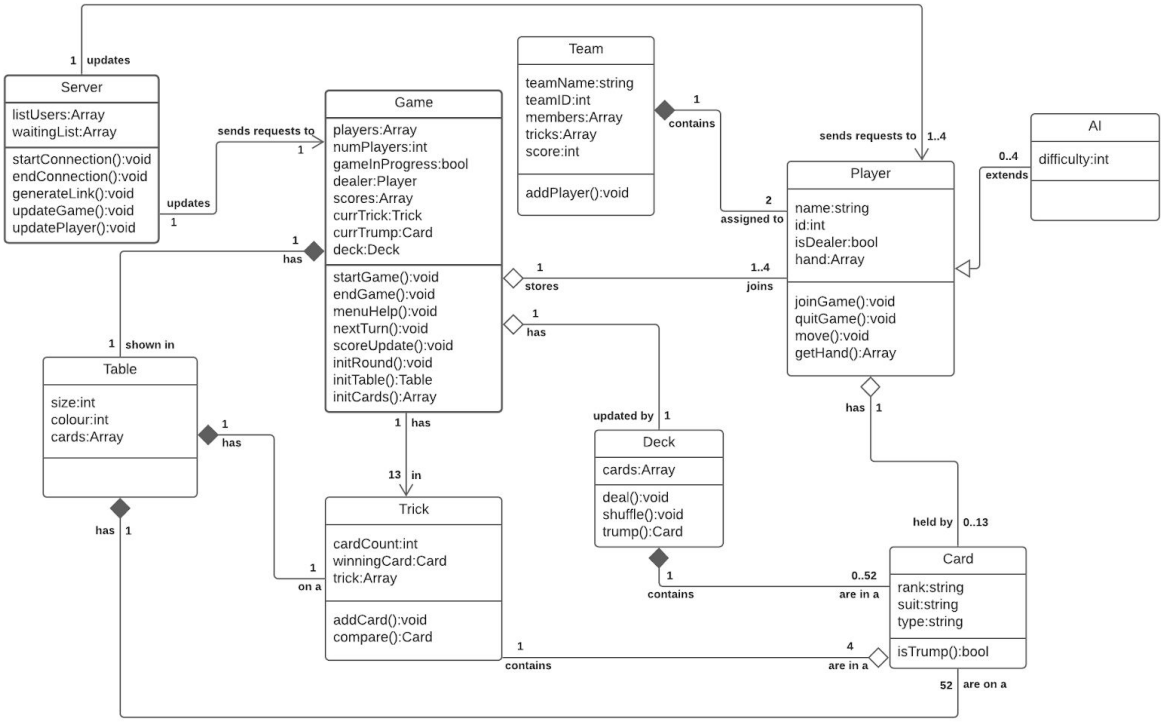
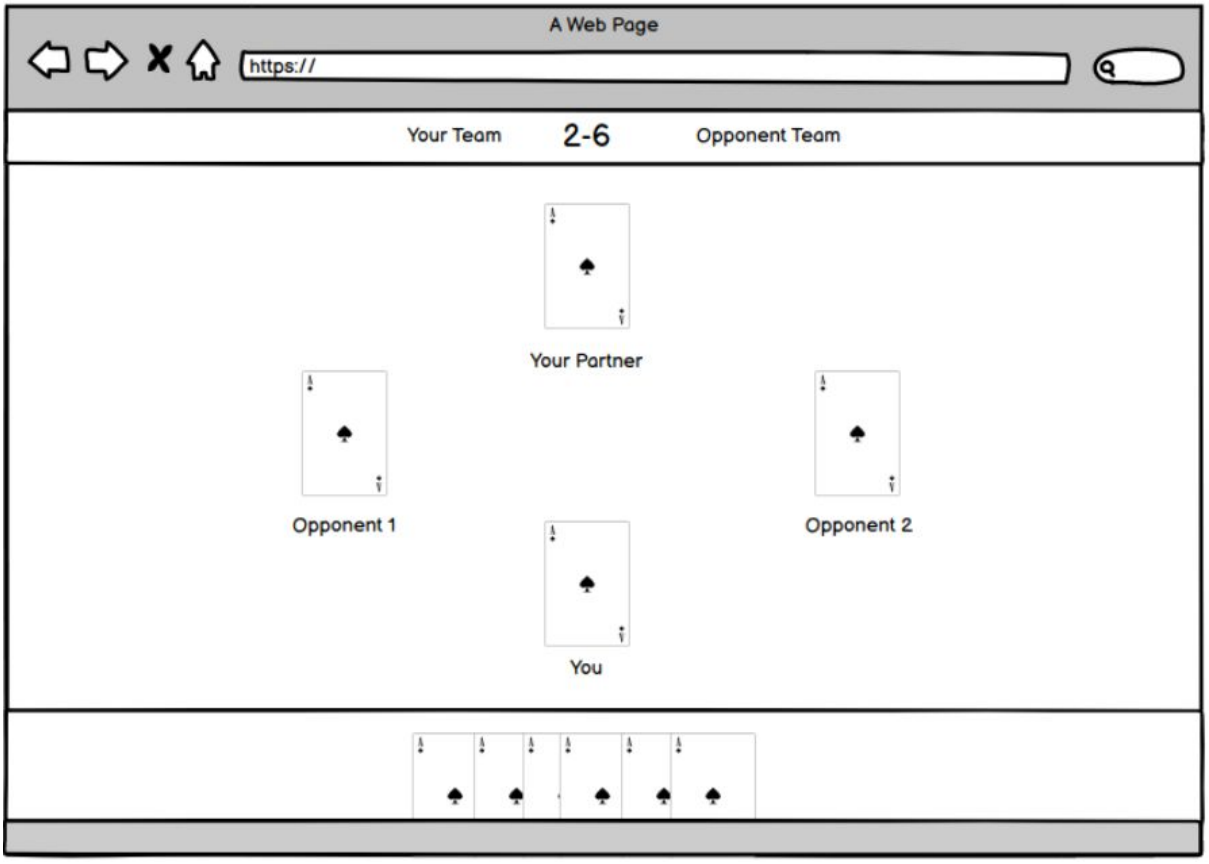
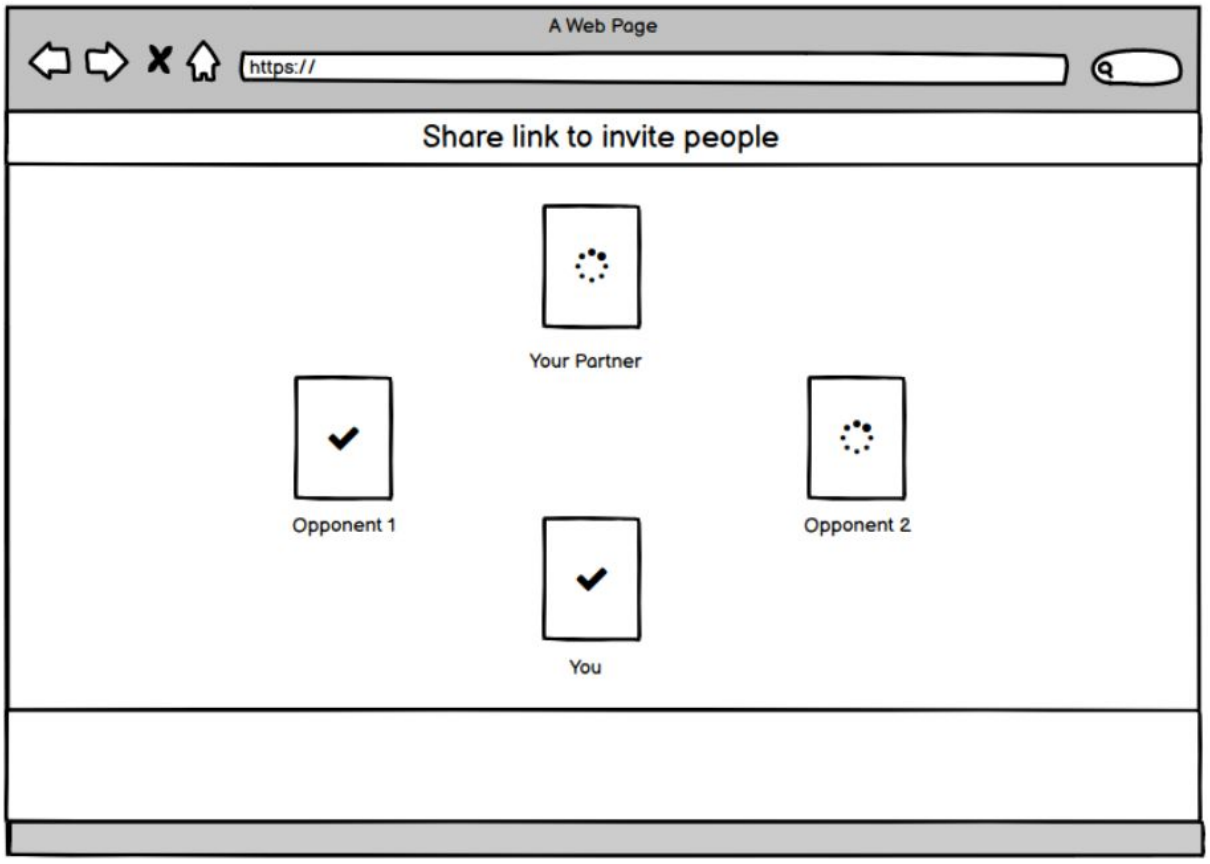
1. Refined class diagrams (These would be updates from your first Domain Class Diagram)



2. User interface mock-ups

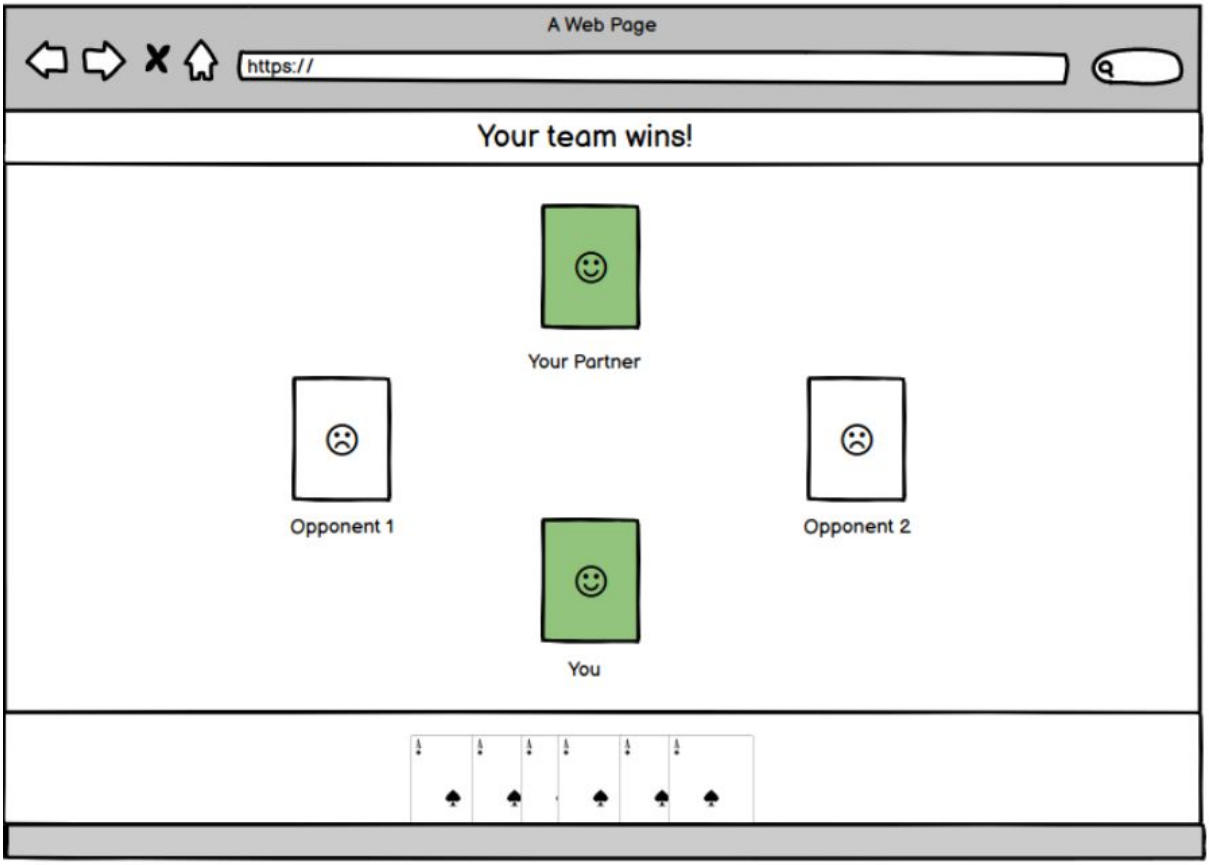


In the middle of a Whist game
Your hand is only visible to you and no one else. Your partner is clearly marked under the card, opposite to you. The score is displayed on the top navigation bar.



Invitation State

Players will need to share the link with others in order to play. The game starts once the server acknowledges there are 4 players in that game.



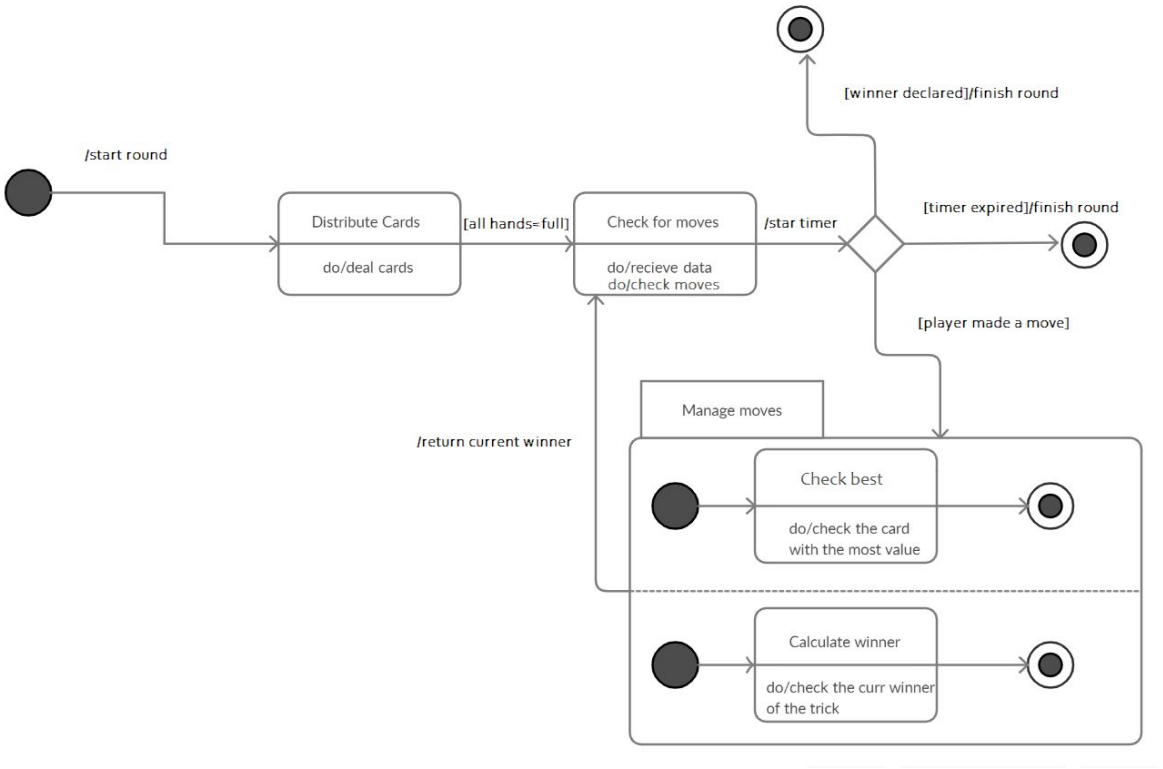
Win State

Winning team's card turns green with a smiley face on it while the losing team gets a frown on their card. If your team wins, the message 'Your team wins!' will be displayed on the top navigation bar, otherwise, it would display 'Your team lost'.

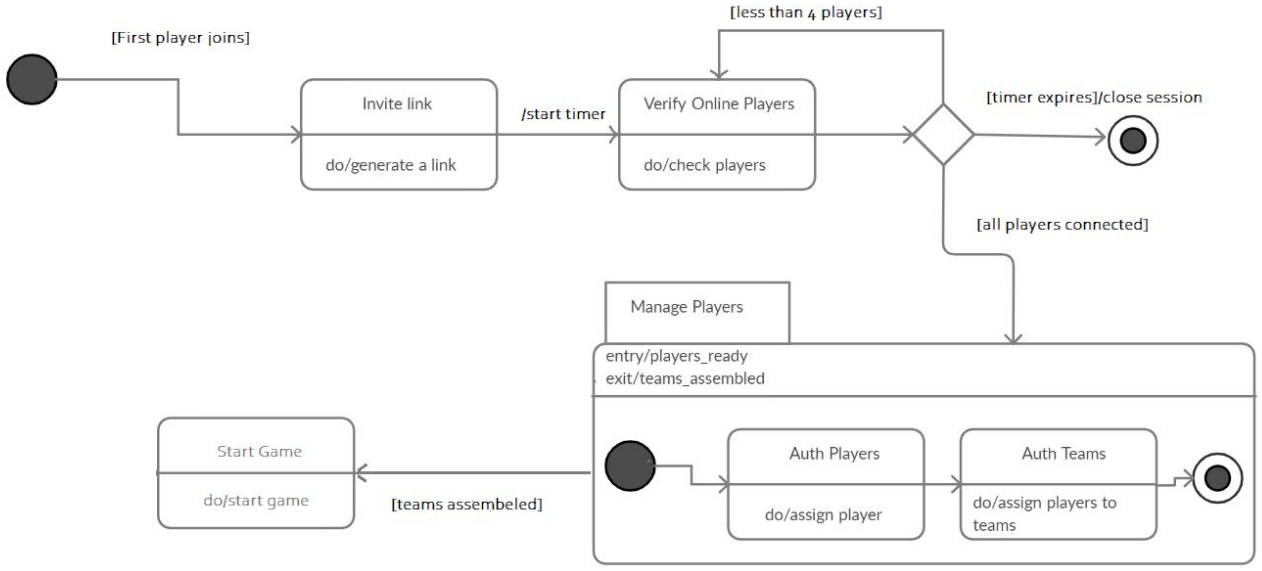
3. Client-Server experiments (Implementation ideas on your networking)

User Interface Mockup:
- We don't need the main menu
- Games can start whenever they visit the link (e.g. localhost:3000) and then redirect automatically to a new game e.g. localhost:3000/3eFgO1n
- AI is actually not so simple; there is strategy involved in the game
- Let's cut AI and just make it 4 player
- This makes implementation and interface simpler
- We can get all of the functionality done and done well in only one screen (only one design, only one web page required)

4. State machines. Pick Two e.g. One State diagram and a Sub-state diagram.



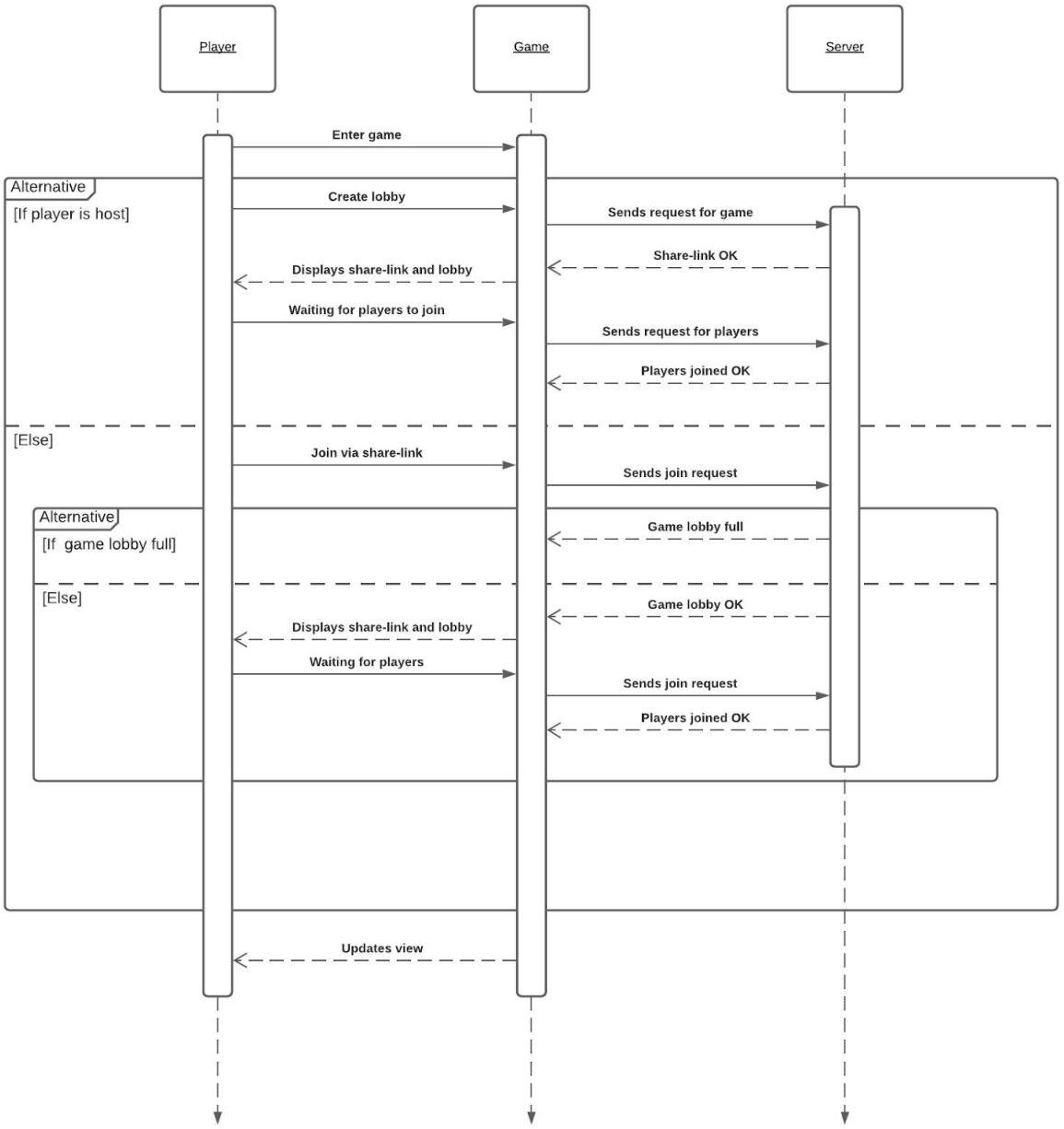
(State machine diagram describing the way rounds are played)



(State Machine diagram describing how players joining game get handled)

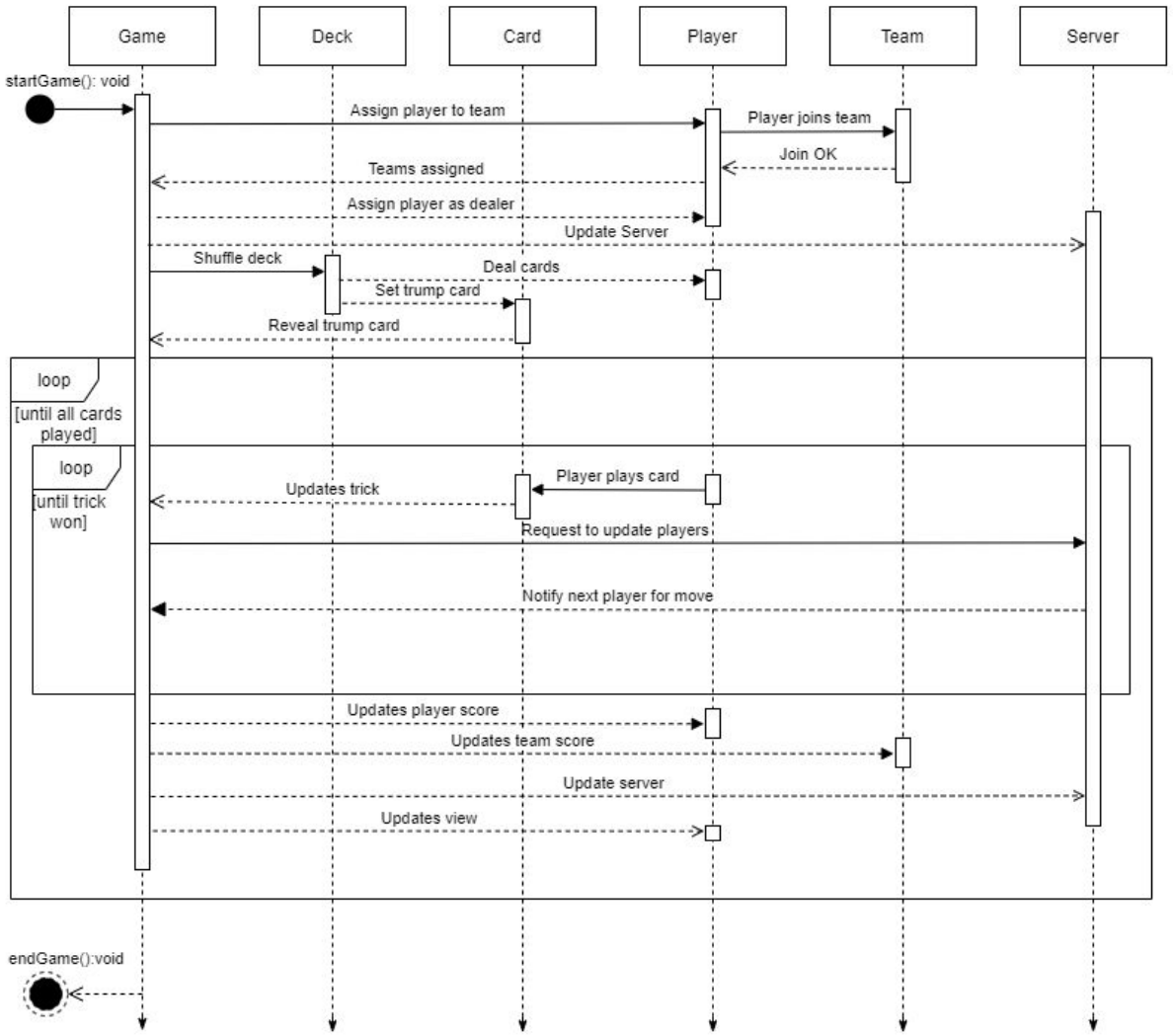
5. Sequence diagrams (Draw 4 based on the Use Cases - use UML 2.x format, IBM/Rational) For example One Sequence diagram with 3 separate sub-diagrams

User joins game:



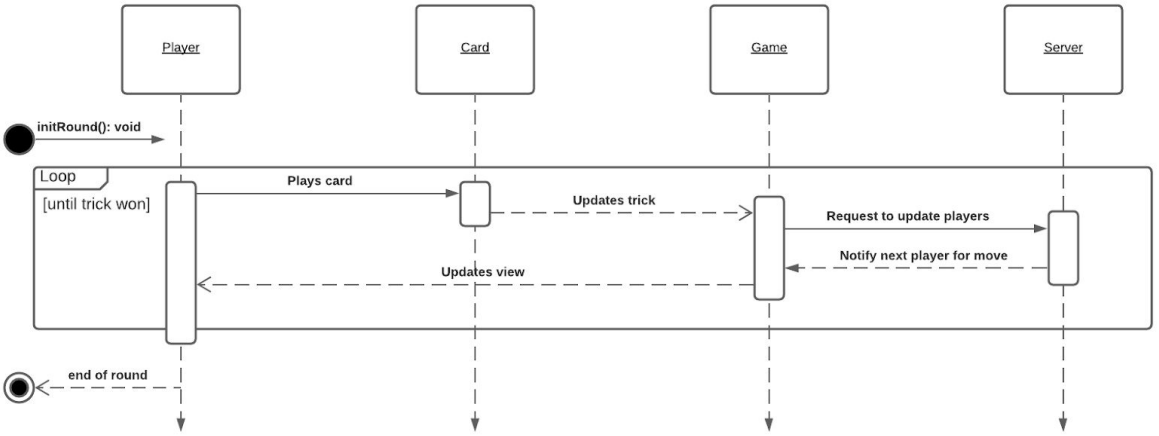
(Sequence diagram showing how users join/host a game server)

User plays game:



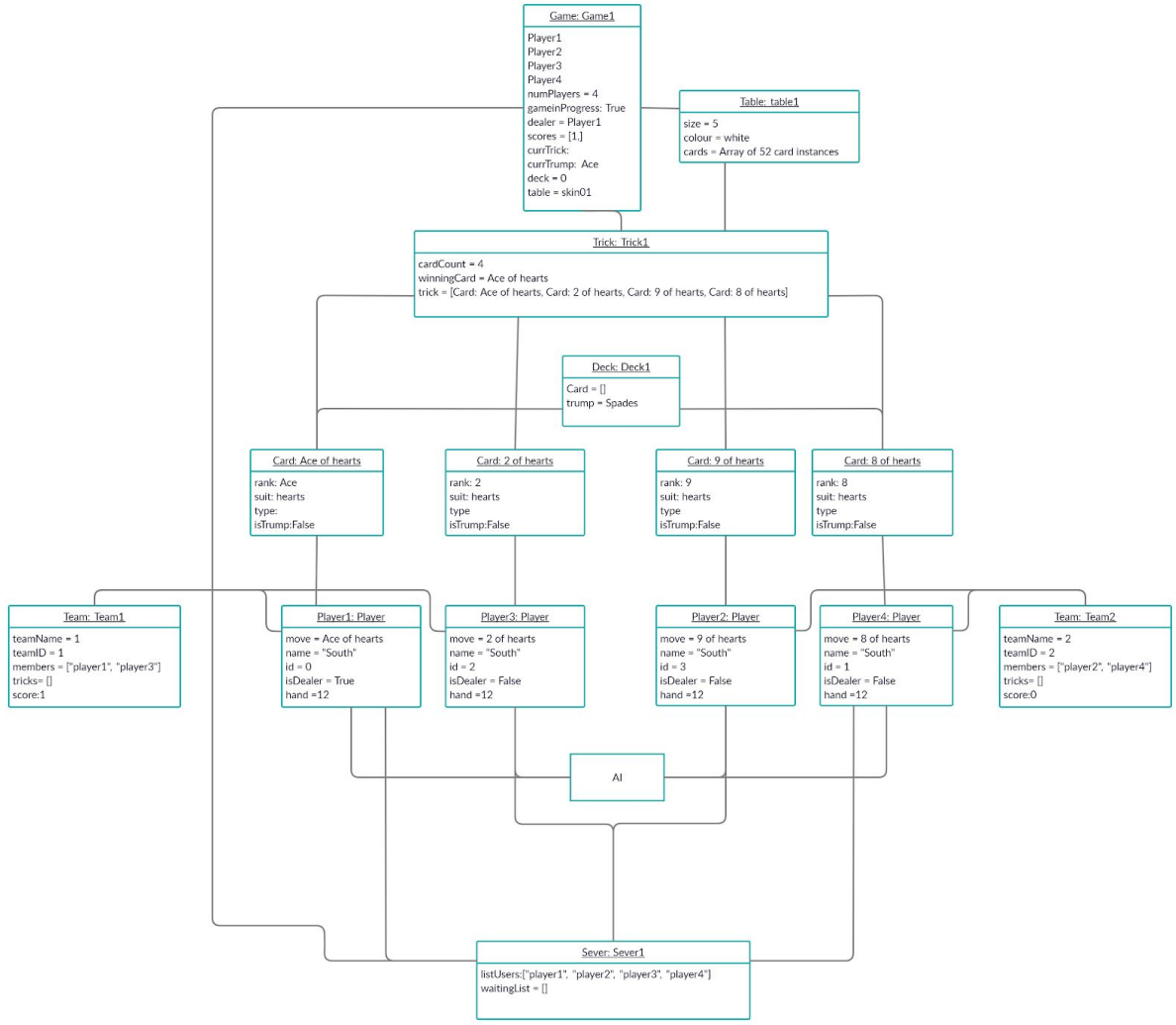
(Describes interactions among classes during the game)

User plays card:

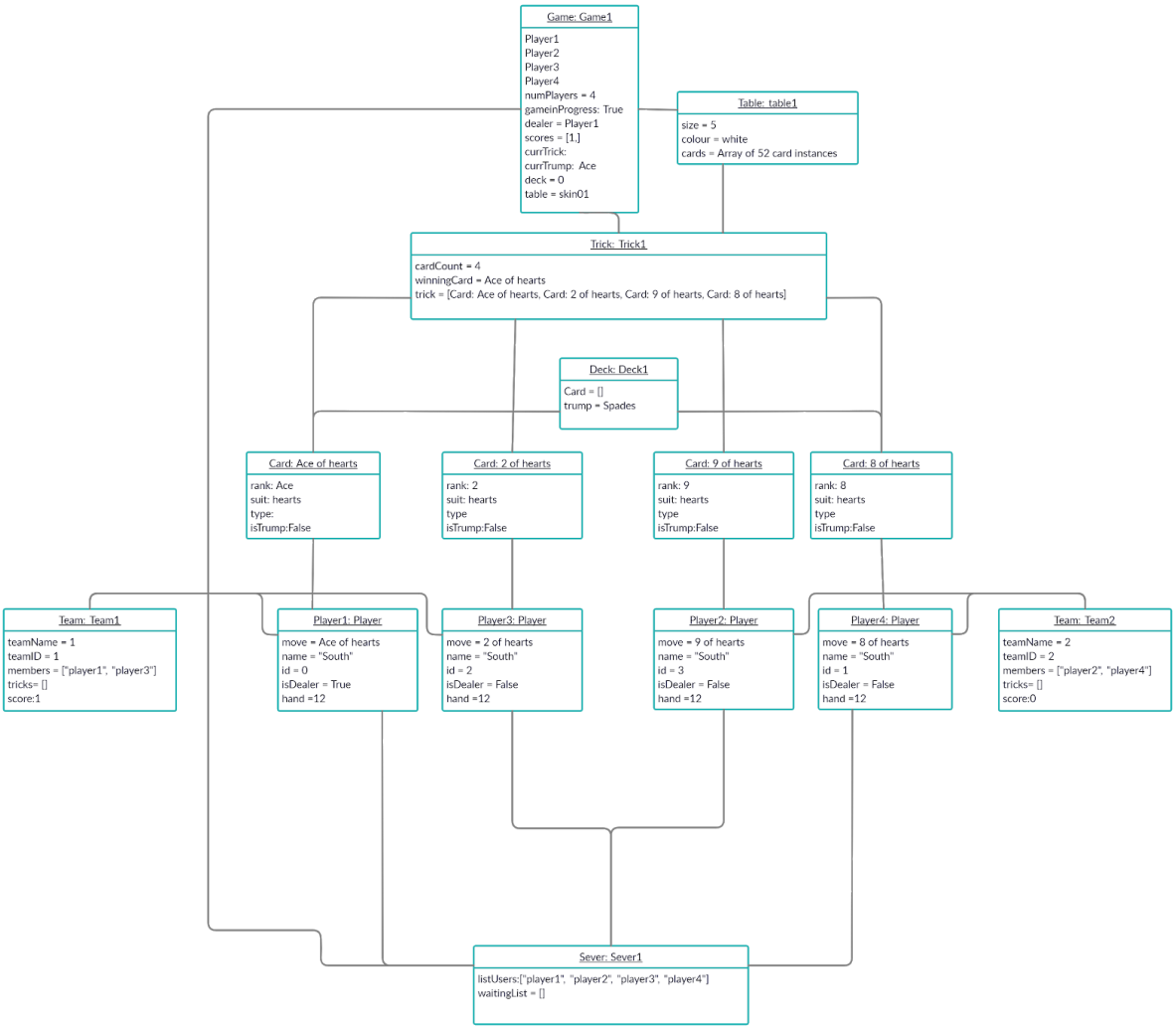


(Shows how a player plays a card per trick. Also shown in the “user plays game” sequence diagram.)

6. Object diagrams (Give examples of the part of the Class Diagram that seem important to the game) e.g. show how a sequence diagram is implemented with objects, in particular show attribute values.



(Object diagram with AI)

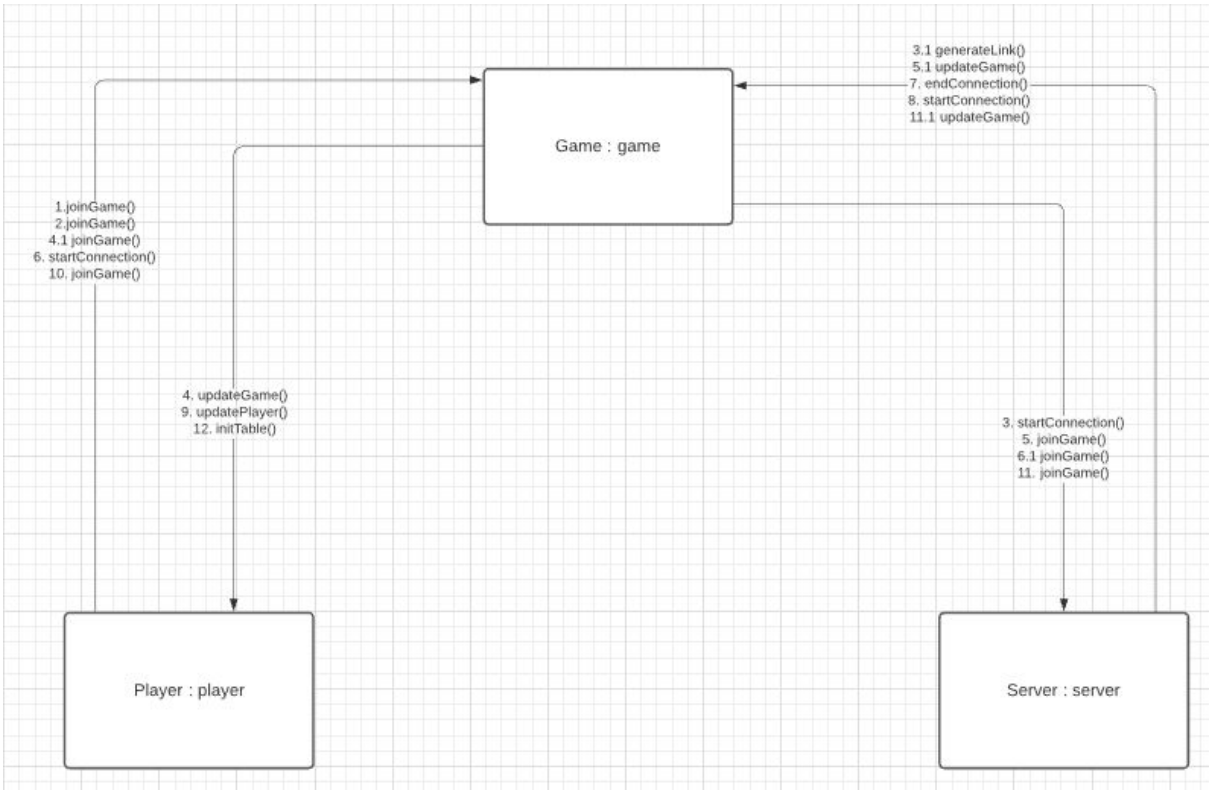


(Revised object diagram without AI)

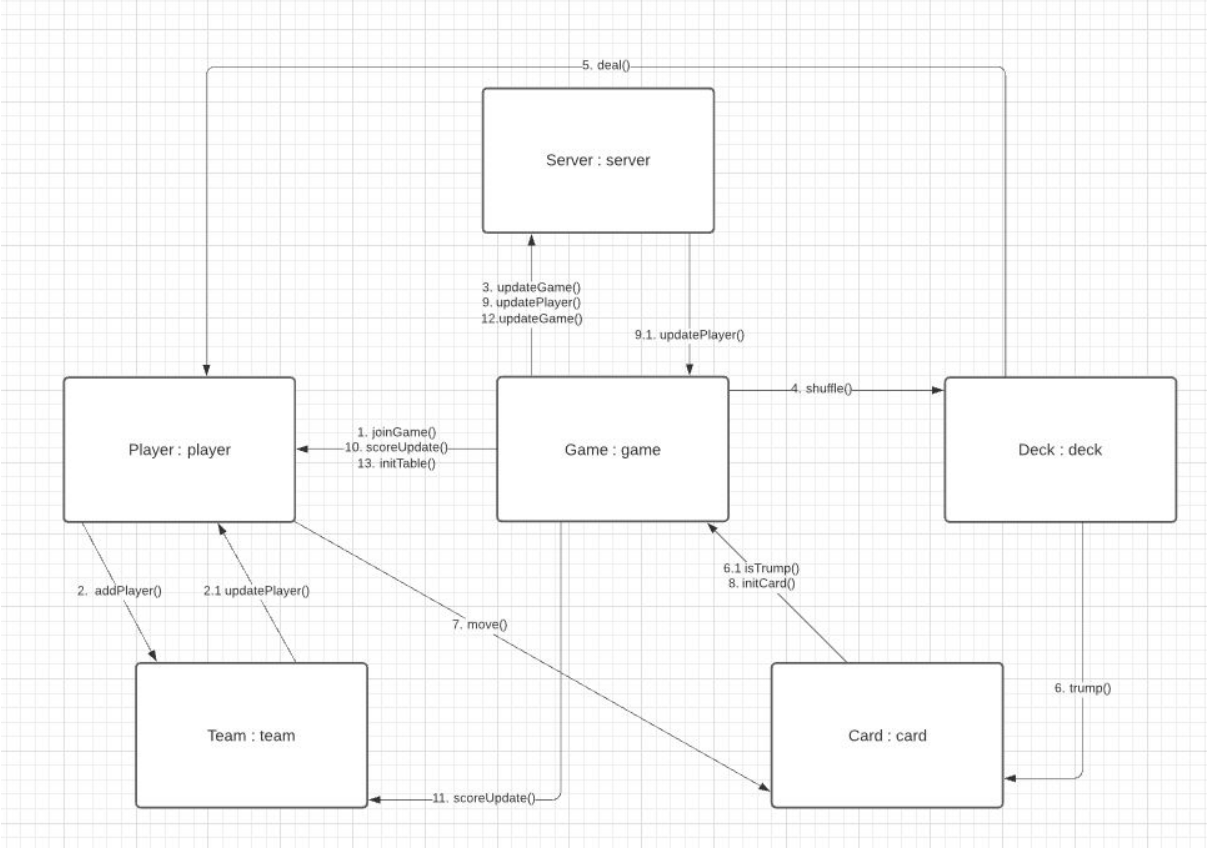
These object diagrams show the object interactions.
The scenario displayed is based on the first informal scenario.

7. Draw 4 Collaboration (now known as communication) based completely on your sequence diagrams in 5

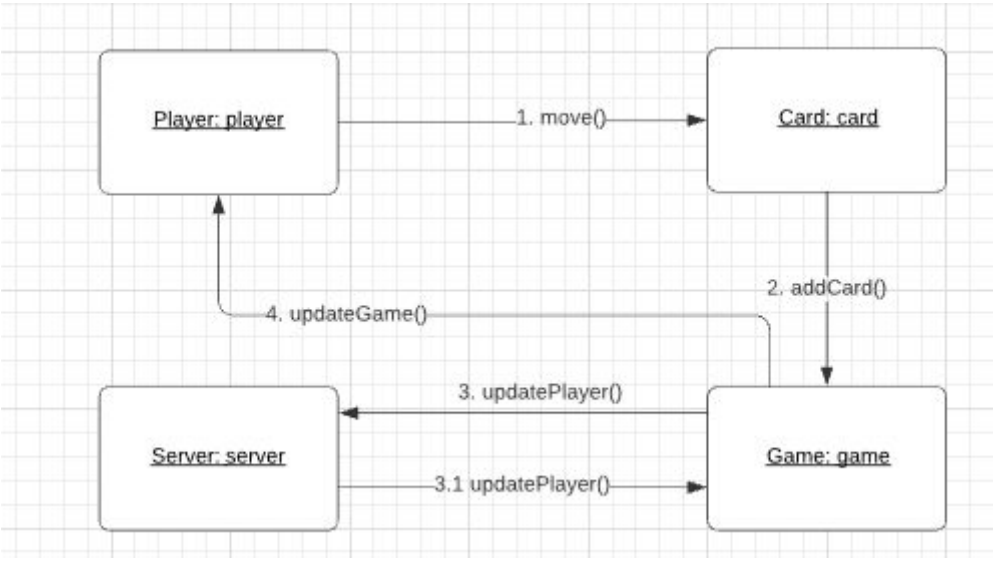
User joins game Collaboration (Communication) diagram:



User plays game Collaboration (Communication) diagram:



User plays card Collaboration (Communication) diagram:



8. Class skeletons from 8. (i.e. code prototypes, based on your actual implementation language)

Game class

```
class Game{
	[Player] players
	Int numPlayers
	Bool gameInProgress
	Player dealer
	[Int] scores
	Trick currTrick
	Card currTrump
	Deck deck

	startGame(){
		// starts game after players join and teams assigned
	}

	endGame(){
		// ends game after team wins game
	}

	menuHelp(){
		// displays a text-based menu for a quick tutorial of the game
	}

	nextTurn(){
		// assigns next player turn to play card
	}

	scoreUpdate{
		// updates the score of each player and team
	}

	initRound{
		// initialises the round for tricks to be played
	}

	initTable(){
		// updates the view of the game
	}
```

```
        initCards(){
                // initialises the cards for the current game once the cards are shuffled and
                // dealt
        }
}
```

Player class

```
class Player{
        String name
        Int ID
        Bool isDealer
        [Card] hand

        joinGame(){
                // allows player to join game through share-link
        }

        quitGame(){
                // allows player to quit game
        }

        move(){
                // player makes their move
        }

        getHand(){
                // player gets a new hand per trick/game
        }
}
```

Server class

```
class Server{
        [Array] listUsers
        [Array] waitingList


        startConnection(){
                // initialises game connection for players
        }
        endConnection(){
                // terminates game connection for players
        }

        generateLink(){
                // generates a share-link for players to access the online game
```

```
        }

        updateGame(){
                // provides updates to the current game
        }

        updatePlayer(){
                // updates players for any changes mid game
        }
}
```

Table class

```
class Table{
        Int size
        Int colour
        [Card] cards
}
```

Trick class

```
class Trick{
        Int cardCount
        Card winningCard
        [Card] trick

        addCard(){
                // appends cards played by players to the current trick
        }

        compare(){
                // compares the cards in current trick to find strongest card played
        }
}
```

Card class

```
class Card{
        String rank
        String suit
        String type

        isTrump(){
                // checks if the current card is trump card or trump suited card
        }
}
```

AI class

```
class AI{
	Int difficulty
}
```

Deck class

```
class Deck{
	[Cards] cards

	deal(){
		// deals the cards to players
	}

	shuffle(){
		// cards are shuffled in the deck
	}

	trump(){
		// sets random card as trump card
	}
}
```

Team class

```
class Team{
	String teamName
	Int teamID
	[Array] members
	[Card] tricks
	Int score

	addPlayer(){
		// game adds player to a team
	}
}
```

9. Minutes or notes of team meetings.

Date: 30th October 2020 Time: 9 - 9.40pm

Participants: Andrey, Alan, Gytis, Vincent, Melanie
Andrey talked about his progress with state machine diagrams. Asked for feedback on the diagram. The black circle signifies the start of the system while the black circle inside the white circle denotes it's the end of the system.

Sequence diagrams to be done first then collaboration diagrams.
All diagrams should be worked over a long period of time, as it is a crucial part of the project.

Diagrams to choose from:
sequence diagram, collaboration diagram and class skeletons.

Date: 2nd November 2020 Time: 9 - 9.15pm
Participants: Andrey, Alan, Gytis, Vincent, Melanie

Short meeting, just to check up on everyone's diagrams. Provide feedback and implement it where possible.

Date: 6th November 2020 Time: 9 - 9.50pm
Participants: Andrey, Alan, Gytis, Vincent, Melanie
We decided for the benefit of the team we should all be exposed to making the diagrams, The team decided that we should all work individually on two diagrams and pick the best one for the final document.

Going through diagrams.
Alan: class skeleton and collaboration diagram
Vincent: sequence diagram and class skeleton
Andrey: collaboration diagram and class skeleton
Melanie: collaboration diagram and sequence diagram
Gytis: collaboration and class skeleton diagram

Collaboration diagrams use function names
sequence diagrams use action words.

Goals for next Friday:
Try to finish the diagrams.

9th November Meeting cancelled due to assignments.

Date: 13th November 2020 Time: 9 - 9.20pm

Participants: Andrey, Alan, Gytis, Vincent, Melanie

Review class skeletons and collaboration diagrams
Next meeting will be Wednesday 18th November
Put all diagrams together in a document by Wednesday.
Next meeting on 20th November.

Date: 20th November 2020 Time: 9 - 11pm

Participants: Andrey, Alan, Gytis, Vincent, Melanie

We picked out the best diagrams the team made, we combined into one document and submitted it.