	Input	Expected Output
New Basic Rummy Game use case	A user should be able to play a new Basic Rummy Game .	User on the Basic Rummy game screen	Click on button: New Basic Rummy Game	Go to Basic Rummy Game play screen.
Deal Cards use case	A dealer - player should be able to deal similar sized hand of cards to other users.	User on the Basic Rummy Game play screen	Number of players Click on button: Deal	Everyone should get similar number of cards.
	Cards should be dealt on at a time.	User on the Basic Rummy Game play screen	Number of players Click on button: Deal	Everyone should get one card at a deal turn.

Start the Discard Pile	A user should be able to place the next card face up on the table to start the discard pile	Dealt cards	Click on button: Discard Pile	Place the next card of deck face up.
Place Drawing Stack	Game system should be able to create a drawing stack after starting the discard pile.	Started discard pile	Click on button: Create Drawing Stack	Creation of drawing stack.
Sort the Cards	A user should be able to sort the cards.	Placed drawing stack	Click on button: Sort cards	Cards get sorted according to ranks and suits.
Play Cards	A user should be able to play the game.	Cards sorted	Click on buttons: Meld cards, Lay-off cards, Discard cards	Meld cards, Lay-off cards and Discard cards buttons should be clickable.
Meld Cards	A user should be able to meld the cards.	Cards sorted	Click on buttons: Group Melding, Sequence Melding	A combination of cards should place face up.
Meld Cards - Groups	A user should be able to meld the cards- Group wise.	Cards sorted	Click on button: Group Melding.	Three or four cards of the same rank should face up.
Meld Cards - Sequences	A user should be able to meld the cards- Sequence wise.	Cards sorted	Click on button: Sequence Melding.	Three or more cards of the same suit in consecutive order should face up.
Lay off Cards	A user should be able to lay-off the	Cards sorted	Click on button: Lay-off cards	A card or cards from hand

	cards.			should be faced up to a meld already face up.
	Cards added to a meld by layoff must make another valid meld.	Cards sorted	Click on button: Lay-off cards	A card or cards from hand should be faced up to a meld already face up.
Discard Cards	A user should be able to lay-off the cards.	Cards sorted	Click on button: Discard cards	A card on top of the discard pile should face up.
Make a New Pool	A user should be able to turn over the discard pile without shuffling to form a new stock, so that I can continue the play.	Game playing	Click on button: Create a drawing stack	Form a new drawing stack.
Going-out	A user should be able to go out once get rid of the last card.	Game playing	Click on button: End game	Should redirect back to the main menu page.
View Scores	Once the game ended should be able to view the scores of players.	Game end	Click on button: View Scores	Scores of users should show in a pop-up screen.

Table 3.2.3.

3.3. ANALYSIS CLASSES

According to the initial go through of use cases, parts of the system are as follows:

- A controller object representing the Card Play Game itself, who manages the boundary objects.

- Boundary objects representing the individual component parts of the Card Play Game:

    - Login panel
    - Registration
    - User login
    - Guest login
    - Main menu panel
    - Game server
    - Game menu panel
    - Switch game panel
    - Basic Rummy game panel

- Controller objects corresponding to use cases:
    - Switch game play
    - Rummy game play

- Entity objects:
    - Log of player
    - Log of scores

This leads to the following diagram of analysis classes:

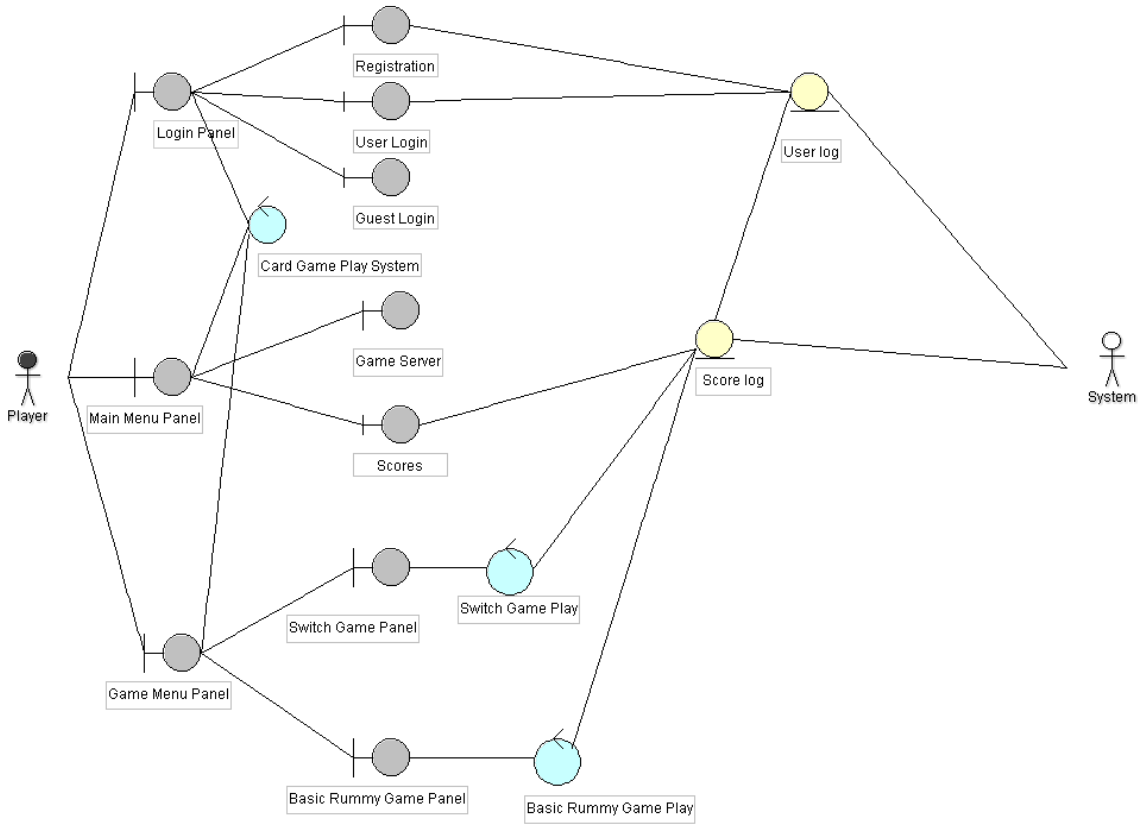

Image 3.3.1.

View full size image from here.

4.DESIGN

4.1.CRC CARDS FOR CARD GAME SYSTEM

CRC card are used to assign responsibilities to various classes for various use cases.

- Class CardGameSystem

Boundary objects - Component parts of the CardGameSystem:

- LoginPanel
- Registration
- UserLogin
- GuestLogin
- MainMenuPanel
- GameServer
- GameMenuPanel
- SwitchGamePanel
- BasicRummyGamePanel

Controller objects corresponding to various use cases:

- SwitchGamePlay
- RummyGamePlay
- DrawCardsSwitch
- DrawCardsRummy

Entity objects that is necessary to assign responsibilities to other objects:

- User
- Score
- DiscardPile
- DrawingStack
- Layoff
- Meld
- Hand
- Deck

Class CardGameSystem	
Responsibilities	Collaborators
• Provide access to component parts of the Game system.	

Class LoginPanel	
Responsibilities	Collaborators

• Display login options accept a choice from keyboard.	• UserLogin • GuestLogin

Class Registration	
Responsibilities	Collaborators
• Allow to enter registration details through key board. • Verify user details. • Send a email to user upon the registration.	• User

Class UserLogin	
Responsibilities	Collaborators
• Allow to enter user credentials through key board. • Verify the user credentials.	• User

Class GuestLogin	
Responsibilities	Collaborators
• Allow to enter user name through key board.	

Class MainMenuPanel	
Responsibilities	Collaborators
• Display main menu options and accept a choice from the keyboard.	• GameServer • GameMenuPanel

Class GameServer	
Responsibilities	Collaborators
• Display server types and accept a choice from keyboard. • Allowing to enter username and accept the response from keyboard. • Display the server connection options. • Initiate connection to players. • Terminate connections.	

Class GameMenuPanel	
Responsibilities	Collaborators
● Select a game type	● SwitchGamePanel ● BasicRummyGamePanel

Class SwitchGamePanel	
Responsibilities	Collaborators
● Allow to play Switch	

Class BasicRummyGamePanel	
Responsibilities	Collaborators
● Allow to play Basic Rummy	

Class SwitchGamePlay	
Responsibilities	Collaborators
● Act according to the game rules.	

Class RummyGamePlay	
Responsibilities	Collaborators
● Act according to the game rules.	

Class User	
Responsibilities	Collaborators
● Store User details	

Class Score	
Responsibilities	Collaborators
● Store the scores of users.	

Class Deal	
Responsibilities	Collaborators
● Allow to enter number of players through keyboard.	Deck

- Deal cards equally among players.

Class Deck	
Responsibilities	Collaborators
• Representing the suits and values of 52 cards in the standard card deck.	

Class Hand	
Responsibilities	Collaborators
• Sort the cards according to suit and value. • Allow to draw cards.	• Deck • DrawCardsRummy

Class DrawCardsRummy	
Responsibilities	Collaborators
• Allow to select an option to get rid of cards. • Allow to discard cards.	• Hand

Class Meld	
Responsibilities	Collaborators
• Allow to meld cards group wise or sequence wise.	• Hand

Class LayOff	
Responsibilities	Collaborators
• Allow to lay-off cards.	• Meld • Hand

Class DrawingStack	
Responsibilities	Collaborators
• Allow to place remainder of the deck face down after starting the discard pile in RummyGamePlay.	• Deck • DiscardPile • RummyGamePlay • SwitchGamePlay

• Allow to place remainder of the deck face down after the deal in SwitchGamePlay.	• Deal

Class DiscardPile	
Responsibilities	Collaborators
• Allow to reveal the top most card from DrawingStack in RummyGamePlay. • Allow to select next card after the deal from deck in SwitchGamePlay.	• Deck • DrawingStack • RummyGamePlay • SwitchGamePlay • Deal

Class DrawCardsSwitch	
Responsibilities	Collaborators
• Should allow to draw any single card.	

4.2. CLASS DIAGRAM FOR CARD GAME SYSTEM

Class diagram for the full Card Play Game System including login and initial steps and both Switch and Rummy games is as follows:

> Note that, this class diagram will be a large scaled one with the attributes, operations and other components of each class; therefore, only the class name is given here and details of classes are given in Detailed Design section.

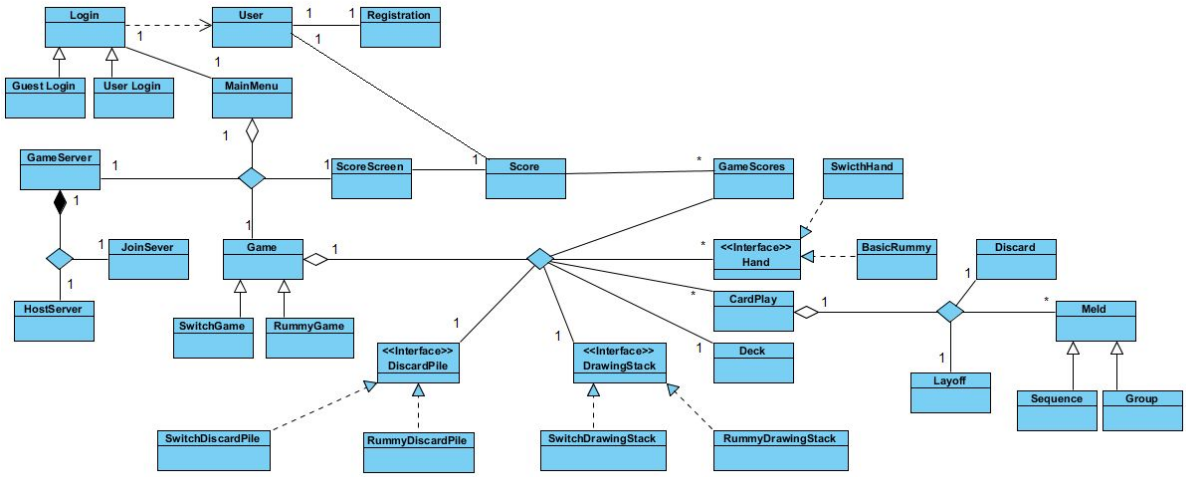

Image 4.2.1

View full size image from here.

4.3. STATE CHARTS FOR CARD GAME SYSTEM

Identified statechart diagrams for the Card Game System are as follows:

STATE CHART FOR CARD GAME SYSTEM UPTO GAME SELECTION

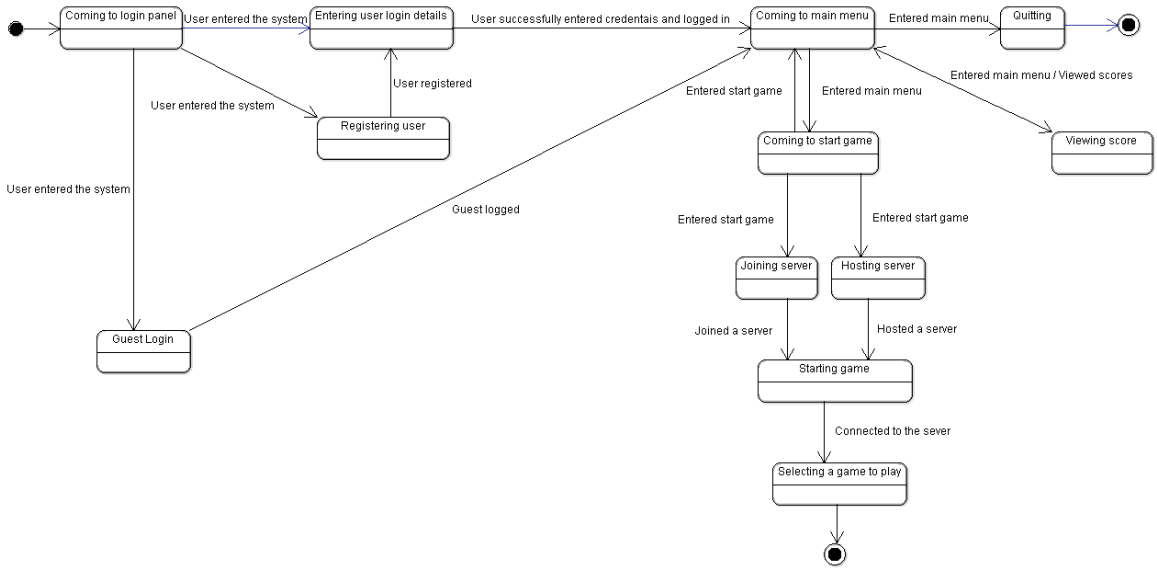

Image 4.3.1.

View full size image from here.

STATE CHART FOR BASIC RUMMY GAME PLAY

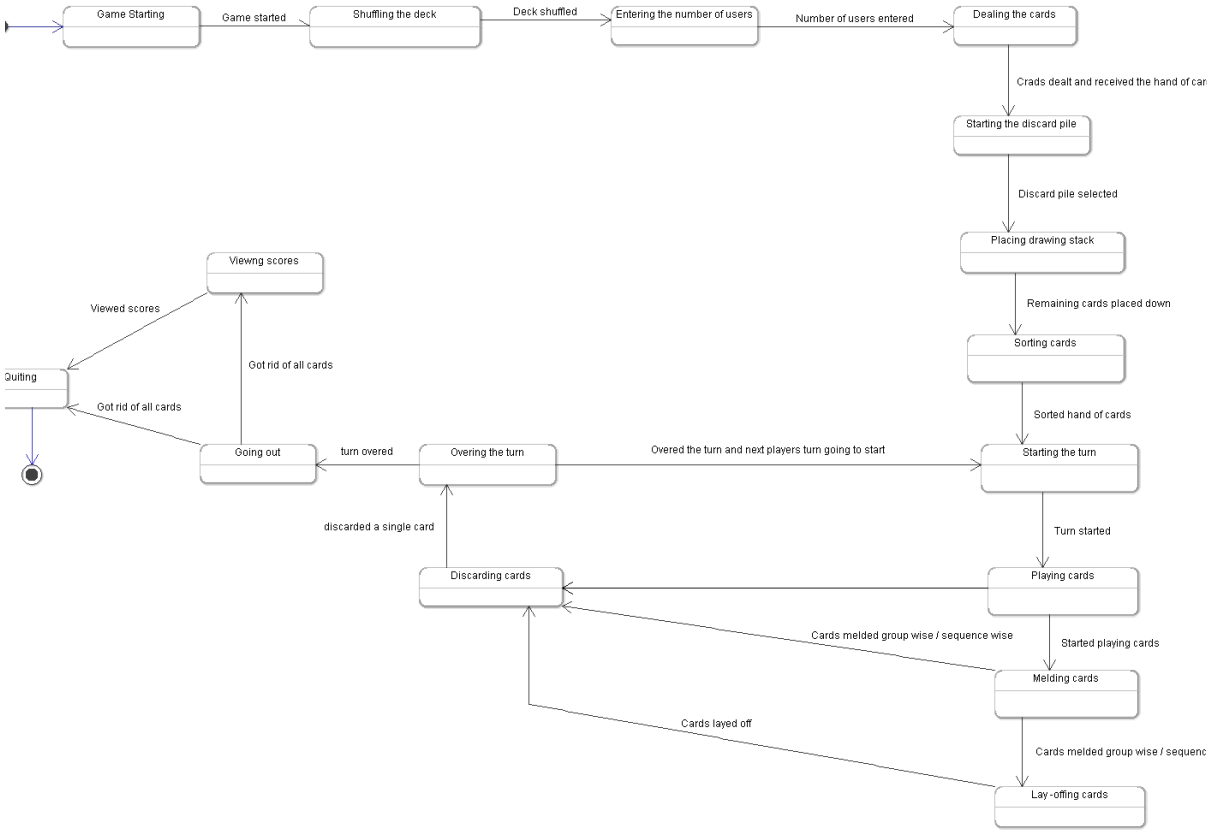

Image 4.3.2.

View full size image from here.

STATE CHART FOR BASIC RUMMY GAME PLAY [INCLUDE MAKE NEW DRAWING STACK USE CASE]

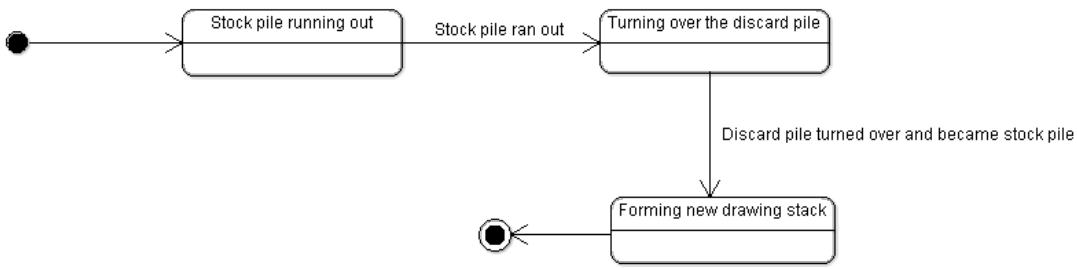

Image 4.3.3.

View full size image from here.

STATE CHART FOR SWITCH GAME PLAY

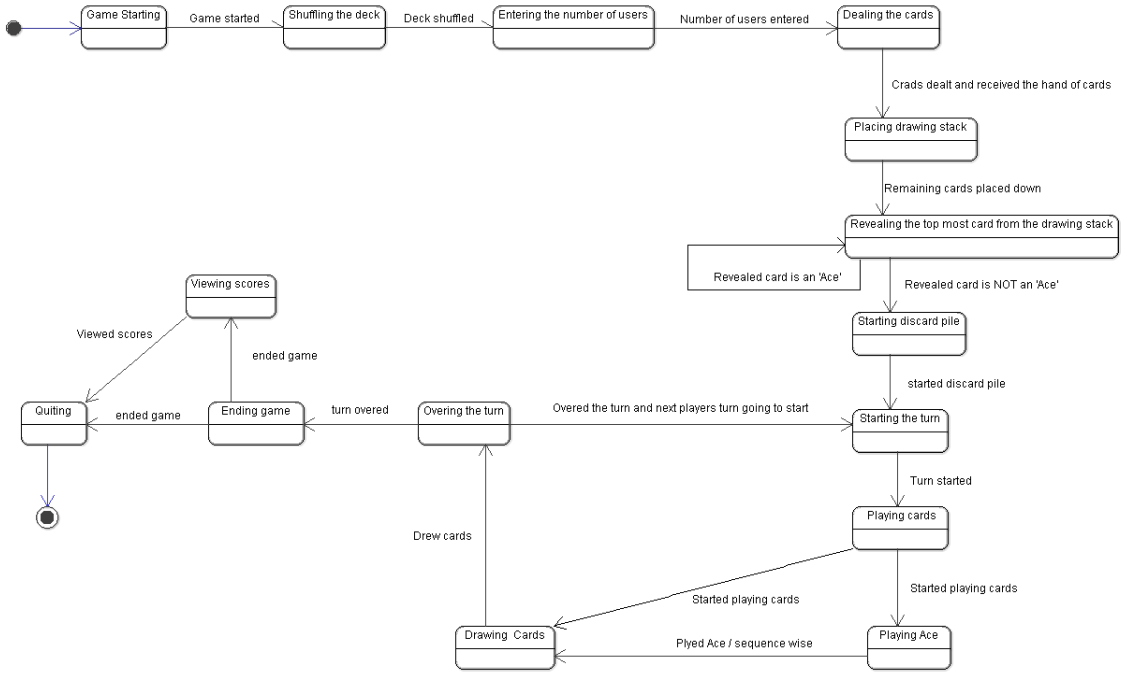

Image 4.3.4.

View full size image from here.

STATE CHART FOR SWITCH GAME PLAY [INCLUDE MAKE NEW DRAWING STACK USE CASE]

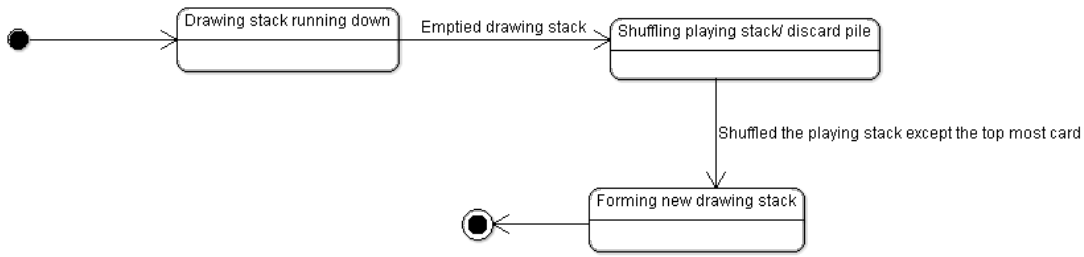

Image 4.3.5.

View full size image from here.

4.4. INTERACTION DIAGRAMS FOR CARD GAME SYSTEM

You can find interaction diagrams for major use cases of the interaction diagram as follows:

REGISTER WITH THE CARD GAME SYSTEM

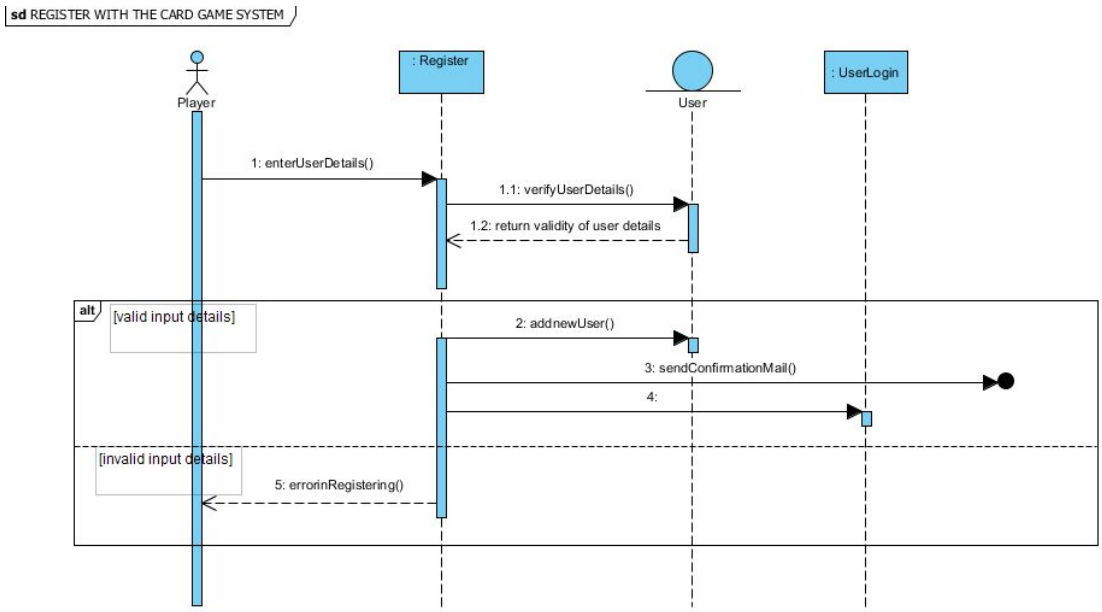

Image 4.4.1.

View full size image from here.

USER LOGIN TO THE CARD GAME SYSTEM

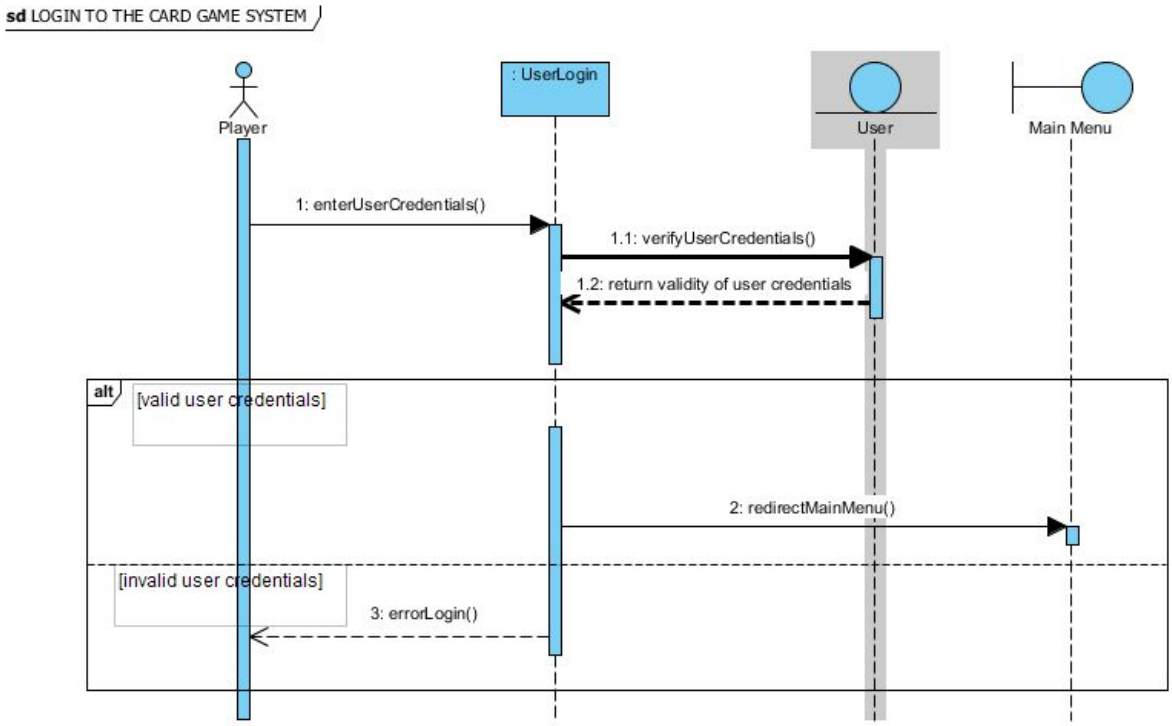

Image 4.4.2.

View full size image from here.

GUEST LOGIN TO THE CARD GAME SYSTEM

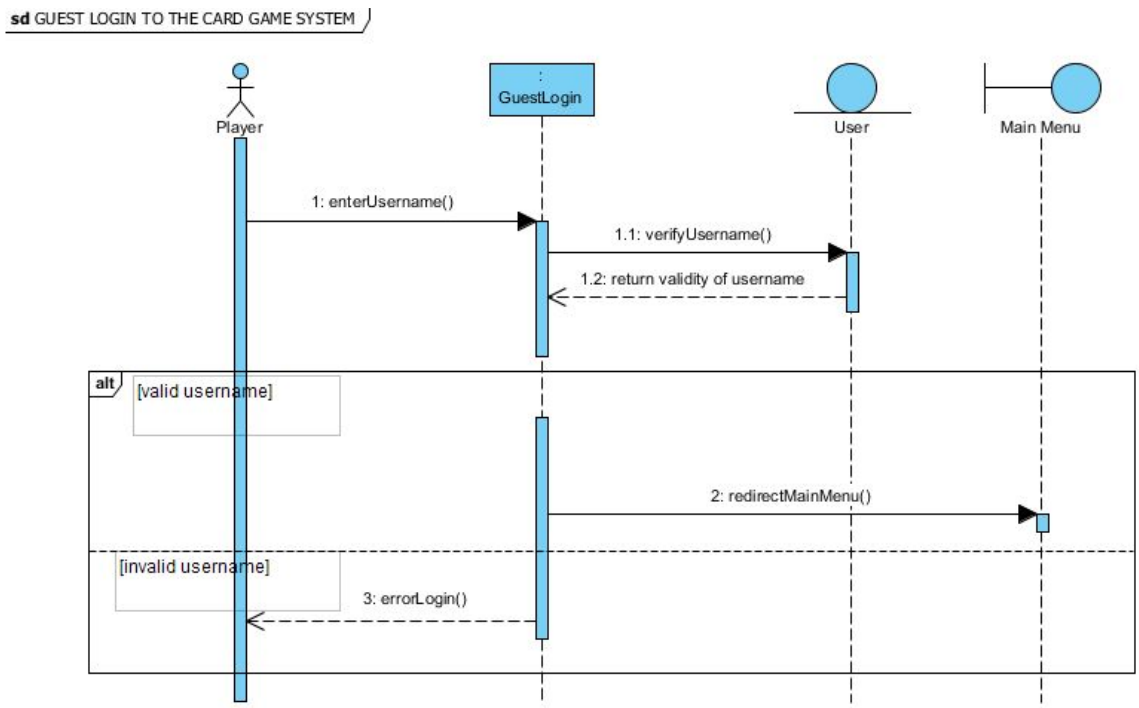

Image 4.4.3.

View full size image from here.

SWITCH GAME SHUFFLE, START DISCARD PILE AND PLACE DRAWING STACK

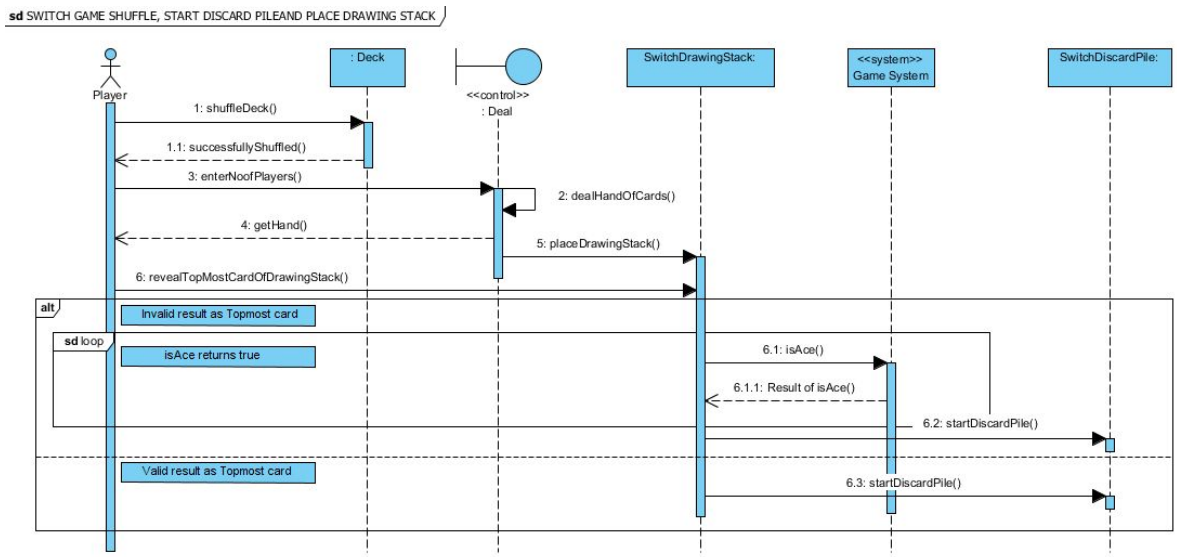

Image 4.4.4.

View full size image from here.

SWITCH GAME CREATING NEW DRAWING STACK

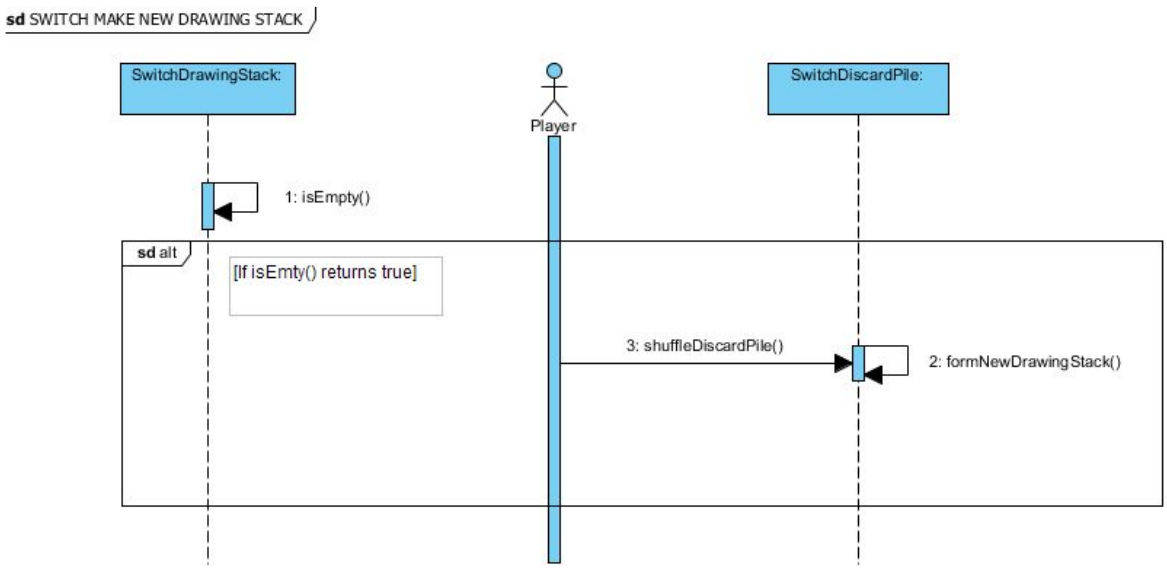

Image 4.4.6.

View full size image from here.

SWITCH GAME END

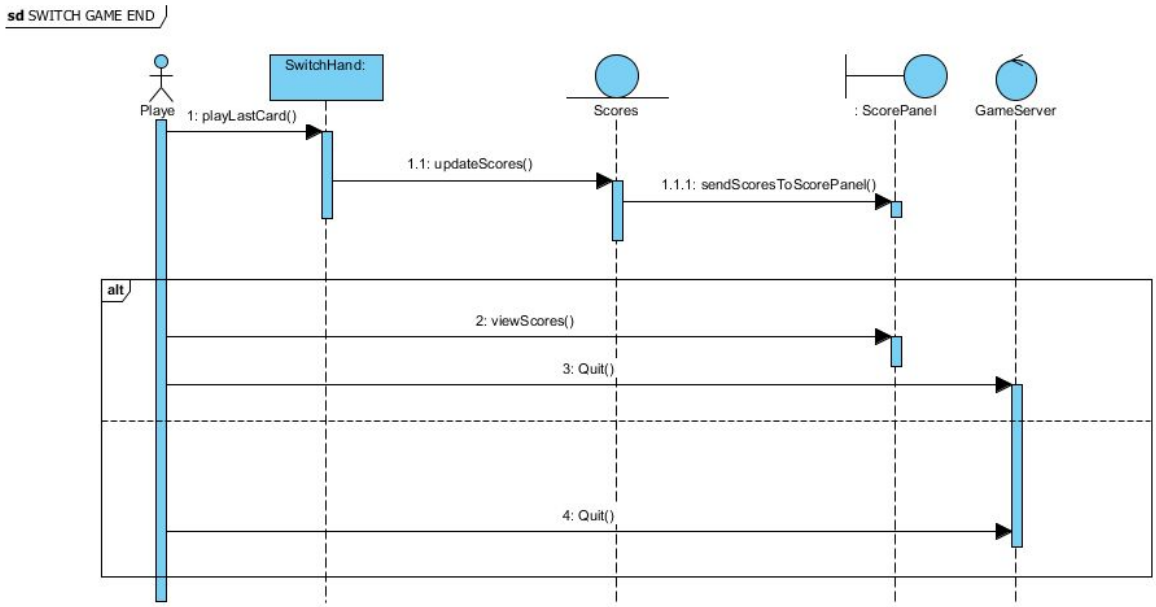

Image 4.4.7.

View full size image from here.

BASIC RUMMY GAME SHUFFLE, START DISCARD PILE AND PLACE DRAWING STACK

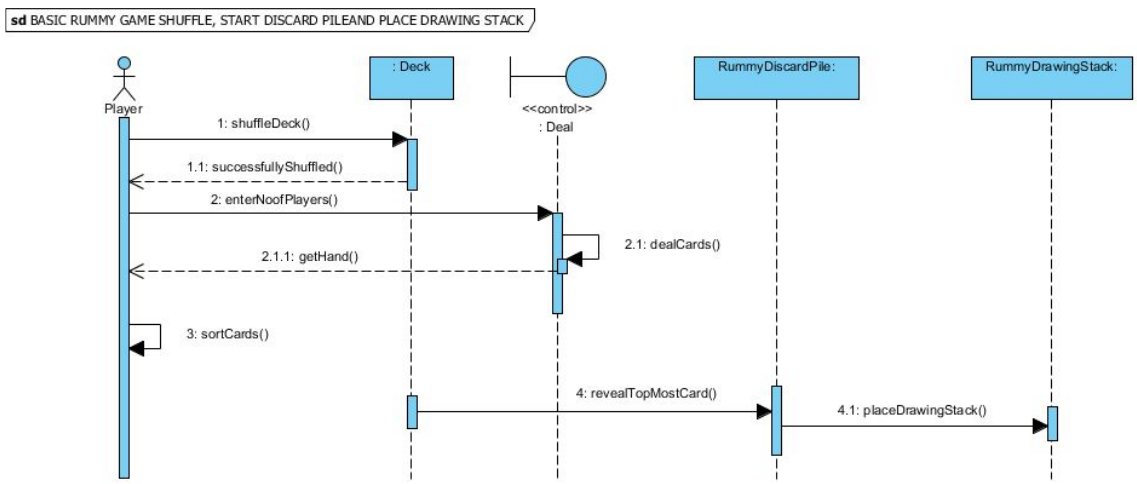

Image 4.4.8.

View full size image from here.

BASIC RUMMY GAME CREATING NEW DRAWING STACK

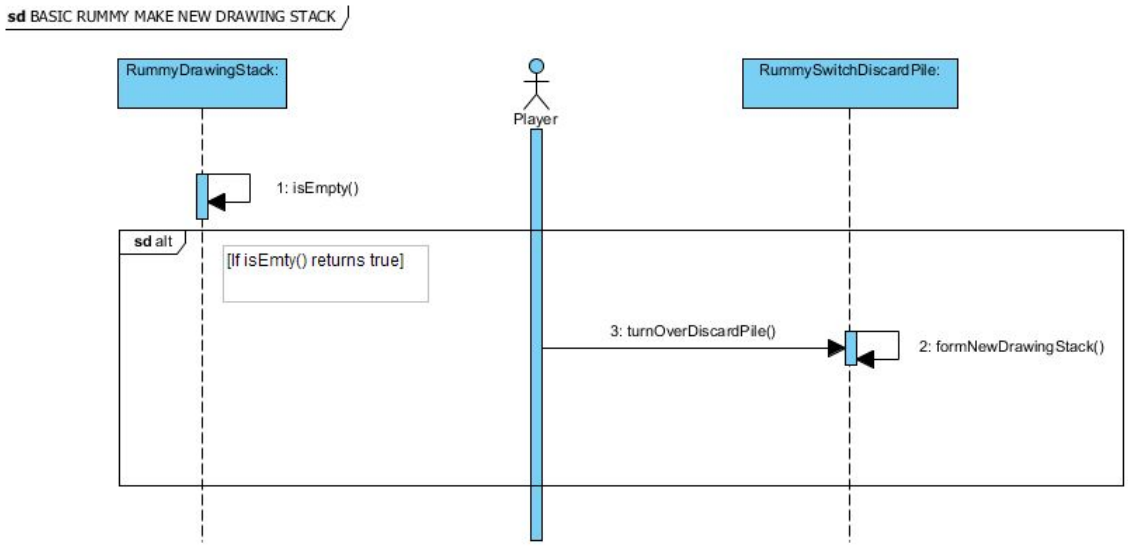

Image 4.4.9.

View full size image from here.

BASIC RUMMY GAME END

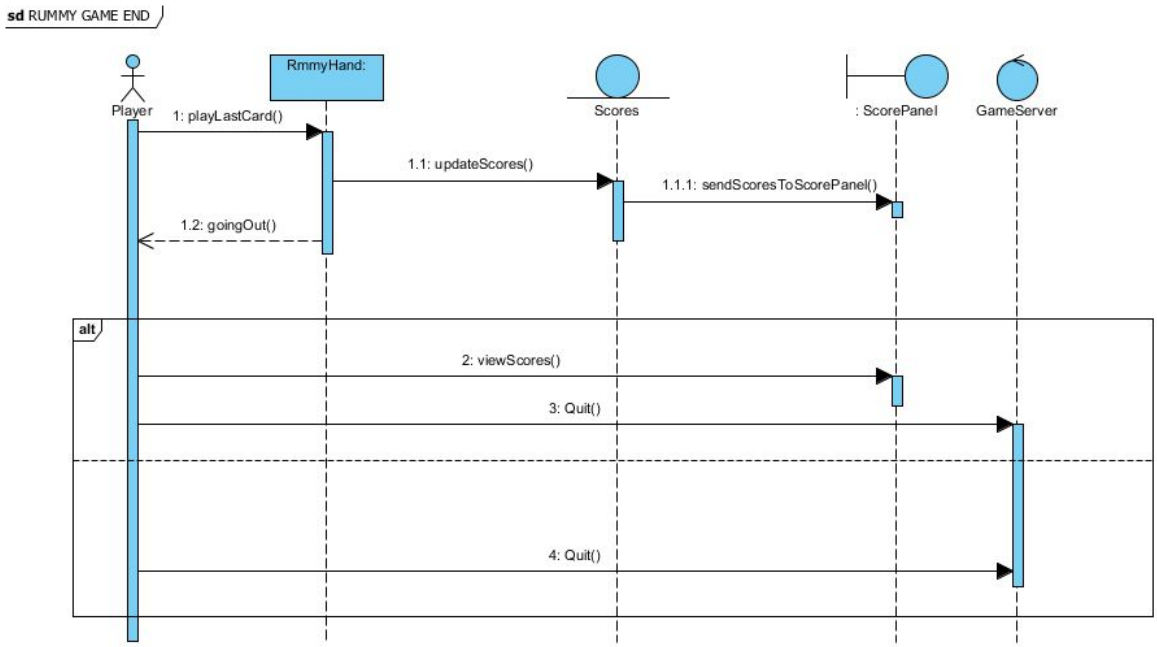

Image 4.4.10.

View full size image from here.

5.DETAILED DESIGN

5.1. DETAILED DESIGN FOR CARD GAME SYSTEM

Here you can find details on attributes and operations of classes we presented at class diagram in design phase.

Login
+ btnUserLoginClick() : void + btnGuestLoginClick() :void +btnRegisterClick() : void +checkValidity() : boolean

User Login
- txtUsername : string - txtPassword : string
+checkValidity() : boolean

Guest Login
- txtUsername : string
+checkValidity() : boolean

User
-Username: String -Password: String -Email : String
+ getUsername() : string + getPassword() : string + getEmail() : string + setUsername(): string

+ setPassword() :string + setEmail() : string

Register
- User : User -Username: String -Password: String -ConfirmPassword: String -Email : String
+ setUsername() : string +setPassword():string +setConfirmPassword(): String +setEmail(): String +checkPasswordValidity(): boolean +checkUsernameValidity(): boolean

MainPage
+ btnExit() : void + btnStartGame() : void + btnViewScores() : void +btnPlayGame(): void

Game
+ btnRummyClick() : void + btnSwitchClick() : void

SwitchGame

+ btnNewSwitchClick() : void

RummyGame
+ btnNewRummyClick() : void

GameServer
- ChannelName : string + setChannelName() : string
+ btnHostSeverClick() : void + btnJoinSeverClick() : void

HostServer
- ChannelName : string + setChannelName() : string
+ btnHostClick() : void

JoinServer
- ChannelName : string + setChannelName() : string
+ btnJoinClick() : void

ScoreScreen
- getScore() : int

+ setScore() : void

Score
- gameId : double - username : string - score : int - gameType : int
+getGameId() : double +getUsername() : string +getScore() : int +getGameType() : int +setGameId() : double +setUsename() : string +setScore() : int +setGameType() : int

Deck
- suit : string - rank : char
-getDeck() - shuffleDeck() +getRank() + getSuit()

DiscardPile
+lastCard: String
+startDiscardPile() :void +getLastCardPlayed() +placeFaceDown() :void +revealTopCard() : void

SwitchDiscardPile
+switchDiscardPile() :void +startDiscardPile() :void +revealTopCard() : void +validateNotAce() : bool

RummyDiscardPile
+rummyDiscardPile() +startDiscardPile()

DrawingStack
+revealTopCard() : void -isEmpty : bool +getRemainderOfDeck() +placeFaceDown() :void +shufflePlayingStack() :void +formNewPool() : void

SwitchDrawingStack
+getRemainderOfDeck() +placeFaceDown() : void +shufflePlayingStack() : void +formNewPool() : void

RummyDrawingStack

+formNewPool() : void +turnOverDiscardPile() : void +formNewPool() : void

Hand
-NoOfPlayers: int
+sortCards() : void +dealCards() : void +setNoOfPlayers() : int +getNoOfPlayers() : int

SwitchHand
-NoOfPlayers: int
+dealCards() : void +setNoOfPlayers() : int +getNoOfPlayers() : int

RummyHand
-NoOfPlayers: int
+sortCards() : void +dealCards() : void +setNoOfPlayers() : int +getNoOfPlayers() : int

CardPlay
+drawCards() : void +getSuitOfCard() +getRankOfCard() +playAce() : void +meldCards() : void +placeFaceUp() : void getNoOfCards() : int

LayOff
+layingoff() : void +checkForMelds() : boolean getNoOfCards() : int

Meld
+Meld: set<>
+meldCards() : void +placeFaceUp() : void getNoOfCards() : int

Sequence
+getExistingSequences() +meldCards() : void +placeFaceUp() : void +meldSequenceWise() +getSuitOfCard() +orderBySuit() getNoOfCards() : int

Group
+getExistingGroups() +meldCards() : void +placeFaceUp() : void +getRankOfCard() +meldGroupWise() getNoOfCards() : int

Discard
+discardCard() : void getNoOfCards() : int

5.2. PACKAGE DIAGRAM FOR CARD GAME SYSTEM

Classes of the Card Game System can be packaged into 3 main packages as follows:

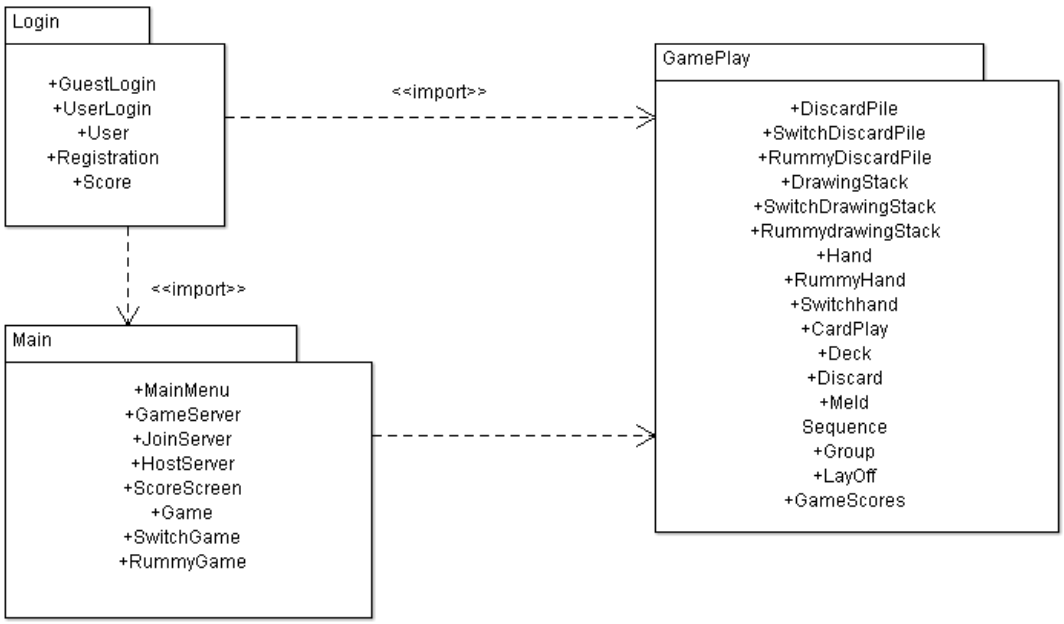

Image 5.2.1

View full size image from here.