Design document

Group C

Jiaqi NI (2217774)

Jianfei Feng (2204529)

Kalina Petrova (2221667)

Zhang Wen (2351420)

Liu Menchuan (2151655)

Lin Jiefan (2617269)

Class: Ei7S

Mentor: Li Li

Version: first version

Contents

[Class diagram 3](#_Toc466829706)

[Class description 6](#_Toc466829707)

[Server side 6](#_Toc466829708)

[Card 6](#_Toc466829709)

[Account 6](#_Toc466829710)

[Game 6](#_Toc466829711)

[Player 6](#_Toc466829712)

[ValidCombination 7](#_Toc466829713)

[ValidComparison 7](#_Toc466829714)

[DBHelper 7](#_Toc466829715)

[CDoudizhuService 7](#_Toc466829716)

[IDoudizhuService 7](#_Toc466829717)

[Callbacks 8](#_Toc466829718)

[Client side 8](#_Toc466829719)

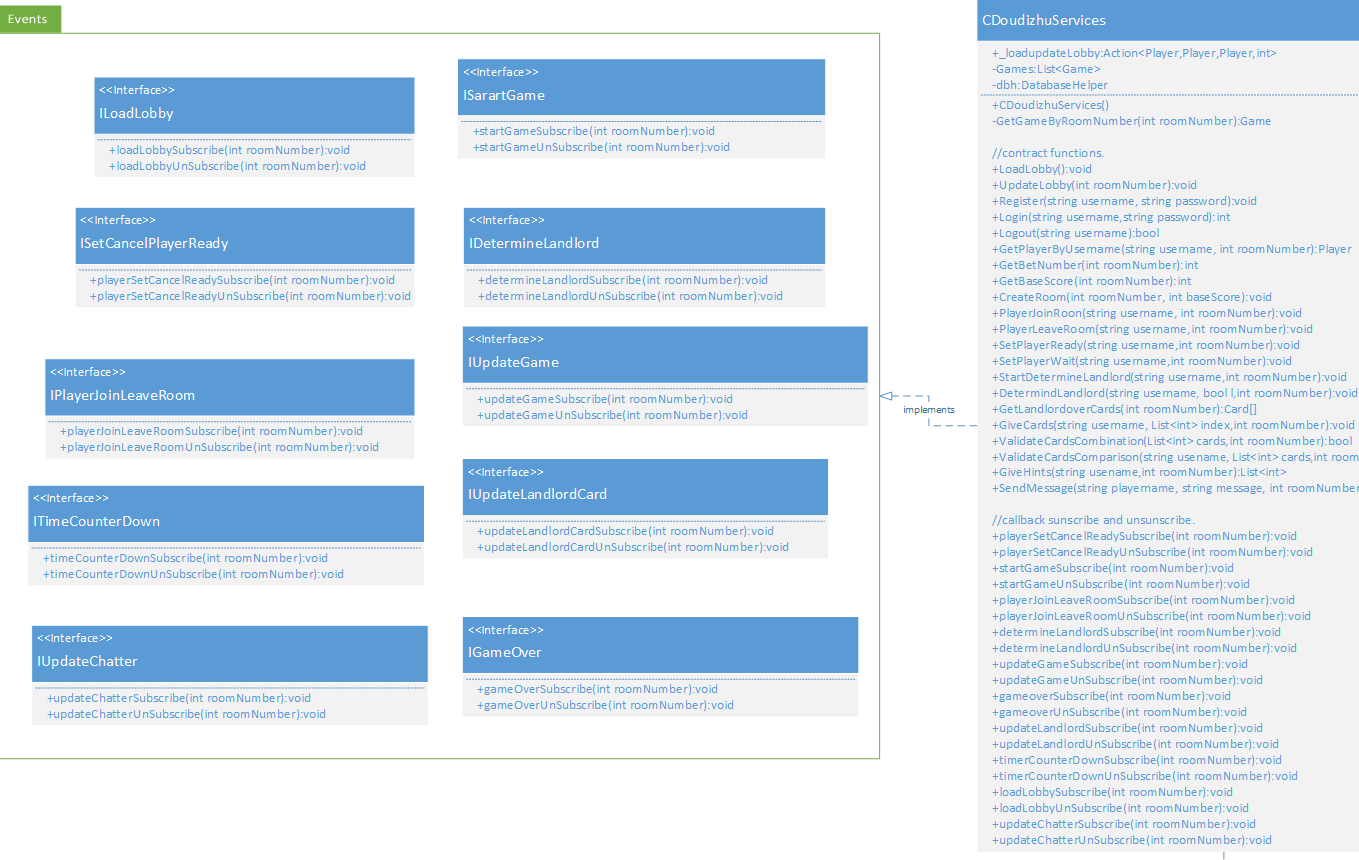
[Login 8](#_Toc466829720)

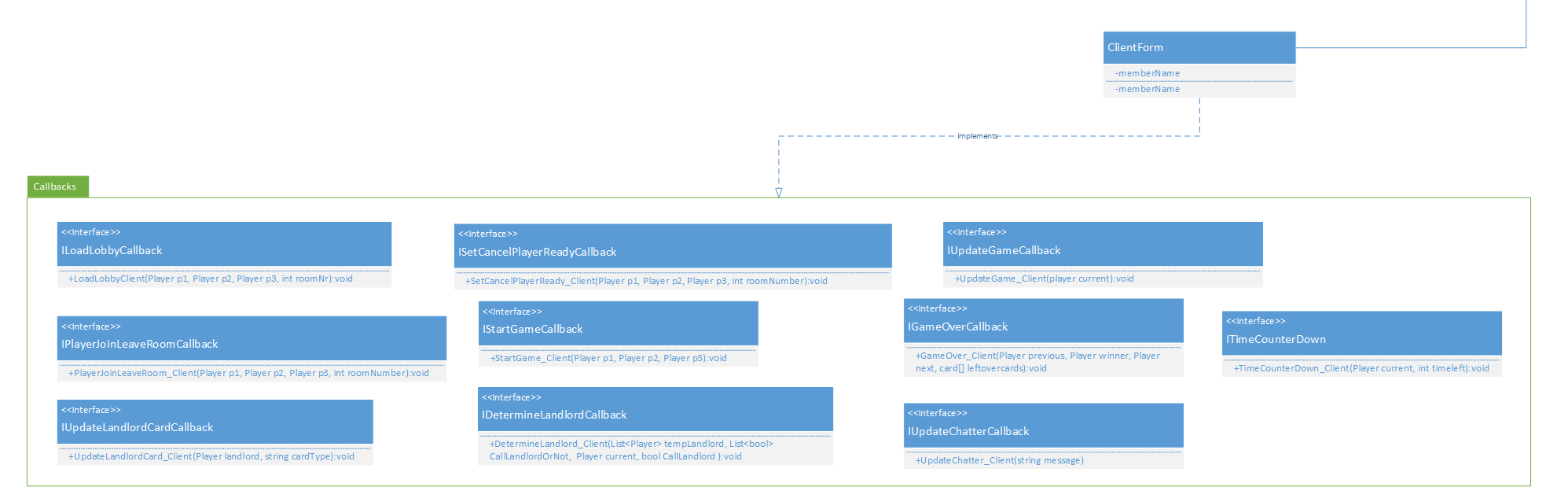
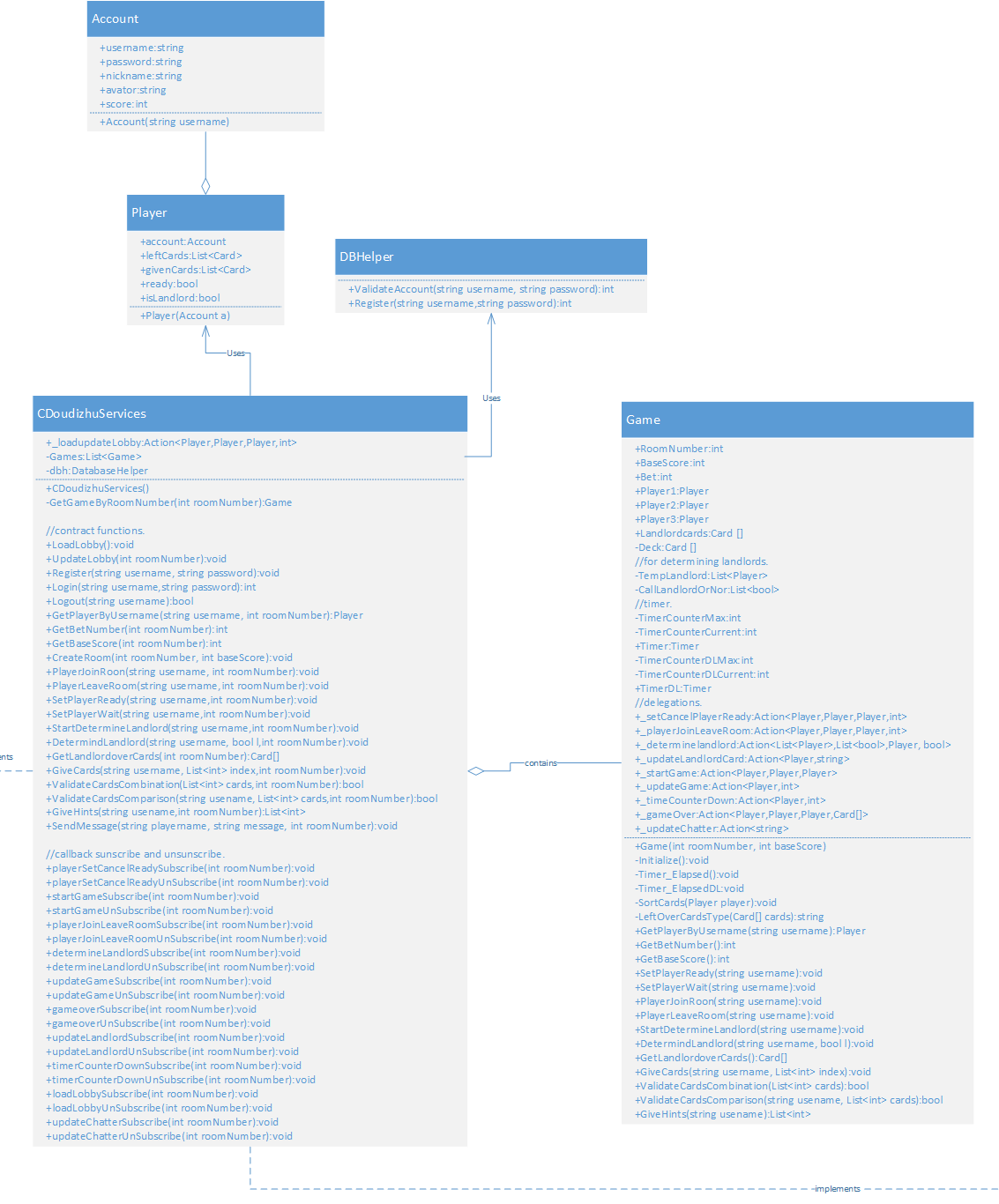
[Lobby 8](#_Toc466829721)

[Doudizhu 8](#_Toc466829722)

[User interface 8](#_Toc466829723)

# Class diagram





# Class description

## Server side

### Card

The card class represents the cards that are going to be played with inside the game. Card objects are represented with the following characteristics:

* Color – represents the suit of the card (Spade, Hearts, Clubs and Diamonds)
* Size – represents the number of the card (From 2 to Ace), but since we’re also playing with the Joker cards of a deck, we must include them as playable cards with numbers 15 and 16
* Image – each card is represented by an image inside the resources folder.

### Account

The account class holds information about the logged in user.

* Username – the username with which the user logs in.
* Nickname – the nickname that is shown to other players and with which user is identified in game.
* Password – the password used for login.
* Friends – a list of players that the user has added as friends through the game.
* PlayerImage – the image that the user has uploaded as his profile picture.

### Game

The game class specifies all the variables and actions that take place during a played match.

* RoomNumber – a unique integer that identifies the room.
* DuringGame – a Boolean that reveals whether the room is currently in game or not.
* BaseScore
* Bet
* Deck – A collection of Card objects that make up a whole deck.
* Landlordcards – a collection of cards that are supposed to be handed to the landlord player.
* Players – 3 Player objects that identify each of the participants in the room and one extra Player object that reveals who is the Player that currently needs to make a move.
* TempDizhu – auxiliary variable for determining the landlord.
* CallDizhuOrNot –auxiliary variable for determining the landlord.
* Timer – to make sure the game continues even without all the players active, a timer is implemented to limit each player’s turn.
* setCancelPlayerReady – a delegate to check whether a player is ready or not.
* playerJoinLeaveRoom – a delegate that checks who has joined and left the room.
* Determinelandlord – a delegate that determines the landlord for the round of the game.
* UpdateLandlordCard – updates the landlord’s hand.

### Player

This class represents the user while in game as part of the room he is currently in.

* Account – the account bound to this player on login.
* Left cards – a list of cards that are left in the player’s hand.
* LastCards – the last played hand by the player.
* Previous player – the player that is supposed to play before the player object.
* Next player – the player that is supposed to play after our player object.
* isFreeGiver – whether this player is free to give any cards, which means his last given cards combination cannot be beaten by other players.

### ValidCombination

This class has functions that validate the hand chosen by the player to be given on the table. Most of the functions are just Boolean functions that check whether the hand is one of the specified possible combinations.

### ValidComparison

### DBHelper

This class is purely devoted to connecting the server side to the database and communicate with it.

* Register – adds a new account to the database based on user input.
* Validate – checks login information for the user.
* Quit – changes user login status when he leaves.

### CDoudizhuService

The Service is the core of the server side as it implements all the functionalities inside the following interfaces:

* IDoudizhuService – responsible for main functionalities.
* ILoadLobby – Load all the players in the lobby, including their usernames and status (ready or not).
* ISetCancelPlayerReady – subscribe and unsubscribe methods for player ready functionality.
* IPlayerJoinLeaveRoom – subscribe and unsubscribe methods for joining and leaving of players.
* IStartGame – subscribe and unsubscribe methods for the game room.
* IDetermineLandlord – subscribe and unsubscribe methods for determining the landlord.
* IUpdateGame – update the current game, this will be called after some player gives cards.
* ITimeCounterDown – timer for calling landlord or giving cards. If a player fails to call landlord or give cards, he/she will be automatically passed.
* IGameOver – update game information. This will be called when a game is finished and will give the results to clients.
* IUpdateLandlordCard – update the landlords’ cards. When landlord is decided, he/she will gets 3 extra cards, this function is used to insert that three cards to the landlords’ hand cards.
* IUpdateChatter – for the chatting function. Update the chatting message.

### IDoudizhuService

The most important interface that takes care of the basic operations of the game.

* Login – logins the user with an existing account.
* Register – creates a new account on the database.
* Logout –logs the player out of the game.
* getPlayerByUsername – gets the player by his specified username.
* GetBetNumber – return the bet number of the game. (1, 2, 4, 8 …), if some player wins, he/she will earn bet number\*base score credits.
* GetBaseScore – return the base score of the game. (100, 200, 300 …), if some player wins, he/she will earn bet number\*base score credits.
* CreateRoom – creates a new room inside the lobby which can be accessed by the players.
* PlayerJoinRoom – allows a player to join a specified room by its number.
* PlayerLeaveRoom – allows the player to leave the room if he wishes.
* SetPlayerWait – Sets the player’s in-game status to waiting prior to the start of the game.
* SetPlayerReady – sets the player’s in-game status to ready prior to the start of the game.
* StartdetermineLandlord – determines the landlord for the current turn.
* DetermineLandLord – determines whether the player is landlord or not.
* GetLeftoverCards – returns a list of Card objects that need to be handed to the landlord after he has been determined.
* ValidateCardsCombination – validates whether the current card combination selected is a valid one.
* ValidateCardsComparison – validate whether the current card combination selected is big enough to beat the current card combination of the game.
* GiveHints – the computer will give the player a hint what he could play from his hand.
* GiveCards – hands over cards to players.
* SendMessage – allows the user to send an in-game message.
* PlayerQuit – allows the player to quit the game and removes him from the list of active online players.
* InGame – determines whether a room is currently playing or not.
* LastGivenCardAudio – this will return a string, which is the sound audio’s path. When some player makes some move, the program will give proper audio, and the server will use this function to find the correct audio.
* setCancelTuoguan – let the computer take over. (AI).

### Callbacks

There are a number of callbacks that are later implemented on the client side and return feedback back to the players depending on which events they have been subscribed to.

* ILoadLobyCallback – callback methods to update the lobby in the client side.
* ISetCancelPlayerReadyCallback –callback methods to update the game in the client side. When some player sets his/her status to ready, all players in that room will be informed.
* IPlayerJoinLeaveRoomCallback –callback methods to update the game in the client side. When some player leave/join a room, all players in that room will be informed.
* IStartGameCallback –callback methods to update the game in the client side. When all players in a room is ready, then game starts.
* ITimeCounterDownCallback –callback methods to update the game in the client side. This will be used in 2 cases: 1. determining the landlord. 2. A player is trying to give some cards.

Since these two function could take infinite time if we assume a player cannot make the decision or he/she intend to sabotage the game, the program will use a timer to prevent that.

* IUpdateChatterCallback –callback methods to update the game in the client side. When some player sends a message, all players in that room will receive that message.
* IUpdateGameCallback – callback methods to update the game in the client side. When some player gives some cards, all players in that room will see what he/she gives.
* IUpdateLandlordCardCallback – callback methods to update the game in the client side. When the landlord is determined, all players need to know that.
* IGameOverCallback – callback methods to update the game in the client side. When the game is over, all players will receive the result.

## Client side

### Login

The main form that opens once the application is ran. It will allow the user to login with an existing account and then redirect him to the main lobby. If the user still doesn’t have an account, he can choose to create one here.

This form implements the callbacks that are connected to the

### Lobby

The lobby is the form that is shown at the beginning when a user has logged in. In here the user can view all the current rooms he could join as well as check his information and converse with other online players.

The Lobby Form implements the ILobbyUpdateCallback.

### Doudizhu

This form concerns only the current game the user is playing. It will only be shown if a user is currently in-game.

The Doudizhu form implements most of the callbacks that concern the game.

## User interface