The User Requirements Specification document contains requirements which the application will meet and the features it will possess.

User Requirements Specification

Fight The Landlord

Mengchuan Liu

Jiefan Lin

Jianfei Feng

Jiaqi Ni

Wen Zhang

KalinaPetrova

Contents

[1. Introduction 2](#_Toc463554204)

[2. Game Introduction 3](#_Toc463554205)

[2.1 Players, Cards, and Deal 3](#_Toc463554206)

[2.2 Valid cards types (combinations) 3](#_Toc463554207)

[2.3 Auction 4](#_Toc463554208)

[2.4 Play rules 4](#_Toc463554209)

[2.5 Example of expected cooperation 4](#_Toc463554210)

[2.6 Scoring 5](#_Toc463554211)

[3. Functional Requirements (M: Mandatory HD: highly desired) 6](#_Toc463554212)

[3.1 General Features 6](#_Toc463554213)

[3.2 Use Cases 7](#_Toc463554214)

[ Register 7](#_Toc463554215)

[ Login 7](#_Toc463554216)

[ Logout 8](#_Toc463554217)

[ Determine The Landlord 8](#_Toc463554218)

[ Give Leftover Cards to Landlord 9](#_Toc463554219)

[ Determine The Order 9](#_Toc463554220)

[ Select Cards 10](#_Toc463554221)

[ Hand Over Cards 10](#_Toc463554222)

[ Give Hints to Player 11](#_Toc463554223)

[ Record Total Scores 11](#_Toc463554224)

[ Pass In The Turn 12](#_Toc463554225)

[ Game Finish 12](#_Toc463554226)

[ Reset Password 12](#_Toc463554227)

[3.3 MOSCOW 13](#_Toc463554228)

[4. System Requirements 14](#_Toc463554229)

[Hardware 14](#_Toc463554230)

[Software 14](#_Toc463554231)

[5. Non-functional Requirements 15](#_Toc463554232)

# Introduction

The User Requirements Specification document contains requirements which the application will meet and the features it will possess. Every software application which is to be developed needs to meet three kinds of functional requirements, system requirements and non-functional requirements.

Non-functional requirements of an application have to deal with its quality aspects such as how difficult it is to use it, how efficient, is it reliable and how maintainable is it.

# Game Introduction

Fight the Landlord (Dou Di Zhu) is a climbing game primarily for three players. In the game, one player, the “landlord”, plays alone and the others form a team. The landlord’s aim is to be the first to play out all his cards in valid combinations, and the ‘Farmer’ team wins if any one of them manages to play all their cards before the landlord. The game is very popular all over China and is also extensively played online.

## Players, Cards, and Deal

This game uses a 54-card pack including two jokers, red and black. 4 suits (Spade, Heart, Club, Diamond) with 13 cards in each.

The cards rank from high to low:

**-Red joker, black joker, 2, A, K, Q, J, 10, 9, 8, 7, 6, 5, 4, 3.**

Suits are irrelevant (no ranking between suits).

Cards are shuffled randomly, distributed to players counter-clockwise around the table until each player has 17 cards. Each player can only see his own cards. The last three cards are visible to everyone and belong to the landlord. So landlord has a total of 20 cards.

## Valid cards types (combinations)

In this game, there are thirteen types of combination that can be played:

1. Single card - ranking from three (low) up to red joker (high) as explained above
2. Pair - two cards of the same rank, from three (low) up to two (high)
3. Triplet - three cards of the same rank
4. Triplet with an attached card - a triplet with any single card added, for example 6-6-6-8. These rank according to the rank of the triplet - so for example 9-9-9-3 beats 8-8-8-A
5. Triplet with an attached pair - a triplet with a pair added, like a full house in poker, the ranking being determined by the rank of the triplet - for example Q-Q-Q-6-6 beats 10-10-10-K-K.
6. Sequence - at least five cards of consecutive rank, from 3 up to ace - for example 8-9-10-J-Q. Twos and jokers cannot be used.
7. Sequence of pairs - at least three pairs of consecutive ranks, from 3 up to ace. Twos and jokers cannot be used. For example 10-10-J-J-Q-Q-K-K.
8. Sequence of triplets - at least two triplets of consecutive ranks from three up to ace. For example 4-4-4-5-5-5.
9. Sequence of triplets with attached cards - an extra card is added to each triplet. For example 7-7-7-8-8-8-3-6. The attached cards must be different from all the triplets and from each other. Although triplets of twos cannot be included, a two or a joker or one of each can be attached, but not both jokers.
10. Sequence of triplets with attached pairs - an extra pair is attached to each triplet. Only the triplets have to be in sequence - for example 8-8-8-9-9-9-4-4-J-J. The pairs must be different in rank from each other and from all the triplets. Although triplets of twos cannot be included, twos can be attached. Note that attached single cards and attached pairs cannot be mixed - for example 3-3-3-4-4-4-6-7-7 is not valid.
11. Bomb - four cards of the same rank. A bomb can beat everything except a rocket, and a higher ranked bomb can beat a lower ranked one.
12. Rocket - a pair of jokers. It is the highest combination and beats everything else, including bombs.
13. Quadplex set - there are two types: a quad with two single cards of different ranks attached, such as 6-6-6-6-8-9, or a quad with two pairs of different ranks attached, such as J-J-J-J-9-9-Q-Q. Twos and jokers can be attached, but you cannot use both jokers in one quadplex set. Quadplex sets are ranked according to the rank of the quad. Note that a quadplex set can only beat a lower quadplex set of the same type, and cannot beat any other type of combination. Also a quadplex set can be beaten by a bomb made of lower ranked cards.

## Auction

There is an auction to determine which player will be the landlord, and play alone against the other two. The possible bids are 1, 2 and 3. The player who is the first to bid is randomly picked. Each player, in turn, may either pass or bid higher than the highest bid so far. If everyone passes the hand is thrown in and there is a new deal. If there is a bid, the bidding continues counter-clockwise, each player passing or bidding higher than the previous bidder until there are two consecutive players pass or someone bids 3, which ends the auction since it is the highest possible bid. The final and highest bidder is the landlord. This player now picks up the three leftover cards from the middle, for a total of 20 cards.

## Play rules

The landlord plays first, and may play any single card or any legal combination. The played cards can be seen by everyone. Each subsequent player in counter-clockwise order must either pass (play no card) or beat the previous play by playing a higher-ranking combination of the **same number of cards** and **same type**. (No ranking between different types or different number of cards).

Example: 2 aces cannot beat 3 tens.

Note: If the combination type is a triplet, QQQ can beat 999, QQQ-77 can beat 999-55, QQQ-7 can beat 999-5. But QQQ cannot beat 999-55 or 999-5, the number of cards and combination have to be same.

The play continues around the table for as many circuits as necessary until two consecutive players pass. The person who played the last card(s) begins again, leading any card or a legal combination.

Note that passing does not prevent you from playing on a future turn.

## Example of expected cooperation

Player A (the landlord) leads 3-3-3 to get rid of some low cards, player B passes, player C plays 5-5-5, player A plays K-K-K and player B plays A-A-A. C and A pass, so B can start again with anything. He leads a single 4.

Note B could have played his aces on his first turn, but preferred to pass to give his partner a chance to get rid of some cards. C will now play if possible, so as not to give the landlord (A) a free chance to lead again. Having beaten A’s second play, B leads a low card to give C the choice of playing another unwanted card of putting the landlord under pressure by playing a high card.

## Scoring

If the landlord runs out of cards first he has won, and each opponent pays him the amount of the bid - 1, 2 or 3 units - provided that no bomb or rocket was played. If one of the other two players runs out before the landlord, the landlord loses and must pay the amount of the bid **to each opponent**. For each occasion when any player played a bomb or rocket, the payment for the hand is doubled. So for example in a hand in which two bombs and a rocket were played, a player who bid 3 will win 24 points from each opponent for going out first, or pay 24 to each opponent if another player goes out first.

# Functional Requirements (M: Mandatory HD: highly desired)

## General Features

|  |  |  |
| --- | --- | --- |
|  | Name | Description |
| Gameplay | Select cards | The player can reselect cards he wants to play |
| Determine first chance to fight to be landlord | Randomly give the chance which player can bid the first |
| Determine landlord | Players fight to be landlord by adding bid |
| Shuffle cards | Randomly reorder all cards |
| Displays cards | Display all the cards player is current holding (High - Low) |
| Check validation of selected cards | When a player gets rid of cards, check the combination is valid and show a proper warning message when the combination is not beatable |
| Pass in the turn | A player is able to pass this turn and keep current cards |
| Give hint to player | A player is able to let the system to suggest him get which card(s) out |
| Highlight selected cards | When the player selects a card, the card will move a bit up |
| Game finish | Give result to all players when game ends |
| Enable / disable background music | Players can disable/enable the background music. |
| Enable / disable sound effect | Players can disable/enable the sound effect. |
| Account system | Register | Create a new account |
| Login | Login with an existing account |
| Reset password | User who has an account can reset password |
| Logout | Logined user can logout |
| Scoring System | Record total scores | The system will record total score of every user |
| Earn 100 score per day | A user can have 100 free scores everyday |
| Check playable of player | If the score is 0 or below, the user cannot continue to play this day. |
| Lobby | Create a room | A user is able to create room |
| Join room | A user is able to select an existing room and join |
| Control the created room | The user who creates the room can start the game when the other two users are ready |

## Use Cases

### Register

Goal**:** The user has created an account with which he can be identified inside the game.

Actor**:** User

Main Success Scenario**:**

1. User clicks on the registration button.
2. System redirects the user to the registration form.
3. User types in username.
4. User types in a password.
5. User retypes the password in a second field.
6. User types in his email.
7. User clicks to confirm his account information.
8. System adds the user data to the database.
9. System informs the user of his success and sends him back to the login screen. (Go to Use-case Login)

Extensions**:**

7.1 Username already exists in the database

7.1.1 System informs the user that his selected username is taken and should be changed.

7.2 Username contains illegal character

7.1.2 System informs user and redirects him back to the registration form.

7.3 Specified password contains illegal characters.

7.3.1 System informs the user of password violation and redirects the user back to the registration form.

7.4 The two provided passwords do not match.

7.4.1 System informs the user that the retyped password does not match the first one and redirects the user back to the registration form.

7.5 Provided email is not a valid one.

7.5.1 System informs the user and asks him to provide a real email address.

8.1 System was unable to update the database

8.1.1 System informs that there was a problem with connecting to the database and redirects user back to the registration form.

### Login

Goal: User has logged in with a valid account and is able to play the game.

Actor: User

Main Success Scenario:

1. User opens the application.
2. System redirects the user to the login page.
3. User types in username.
4. User types in password.
5. User clicks confirm.
6. System informs that user has been successfully logged in.
7. System extracts all needed information about the user.
8. System redirects the user to the main game lobby.
9. User is added to the online player list.

Extensions:

2.1 User does not have an account yet.

2.1.1 User could choose to create one (Go to use-case “Register an account”.)

4.1 User has forgotten his password.

4.1.1 User can choose to reset his password. (Go to use-case “Reset password.”)

5.1 System could not find the combination of username and password.

5.1.1 System informs the user and redirects him back to the login screen.

### Logout

Goal: The user has successfully exited the game.

Actor: User

Main Success Scenario:

1. User is logged in the game lobby.
2. User clicks the logout option.
3. System removes the user from list of online players.
4. System redirects the user back to the login screen.

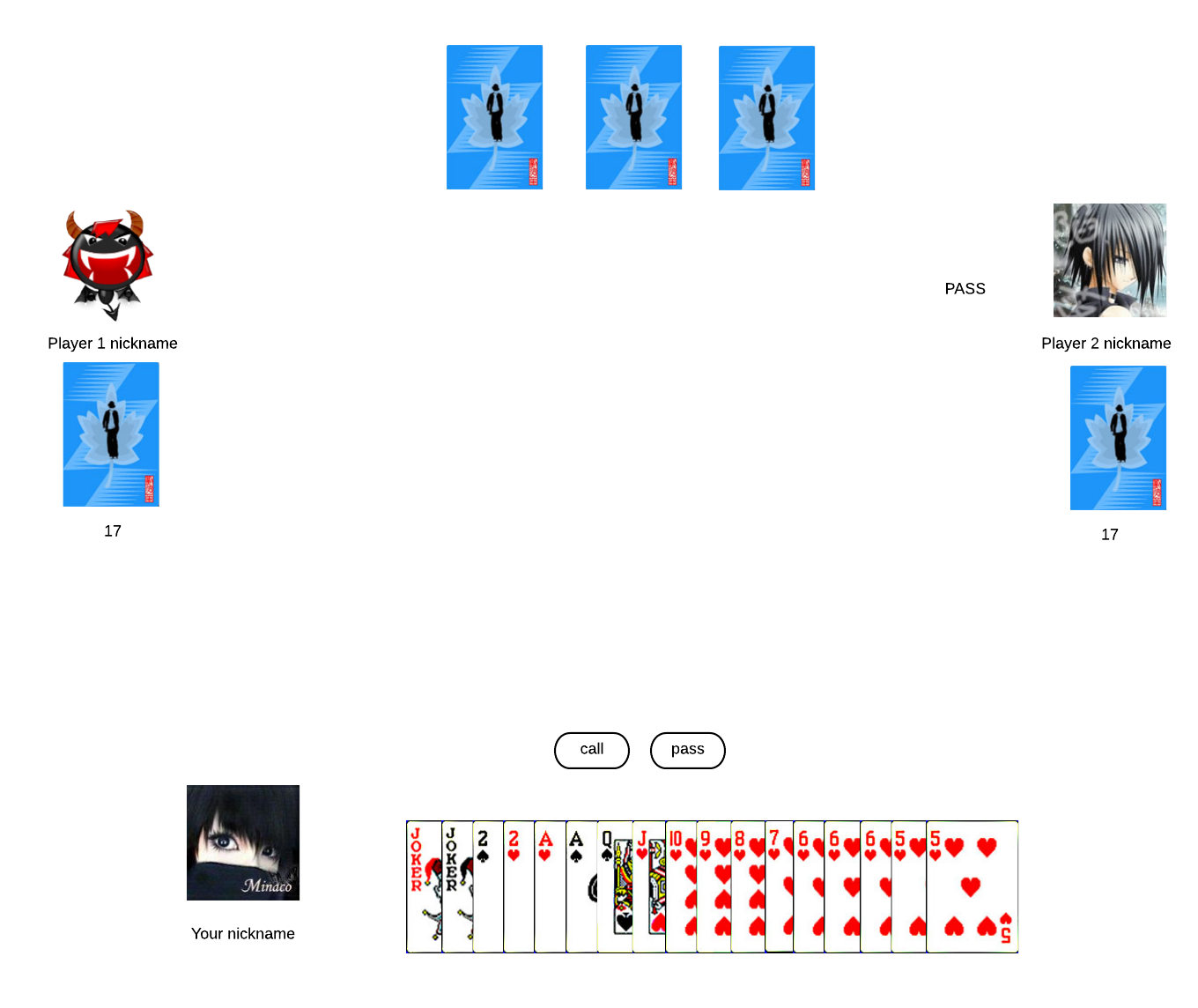
Extensions:

* 1. User is not logged in.
     1. Option to logout will not exists for the user.

2.1 User closes the application without logging out.

2.1.1 System will automatically log out the user.

### Determine the Landlord



Goal: Assign roles to each player, one is a landlord and the rest are farmers.

Actor: Players

Pre-condition: Cards are shuffled randomly, distributed to players counter-clockwise around the table until each player has 17 cards. Each player can only see his own cards. The last three cards are visible to everyone and belong to the landlord. The very beginning multiple is 15 and base point is 10.

Main Success Scenario:

1. The system will randomly determine who the first turn to pick a role is. The player is able to refuse by clicking “Farmer” or agree by clicking “Landlord”.
2. The second player is able to pick a role as well.
3. Now it’s the third player’s turn.
4. Assign 1 landlord and 2 farms and give these three cards to the landlord.
5. The landlord is ready to get rid of his card(s).

Extensions:

1.1 No matter the first player picks landlord or farmer, the multiple stay same which is 15. Give turn to next player.

2.1 If the second player also picks a landlord, the bid will become to 15 \* 2 = 30.

2.2 If the second player also picks the farmer, the bid stays 15.

3.1 If the previous two players picked landlord and the third player picks landlord, the bid will become to 15 \* 2 \* 2 = 60. In other words, the first time picks landlord the multiple is 15 if someone else picks landlord the bid will be doubled.

3.2 If the previous two players picked farmer and the third player picks farmer, that means no one wants to be a landlord, the game will restart again and the cards will be shuffled again.

3.3 During the first round, if there’s one player picked landlord, the rest picked farmer. The one who picked landlord will become a landlord.

3.4 During the first round, if there’s more than one player picked landlord, it will go to another round until a player picks landlord twice.

### Give Leftover Cards to Landlord

Goal: the landlord takes the leftover cards

Actor: sever

Pre-condition: the landlord is determined

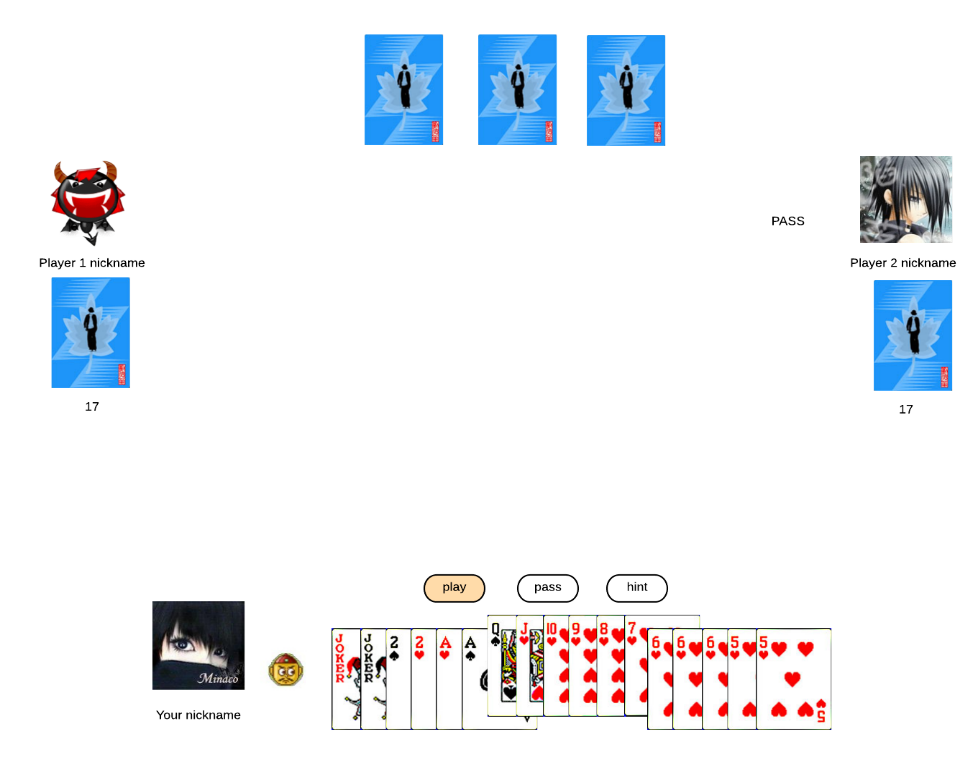
Main Success Scenario:

1. The system will turn over the 3 leftover cards
2. System adds the leftover cards to landlord’s cards
3. System will show that the total amount of landlord’s cads is 20
4. System displays the number of bid system

Exception:

* 1. There is one joker inside leftover cards
     1. System multiple 2 with current bid number
     2. System displays the number of bid system
  2. There are two jokers inside leftover cards
     1. System multiple 4 with current bit number

### Select Cards



Goal: The user select the card that he wants to give out

Actor: User

Pre-condition: It is the player’s turn

Main Success Scenario:

1. The user click on the ideal cards.
2. The user press play button.
3. The system check the rationality of the cards.
4. The system display the cards to all players

Exception:

3.1 The selected cards are not qualified.

3.1.1 The system pop-up the error warning.

3.1.2 The system goes to step 1.

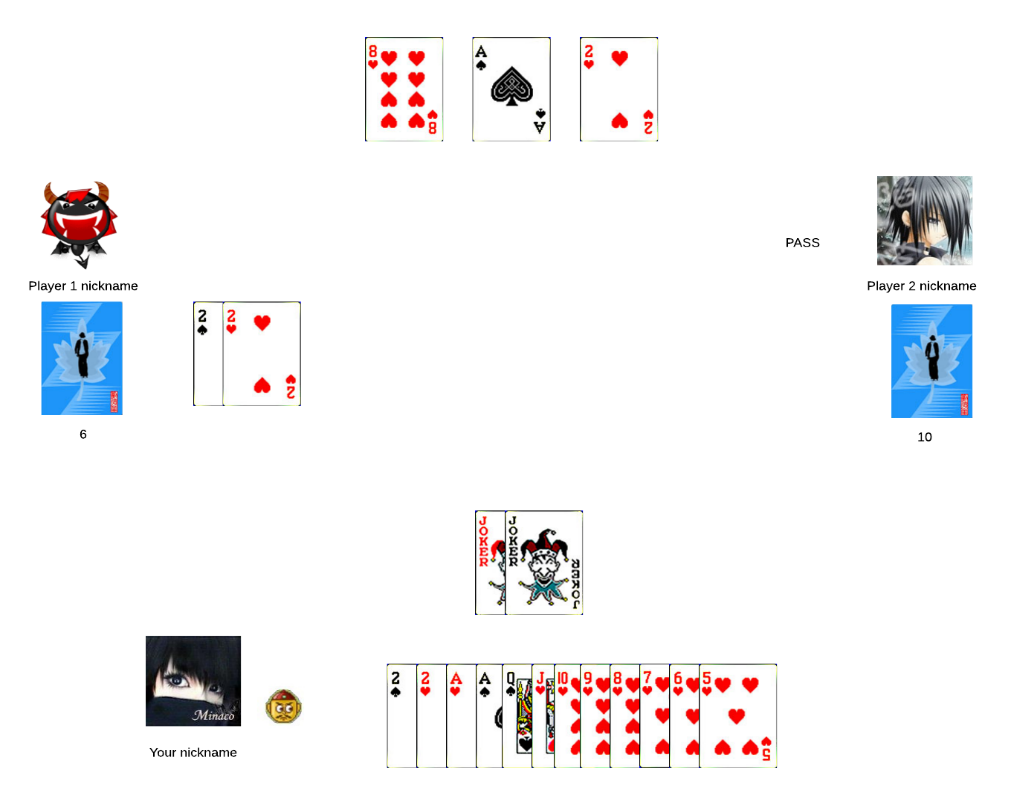
3.2 The selected cards are not qualified.

3.2.1 The system pop-up the error warning.

3.2.2 The user press pass button.

Post-condition: The next player starts to play.

### Hand Over Cards



Goal: players hand over the selected cards

Actor: player

Pre-condition: player has selected cards

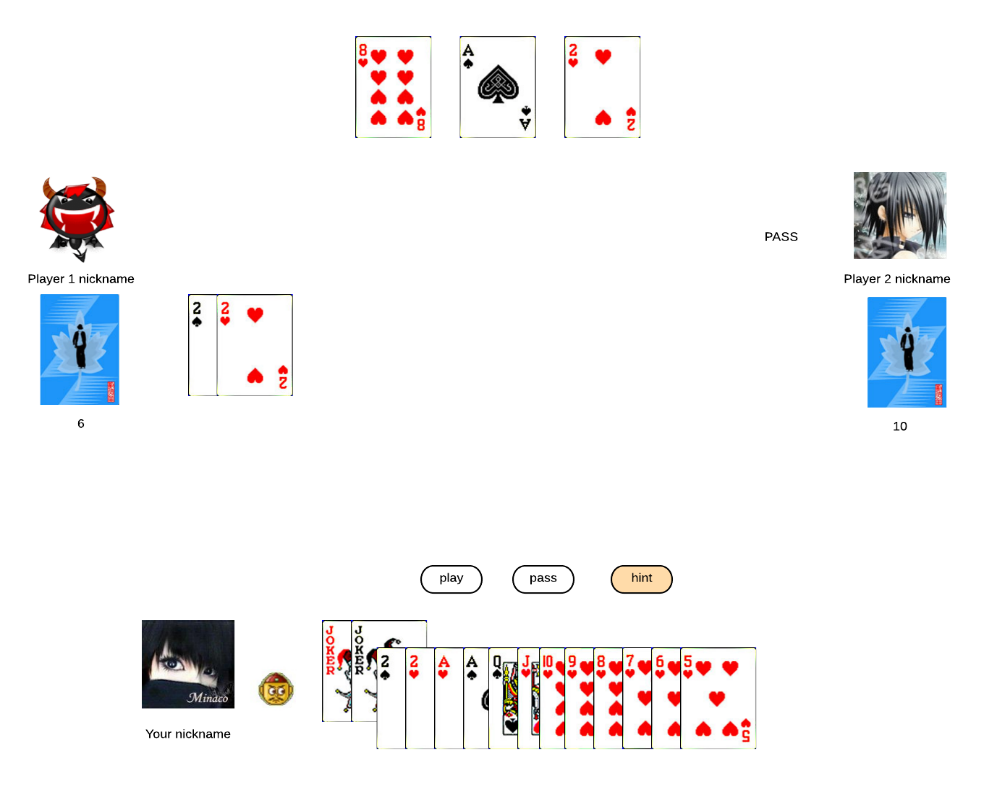
Main Success Scenario:

1. Player clicks on play to hand over the selected cards
2. The system displays the cards from landlord in the middle of the table
3. The system removes the handover cards from player’s cards
4. The system recalculates the total number of player’s cards
5. The system shows the amount of player’s cards

Exception:

* 1. The player selects illegal combination
     1. System displays an error message
  2. the selected cards is smaller compared with previous player
     1. system displays an error message
  3. The player doesn’t click on play within the 20 seconds
     1. System automatically hands over the smallest card
  4. The player clicks on hint
     1. Go to the hint use case
  5. The player 1 clicks on pass
     1. Go to the pass use case

### Give Hints to Player



Goal: The system will give the player a hint to deal the cards.

Actor: User

Pre-condition: It is the player’s turn

Main Success Scenario:

* 1. The player press hint button.
  2. The system brings up the qualified cards.
  3. The press the play button.
  4. The system display cards to all players.

Exception:

3.1 The user presses hint button again.

3.1.1 The system goes to step 2.

3.2 The user press the pass button.

3.2.1 Go to pass use case.

Post-condition: The next player starts to play.

### Pass In the Turn



Goal: User has to click pass button to give up one turn play card   
Actor: User

Pre-condition: Players must keep at least 1 card, game already started and players only allowed to do this operation after other players play a card

Main Success Scenario:

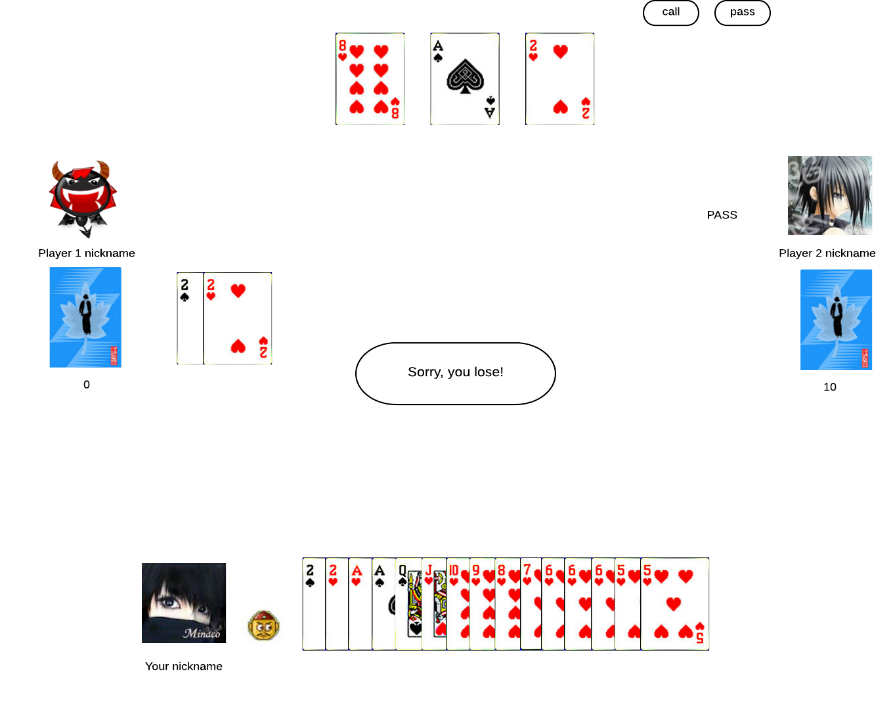
1. Other players played a card
2. System calculate whether player have card to beat it
3. System found no possibilities to beat the card and show pass button
4. Player click pass button and skip this turn

Exception:

* 1. Players didn’t do any operation during the set time
     1. System will do pass function automatically after set time over.

Post-condition: Player has to click pass button to skip this turn.

### Game Finish



Goal: User finished match and system show prompt message win or lose

Actor: User

Pre-condition: At least there is one player get rid of all the cards

Main Success Scenario:

1. Player get rid of the last card
2. System show “congratulation, you win！” or “Sorry, you lose”

### Record Total Scores

Goal: Summarize total score each player gains during this game.

Actor: Players

Pre-condition: There’s one player got rid of all his cards out.

Main Success Scenario:

1. A summary window will pop up and shows landlord/farmers wins. It summarizes total score each player gains during this game.
2. The player can choose to continue the game on the same table or quit from this table.

Extensions:

* 1. If the landlord wins he will get: base score \* bid. Each farmer will lose: base score\* bid/2
  2. If the landlord loses he will lose: base score \* bid. Each farmer will get: base score\*bid/2
  3. If the player’s total score is 0, he cannot continue to play within an hour/day. A proper message box will notice him.

Exception:

* 1. Player has less than 3 cards left in the hand.
     1. System will show prompt message to tell other players how many cards left in this player

Post-condition: User win/loose the match and get prompt message

### Reset Password

Goal: The user has successfully reset his password and can login again.

Actor: User

Main Success Scenario:

1. User access the login page.
2. User clicks on the “Forgotten password” button.
3. System redirects user to another form.
4. User types in his email address.
5. System sends an email with a link to change password.
6. User clicks on the link provided in the email.
7. On the resulting screen, user types in his new password twice for confirmation.
8. User clicks confirm.
9. System updates
10. System displays a confirmation message.

Extensions:

* 1. User did not provide a valid email address.
     1. System will not send a message.
  2. System could not find email address in the database.
     1. System will inform the user that no user exists with the specified email address.
  3. Specified passwords do not match.
     1. System informs the user and asks him to retype the password correct.

## MOSCOW

Must

* + Join room
  + Select cards
  + Determine first chance to fight to be landlord
  + Determine landlord
  + Shuffle cards
  + Display cards
  + Check validation of selected cards
  + Pass in the turn
  + Give hints to player
  + Highlight selected cards
  + Record total scores

Should

* + Register
  + Login
  + Logout
  + Check playable of player

Could

* + Reset password
  + Create a room
  + Control the create room
  + Enable / disable background music
  + Enable / disable sound effect

# System Requirements

## Hardware

|  |  |
| --- | --- |
| Requirement | Preference |
| The following are the recommended Server hardware requirements,the system can be run in a virtual environment and on a physical server. In both cases, these are the recommended requirements. It is recommended to run 2 servers, whereby the service and database run on a different system.  **CPU:** At least a dual core CPU  **RAM:** On Windows Server 2003/Windows XP at least 2GB and Windows Server 2008/Windows 7 and later user at least 4GB  **HDD:** At least 10GB of free hard drive space  **GPU:** The server has no requirements for the GPU since the server will not use it for computing. | M |
| The following requirements are recommended for the client application  **CPU:** At least a dual core CPU  **RAM:** At least 2GB of RAM  **HDD:** At least 1GB of free hard drive space  **GPU:** A system with Integrated graphics should work fine | M |

## Software

|  |  |
| --- | --- |
| Requirement | Preference |
| The software requirements for the server follows. It is recommended to run a version of Windows Server to run the server. The minimum requirement for Operating system is Windows Server 2003. But it is recommended to run at least Windows Server 2008R2. On the server, it needs to have .NET 4.5 installed. | M |
| The recommended software requirements for the clients is any Windows Operating System with .NET4.5 installed | M |

# Non-functional Requirements

Of course when using an application the things that can bother us or make us happy are not always related to the product’s functionality. What about Usability, Reliability, Performance and Maintainability?

1. Usability – The application will not be intrusive and will have an easy to understand GUI. The application’s interaction will be intuitive.

Usability will be tested using actual user interaction. We will present a group of people that represent our user base with the demo products and monitor how they navigate through the application. Using these observations we will determine which parts are alright as they are and which need modification. We will conduct these user tests during our testing period of each iteration.

Since we are creating a logical card game application, it would be beneficial to present the rules of the game and the different functionalities to the user using a small tutorial to get them used to the application.

1. Performance – This is an important aspect in any Simulation. The application will not take up unnecessary computer memory or computing time. It will match with efficiency standards allowing the simulations to run smoothly.

We will try to test the application on various machines and devices to ensure that the performance is not constrained by any specific requirement. (CPU, GPU, RAM)

1. Reliability – All the simulations from the software will be accurate and precise. They will provide with an ideal situation of traffic control that is not influenced by forces not stated by the client. Any outside tempering with the system will be prevented and prohibited.

During our user testing we will also make sure that the users get the desired results and that there are no exceptions to the rules.

We will also try to cover the code with unit testing as much as possible to make sure no bugs occur due to unexpected input.

1. Maintainability – The software will be provided with error protection protocols. The software will be written in proper standards, making it easy to be accessed from third parties.
2. Extensibility – We will make sure that the code is as clean as possible and structured in a nice way so that there won’t be any difficulty when adding additional features.

To ensure that our architecture and code are extensible and maintainable, we will try to research on all the possible functionalities that could be implemented into our application and take those into consideration when designing the system.