PokerGame Requirements

1 The poker game contains the interaction of a server and multiple clients, and 2 utilities for card representation called Card and type checking called CardTypeSystem.

2 Every card in the poker game represents by integer from 1 - 54. The Card class has a attribute integer for it. The class provides 3 methods,
- 2.1 GetSuit() return the suits in one enumerator for club, spades, hearts and diamonds.
- 2.2 GetNumber() return the number for the card. (AJQK is for 1, 11, 12, 13, little joker for 14, big joker for 15).
- 2.3 GetPriority() to get the integer priority for a single card. The card number in the order of ascending priorities are 3, 4, 5, 6, 7, 8, 9, 10, J, Q, K, A, 2, little joker, big joker.

3 CardTypeSystem saves all the CardTypeClass in a predefined ordered list and give a integer handle to access a specific one. This provides API to server and clients to do card type checking (See in card type checking in section Gameplay).

4 ClientFirstUI:
- 4.1 Display Login and Register form.

5 Register:
- 5.1 Form contains email, nickname, password, password confirm and a submit button.
- 5.2 Password should be shown in '*'.
- 5.3 Upon button clicked, client-side check email, nickname, password format and the fit for password confirm, if not passed, return the result.
- 5.4 Client block user input, and send the form to server.
- 5.5 server check email uniqueness, return the result to client.
- 5.6 client render the result.
- 5.7 Goto Login.

6 Login:
- 6.1 Form contains email, password, submit button.
- 6.2 Upon submit, client block user input and send form to server and return result if not passed.
- 6.3 Server send uid, nickname and score to client.
- 6.4 Goto LoggedIn.

7 LoggedIn:
- 7.1 Display nickname and score, and BeginGame and Logout button, goto BeginGame and Logout respectively.

8 ServerUserDatabase:
- 8.1 Test: 3 user added in advance.
- 8.2 Contain email, nickname, password, score, online status, uid.

9 BeginGame:
- 9.1 Upon BeginGame button clicked, client block the input,
  - 9.1.1 show a cancel button.
  - 9.1.2 send message to server.
- 9.2 if the cancel button is clicked, client block the input and sends a signal to server, server stops the finding procedure and return confirmation signal.
- 9.3 if confirmation received, goto BeginGame.
- 9.4 Server finds players,
  - 9.4.1 -wait until 3.
  - 9.4.2 -according to score.
- 9.5 return gamestart signal, room id, client's the random allocated posid in the room, the other players posid, nickname and score to client.
- 9.6 if start game received, goto GamePlay.

10 GamePlay:
- 10.1 The gameplay is controlled by server. Server asks the clients to do specified action with a timeout.
- 10.2 Server contains 3 players, each with posid and uid, hand cards, the passed sign for a round, and the bet, HandInfo, current hand posid, landlord posid and back cards for the game.
- 10.3 Clients contains 3 players, each with posid and uid, number of hand cards, nickname, score, passed sign and also contains the gameplay message: bet, current hand posid, current landlord posid, HandInfo and my hand cards.
- 10.4 Client has a Surrender button to break the game. Server will send it to other clients and goto EndGame.
- 10.5 Deal:
  - 10.5.1 Server shuffled the 54 cards. Keep 3 back cards and deal. Send the deal result to clients.
  - 10.5.2 Client receives the result and show.

10.6 LandlordGrab:

10.6.1 Server randomly chooses a client to be landlord, send to other clients.

10.6.2 Server randomly chooses a client to ask for Landlord first. The message contains isFirstOne, currentBet.

10.6.3 LandlordGrab loop:

10.6.4 Send messages to other clients for waiting.

10.6.5 The client has four options: +1, +2, +3, pass. Send it to server.

10.6.6 The first player cannot choose pass. The following players are only able to ask higher score or pass.

10.6.7 if +3 is chosen or 2 continuous players choose pass, the landlord is chosen and send the landlord id to the clients.

10.6.8 If not, server update the currentBet. Send the result to other clients.

10.6.9 Clients show the result.

10.6.10 Server ask next player according to posid.

10.6.11 LandlordGrab loop end.

10.6.12 The back cards are dealt to the landlord, send to all clients.

10.6.13 The client show the back cards and the landlord sign, update the display of hand cards after insert the back cards.

10.7 GiveHand:

10.7.1 Client and Server maintain a cardType and a priority number in that type, called HandInfo for the cards of current round.

10.7.2 giveHand loop:

10.7.3 Server send a message of giveHand, contains a sign of first hand of the round to the client.

10.7.4 Server send messages to other 2 clients for waiting and reset the round info if necessary.

10.7.5 the giving client activates give and pass buttons,

10.7.6 if this is the first hand of the round,

10.7.6.1 pass button is disabled.

10.7.6.2 reset client cardType.

10.7.7 Client choose cards and when give buttons pressed,

10.7.8 Do a card type check,

10.7.9 if check passed, submit the card, if not, give a warning.

10.7.10 if pass button pressed, submit a pass signal.

10.7.11 Client deactivates the give and pass buttons.

10.7.12 Server do a validation for the current hand card including the giving hand and another type check, if failed, see as a surrender message. // Security issue.

10.7.13 client and server update the hand cards.

10.7.14 Server send the cards and posid to other 2 clients to display.

10.7.15 Other clients receive the cards, display it and update the card number of that user and update the round info.

10.7.16 The server check endgame condition (no cards left), if so goto endgame.

10.7.17 The server check the passed sign to determine whether it is a new round and who is the next giveHand Client. If a new round, reset Server passed sign.

10.7.18 giveHand loop End.

10.8 CardType Check:

10.8.1 struct CardType contains CardTypeClass handle and CardType Auxiliary data.

10.8.2 struct HandInfo contains CardType and Priority.

10.8.3 CardType Auxiliary data is necessary with more types provided in a CardTypeClass, like a Straight for different number of cards. Specially, CardTypeClass handle 0 indicates no cardType (for first hand per round.).

10.8.4 The aux and priority data is only used with corresponding CardTypeClass, and may not be used.

10.9 CardTypeSystem:

10.9.1 FindType() : receive cards and return HandInfo struct. cardTypeClass handle -1 for not found.

10.9.2 BiggerCheck(): receives 2 HandInfo data, returns if the first hand is bigger than the second. Its Activity Diagram is shown in Figure 1.

10.9.3 IsBomb(): receives a cardTypeClass handle and return if is a bomb. (used in other cardTypeClass checking API.)

10.10 ICardTypeClass:

10.10.1 This is an interface for specific card type classes.

10.10.2 Every implementation of ICardTypeClass has a StaticCheck() method and an isBomb() method.

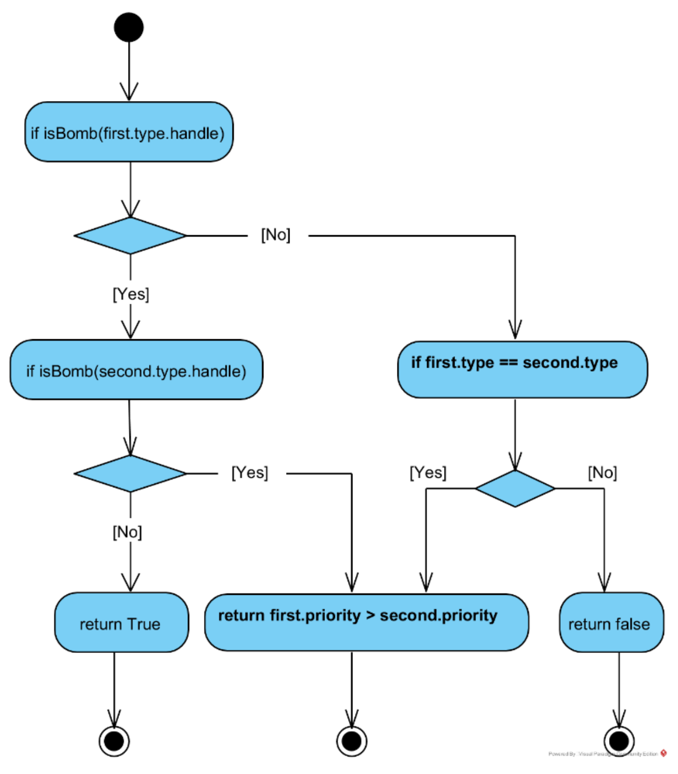

if isBomb(first.type.handle)
[No]
[Yes]
if isBomb(second.type.handle)
if first.type == second.type
[Yes]
[Yes]
[No]
[No]
return True
return first.priority > second.priority
return false

Figure 1 BiggerCheck() Activity Diagram

10.10.3 The StaticCheck() receives a list of card, and check if the cards is of this type, then return HandInfo struct, with -1 CardTypeClass Handle if check failed.
10.10.4 IsBomb() returns if the card type is a bomb.

10.11 Card type classes
10.11.1 when check cards, the order for these type classes must be obeyed:
10.11.2 All bombs share the same set of priority value.
10.11.3 Rocket: 2 jokers, bomb, priority is 999.
10.11.4 Bomb: 4 same number, bomb, priority is the single priority of that number.

10.11.5 **All straights are judged in priority, start at 3 and finish at A
10.11.6 SingleStraight: 34567…, aux is the number of cards, priority is the largest single priority of the number in straights.
10.11.7 DoubleStraight: 334455…, aux is the number of pairs, priority is the largest single priority of the number in straights.
10.11.8 TripleStraight: 333444…, aux is the number of triples, priority is the largest single priority of the number in straights.
10.11.9 QuadsPlus: 3333xy or 3333xxyy, aux is the type of surplus cards (single or pair), priority is the single priority of the number of the Quads. // above the airplane because 33334444 is not 333-444-34.
10.11.10 Airplane: 333444…xy… or 333444…xxyy…, aux are the number of triples and the type of surplus cards, priority is the largest single priority of the number in triple straights.

10.11.11 TriplePlus: 333x or 333xx, aux is the type of surplus cards, priority is the single priority of the triple.
10.11.12 Triple: 333, priority is the single priority.
10.11.13 Pairs: 33, priority is the single priority.
10.11.14 Single: 3, priority is the single priority.

11 EndGame:
11.1 Server send EndGame message to the clients with the posid and winning score. Client display the result, show BeginGame button and Logout button for goto BeginGame and Logout section.

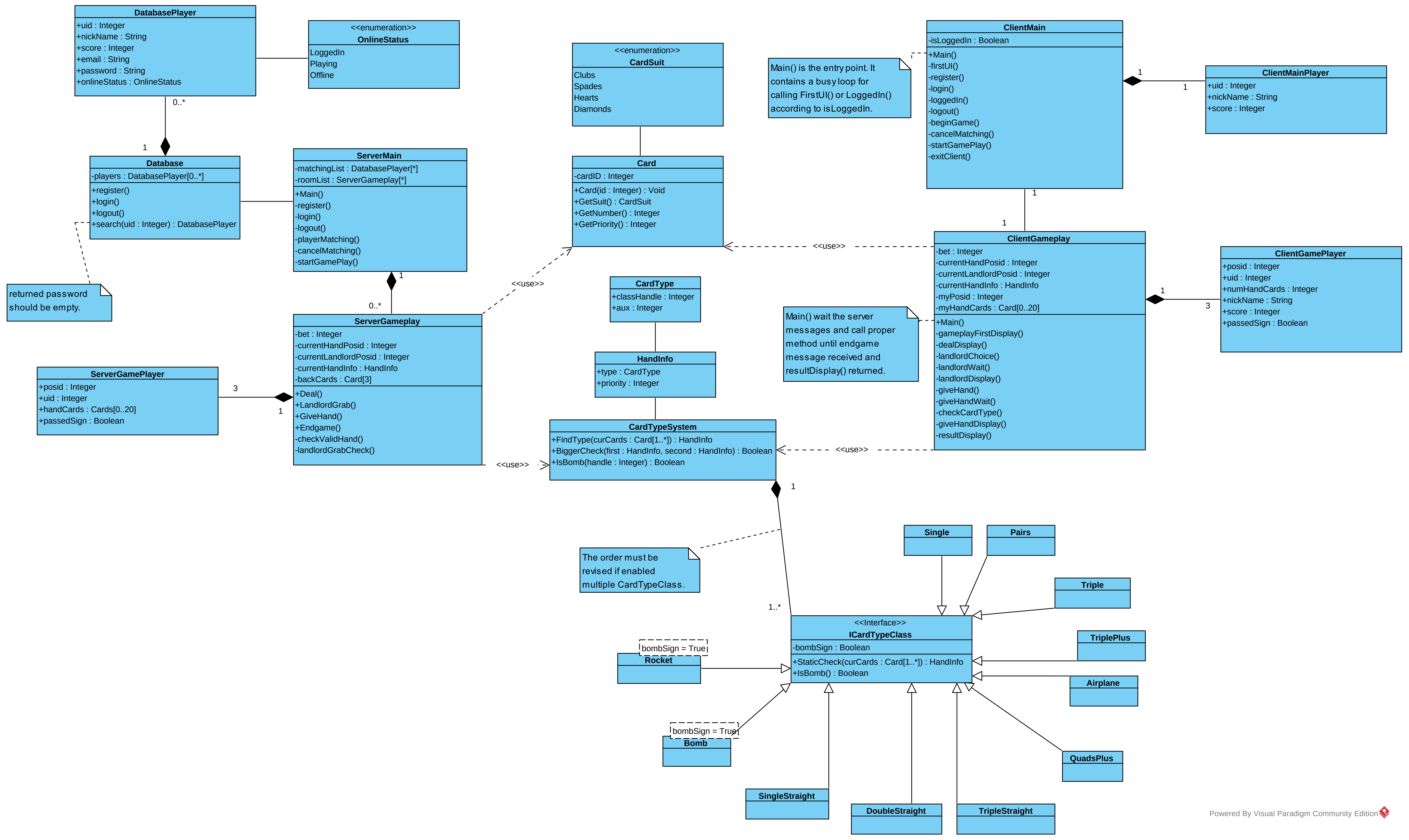
12 Logout:
12.1 send a message to server.
12.2 client goto ClientFirstUI.

13 Better UI

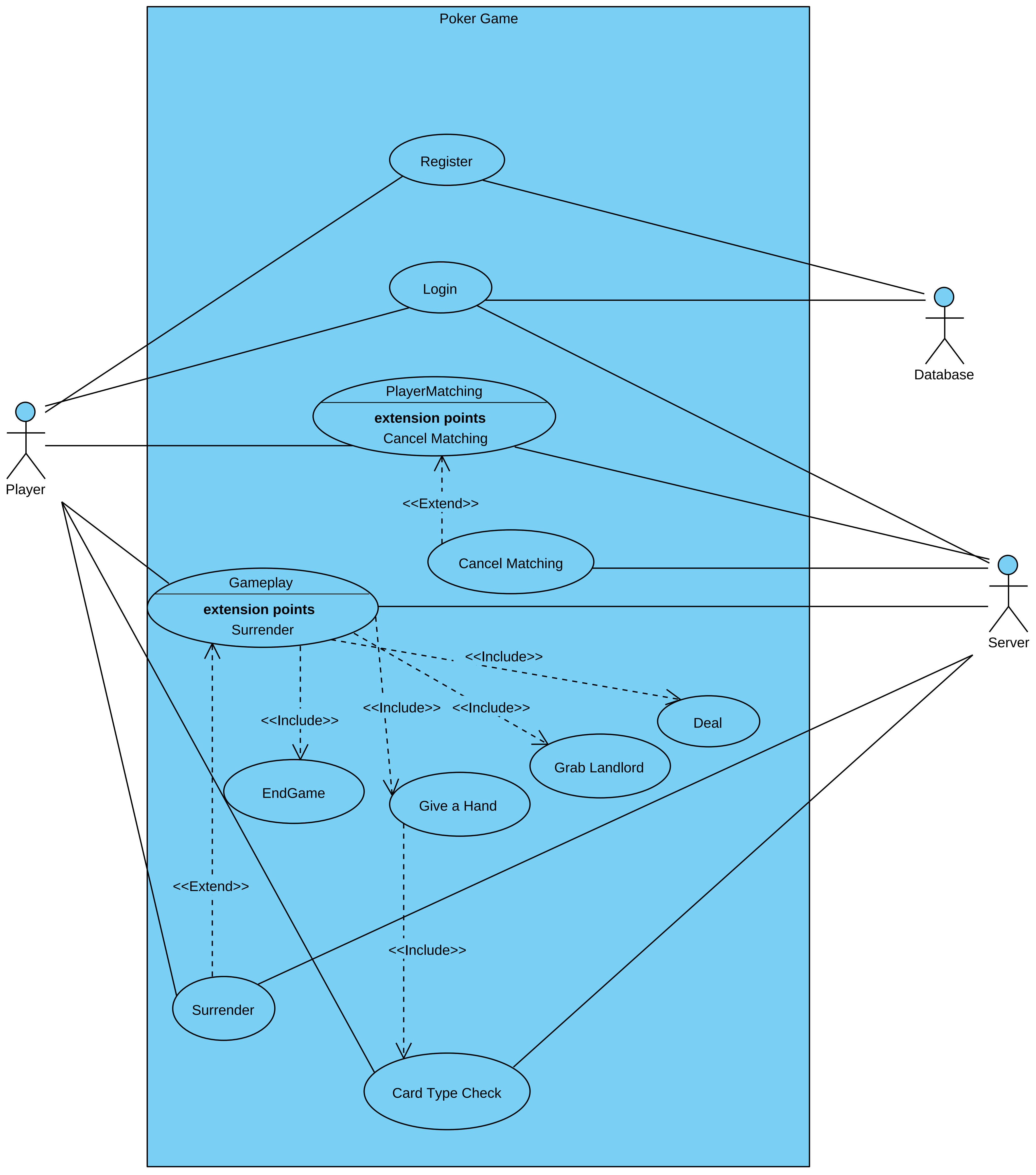
Requirements Priority & Iteration Plan:

Critical: minimal to get the system run.
Important: minimal to get the card type system run.
Expected: minimal to complete all the rules for 3-people “Fight against the landlord” game.
Additional: additional features providing better user experience.

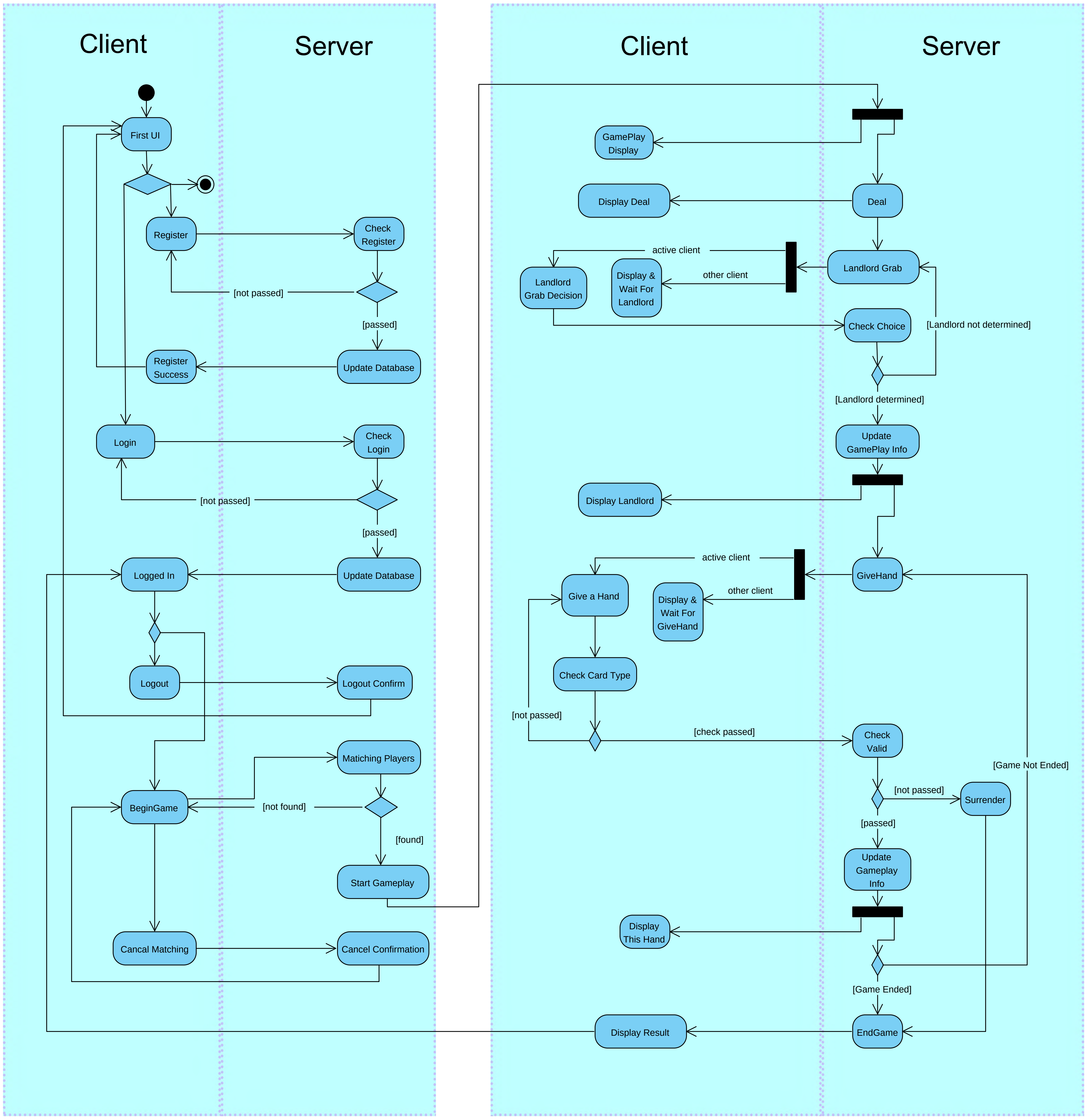
System Composition



Use Cases


Poker Game
Register
Login
Database
PlayerMatching
extension points
Cancel Matching
Player
<<Extend>>
Cancel Matching
Server
Gameplay
extension points
Surrender
<<Include>>
<<Include>>
<<Include>>
<<Include>>
Deal
Grab Landlord
EndGame
Give a Hand
<<Extend>>
<<Include>>
Surrender
Card Type Check

Use Cases Refinement & Domain Analysis


Client
Server
First UI
Register
Check Register
[not passed]
[passed]
Update Database
Register Success
Login
Check Login
[not passed]
[passed]
Update Database
Logged In
Logout
Logout Confirm
Matiching Players
BeginGame
[not found]
[found]
Start Gameplay
Cancal Matching
Cancel Confirmation
Client
Server
GamePlay Display
Display Deal
Deal
active client
other client
Landlord Grab
Landlord Grab Decision
Display & Wait For Landlord
Check Choice
[Landlord not determined]
[Landlord determined]
Update GamePlay Info
Display Landlord
active client
other client
GiveHand
Give a Hand
Display & Wait For GiveHand
Check Card Type
[not passed]
[check passed]
Check Valid
[Game Not Ended]
[not passed]
Surrender
[passed]
Update Gameplay Info
Display This Hand
[Game Ended]
Display Result
EndGame