



Bilkent University

Department of Computer Engineering

---

# Object-Oriented Software Engineering Project

*monopoly-game: Digital version of the well-known board game Monopoly*

## Analysis Report

Group 1C: Ziya Mukhtarov, Ege Kaan Gürkan, Alper Sarı, Mokhlaroyim Raupova, Javid Baghirov

Instructor: Eray Tüzün

Teaching Assistant(s): Barış Ardiç, Emre Sülün, Elgun Jabrayilzade

Iteration 2

December 10, 2020

This report is submitted to the Department of Computer Engineering of Bilkent University in partial fulfillment of the requirements of the Object-Oriented Software Engineering course CS319.

## Table of Contents

1	Introduction.....	5
2	Overview.....	5
2.1	Changes from the Original Board Game .....	5
2.2	Initializing a Game .....	5
2.3	Gameplay .....	6
2.4	Board.....	6
2.4.1	Properties .....	6
2.4.1.1	Street.....	6
2.4.1.2	Railroad .....	6
2.4.1.3	Utility.....	6
2.4.2	Action Space.....	6
2.4.2.1	Go.....	7
2.4.2.2	Chance / Community Chest .....	7
2.4.2.3	Income Tax / Luxury Tax .....	7
2.4.2.4	Free Parking .....	7
2.4.2.5	Go to Jail .....	7
2.4.2.6	Jail / Just Visiting .....	7
2.5	Auctions.....	7
2.6	Getting Out of Jail .....	7
2.7	Building Houses.....	7
2.8	Building Hotels.....	8
2.9	Deals & Trades .....	8
2.10	Mortgaging.....	8
2.11	Player Out.....	9
2.11.1	Owing to Another Player .....	9
2.11.2	Owing to the Bank .....	9
3	Functional requirements.....	9
3.1	Menu Functionality.....	9
3.1.1	Start/Configure a New Lobby .....	9
3.1.2	Search for a Lobby .....	9
3.1.3	Enter a Lobby .....	9
3.1.4	Leave Game/Lobby.....	9
3.1.5	Change In-Game Options .....	10
3.1.6	Adjust Game Settings .....	10

3.1.7	View Game Guide.....	10
3.2	Gameplay Functionality .....	10
3.2.1	Roll Dice .....	10
3.2.2	Manipulate a Tile .....	10
3.2.3	See Game State .....	10
3.2.4	Create/Accept a Trade Offer .....	10
3.2.5	Vote to Kick Player.....	10
3.2.6	Chat via Text, Voice, and/or Video.....	10
3.2.7	Participate in an Ongoing Auction.....	10
3.2.8	Progress through the Game.....	10
3.2.9	Mortgage / Unmortgage .....	10
3.2.10	Go Bankrupt .....	11
3.2.11	Win .....	11
4	Nonfunctional requirements .....	11
4.1	Usability .....	11
4.2	Reliability .....	11
4.3	Performance .....	11
4.4	Supportability .....	11
5	Pseudo Requirements.....	11
6	System models .....	12
6.1	Use Case Model .....	12
6.1.1	Learn How to Play .....	12
6.1.2	Change Settings .....	13
6.1.3	Join Lobby .....	13
6.1.4	Create Lobby .....	13
6.1.5	Set Readiness to Play.....	14
6.1.6	Auction.....	14
6.1.7	Mortgage .....	14
6.1.8	Trade .....	15
6.1.9	Buy House .....	15
6.1.10	Buy Hotel.....	16
6.1.11	Buy Property.....	16
6.2	Dynamic models .....	17
6.2.1	Sequence Diagrams .....	17
6.2.1.1	Lobby Creation Sequence Diagram .....	17
6.2.1.2	Lobby Join Sequence Diagram .....	18

6.2.1.3	Trade Sequence Diagram .....	19
6.2.1.4	Auction Sequence Diagram .....	20
6.2.1.5	Bankruptcy Sequence Diagram .....	21
6.2.2	State Diagrams .....	22
6.2.3	Activity Diagrams .....	22
6.2.3.1	Lobby Activity Diagram .....	22
6.2.3.2	Gameplay Activity Diagram.....	23
6.3	Object and Class Models .....	25
6.3.1	Lobby Class .....	26
6.3.2	User Class .....	26
6.3.3	GamePlayer Class .....	26
6.3.4	Game Class .....	26
6.3.5	Trade and Auction Classes .....	26
6.3.6	Enumerations and Interfaces .....	26
6.3.7	Tiles, Properties, Buildings, and Extras.....	27
6.4	User Interface .....	28
6.4.1	Navigational Path.....	28
6.4.2	Menu .....	28
6.4.3	Create a Game .....	29
6.4.4	Lobby: Owner View.....	29
6.4.5	Join Game .....	30
6.4.6	Lobby: Player View .....	31
6.4.7	Settings.....	31
6.4.8	Main Game View .....	32
6.4.9	Text Chat .....	33
6.4.10	Popups .....	33
6.4.10.1	Auction .....	33
6.4.10.2	Trade .....	34
6.4.10.3	Landing on a Property.....	35
6.4.10.4	Miscellaneous.....	35
7	Improvement Summary.....	36
8	Glossary & references.....	36

# 1 Introduction

One of the most popular board games of history is the Monopoly game. First published in 1935, it still holds its popularity to this day. As a group, we decided to digitalize this game since we all are lovers of Monopoly. We think that the Monopoly game has the potential to be a really good object-oriented project and fun to work on. We are even more excited to test this game by playing it after we complete the implementation phase.

We aim to create digital game software for the Monopoly Classic version [\[1\]](#). In the original board game, the players would take turns and could buy and sell properties, manage their money, win chance cards, and have fun. Each player rolls the dice at the beginning of their turn and plays accordingly to the number obtained by the dice roll, which makes the game dependent on luck. The board game has quite a few interesting rules; for example, if a player gets a double dice roll 3 times in a row, he/she is forced to go to the jail block. More on these rules are given later in this document.

Since we are digitalizing the board game, we decided to add many features that would otherwise be impossible to achieve. For instance, the gameplay will be held in an online environment where players get to play with their friends or random people all over the world without the requirement of being in the same room and risking their safety, especially during the quarantine. Moreover, the players can socialize and make new friends by using the video and voice chat feature.

In this document, the detailed analysis of the software that we are going to build is given. Firstly, all the rules of the game are explained. Then, the functional and non-functional requirements for our game are provided. We intend to comply with all of these requirements. Later in the document, the detailed UML diagrams and UI mockups are shown. We will closely follow these during the implementation phase.

## 2 Overview

### 2.1 Changes from the Original Board Game

The main change in our interpretation of Monopoly compared to the physical board game is the fact that it can be played while using an in-built video and voice communication system, where the players will be able to speak directly with one another and use their cameras if they wish so. We also added a text chat system which can be used whenever a player decides not to communicate using their microphones. Since the board game is played in a single room with other people, we wished to encapsulate that same experience without the requirement of being physically present.

### 2.2 Initializing a Game

In our software, the games will be created using lobbies. Lobbies can either be public or private. Each lobby will have an owner, the player who initially created the lobby. The ownership can be passed to another player. Private lobbies will require a passcode to enter, specified by the owner. Additionally, the lobbies will have a name that can be used to search for. The maximum number of players in a lobby is limited to 6. The players can mark

themselves as ready to play. The game will start when all the players are ready and there is more than one player in the lobby.

## **2.3 Gameplay**

Each player is represented with a figurine on the board. The game ends when all the players except the winner have gone bankrupt. The game contains a board, a bank (handled by computer), player figurines, money (denoted as ~~AA~~) banknotes, Chance cards, Community Chest cards, Title Deed cards, houses, and hotels. Initially, all players are given ~~AA~~1500 from the Bank and their tokens are placed on the Go space.

The players will move on the board according to a dice roll. After a dice roll, the player's token is moved to the number of spaces shown the dice in a clockwise direction. If the dice roll is a double, the player is to take another turn and roll again. If the player rolls a double 3 times in a row, he/she is sent to jail.

The first player to take a turn is determined using a dice roll. Any ties are broken with a reroll. The highest roller is to go first. After that, the players take turns in a clockwise direction. The player can only alter the game if it is their turn.

## **2.4 Board**

The game board is square-shaped and each side contains 9 spaces except the corners that a player can land on. There are 40 spaces in total. Each space can be one of the following types:

### **2.4.1 Properties**

Each property has an associated Title Deed card. If the property is not owned by anyone, a player that lands here can either buy it by paying the price shown on the board to own it or auction it. If the property is already owned, the player landing on that property's space must pay the rent shown on the corresponding Title Deed card. There are 3 types of properties.

#### **2.4.1.1 Street**

There are 22 street spaces divided into 8 sets denoted by color. The color set that a street belongs to is shown on the board. Note that the rent increases as the owner builds more houses, hotels, and/or owns the complete color set.

#### **2.4.1.2 Railroad**

There are 4 railroad spaces. The rent for Rails increases as the owner owns more railroads.

#### **2.4.1.3 Utility**

There are 2 utility spaces. The rent for Utilities increases as the owner owns more utilities and it is dependent on a dice roll.

### **2.4.2 Action Space**

The rest of the spaces are action spaces. Each of these spaces has a unique rule shown below.

#### 2.4.2.1 Go

Whenever the player passes this space, they gain ~~AA~~200 from the bank unless stated otherwise.

#### 2.4.2.2 Chance / Community Chest

If a player lands on this space, they immediately complete the actions from the topmost card from the corresponding pile, and then the card is returned to the bottom of the pile.

#### 2.4.2.3 Income Tax / Luxury Tax

The player must pay the amount shown on the board to the bank.

#### 2.4.2.4 Free Parking

Nothing happens when the player lands here.

#### 2.4.2.5 Go to Jail

The player's figurine is moved to the Jail space immediately. If the Go space is passed, no money is gained. Then, the player's turn is over.

#### 2.4.2.6 Jail / Just Visiting

This is the Jail space. If the player lands here after their usual move, they are to stay at just visiting space and do nothing special.

### 2.5 Auctions

When a player lands on an ownable space that is not owned by anyone, they can decide to auction it. Auctions will occur when a player goes bankrupt or a player wants to get the last house/hotel from the bank as well. The base cost of the auction is ~~AA~~10. Any player, including the player who started the auction, can increase the bid by any amount they want. The highest bidder will win the auction and take the corresponding Title Deed card. If no one bids during the auction, no one owns the space and no money is paid.

### 2.6 Getting Out of Jail

The player that is in jail has 3 options to get out. The first option is to pay ~~AA~~50 at the start of their next turn and move as normal. The second option is to use a special card called **Get Out of Jail Free** at the start of their next turn if they own one, return the card to the pile and move normally. The third option is to try their luck and roll a double using the dice. The player is given 3 turns for trying to roll a double. Note that if the player does not roll a double, his/her turn is skipped. If the double is rolled, the player is free and moves according to the roll. Note that the player does not get any bonus turns that they would get if they rolled a double as their normal turn. After the 3<sup>rd</sup> turn trying for double and failing, the player must pay ~~AA~~50 and move according to his/her last roll.

### 2.7 Building Houses

A player can only build houses on a street if the whole color set is owned by that player and none of the properties from that color set is mortgaged. The cost of a house is written on the Title Deed card of the street. Another restriction on building a house is that the houses in a color set should be increased equally. In other words, a player cannot build a house on a street

if any of the streets from that color set has a smaller number of houses. The maximum number of houses on a street is limited to 4.

There is a total of 32 houses available for all players. If all 32 houses are built by players, then the next player must wait until a house is available to be able to build a house in their street. If a player wants to build a house and the house is the last one available in the Bank, the house is auctioned.

## **2.8 Building Hotels**

A player can upgrade from the houses to a hotel if he/she has built 4 houses on a street. The price of a hotel is shown on the street's Title Deed card. To be able to build a hotel on a street, all the streets from the same color set must have either 4 houses or a hotel. There can only be at most one hotel in a street. Note that if a player wants to build a hotel, the 4 houses must be returned to the bank.

There is a total of 12 hotels available for all players. If all 12 hotels are built by players, then the next player must wait until a hotel is available to be able to upgrade from 4 houses to a hotel in their street. If a player wants to build a hotel and the hotel is the last one available in the Bank, the hotel is auctioned.

## **2.9 Deals & Trades**

The players are free to buy, sell, or swap any of their property with other players upon agreement. Properties can be traded for cash, other property, and/or a **Get Out of Jail Free** card.

To be able to sell a property, all buildings on the color set must be sold to the Bank. If a mortgaged property is sold, the buyer has two options and has to decide immediately: either repay the mortgage and unmortgage the property, or keep the mortgage by paying 10% of the mortgage value to the Bank.

The player can sell their buildings to the Bank only. Hotels are sold for their half price and 4 houses. Houses are sold at half price. The number of hotels and houses on a color set must stay evenly distributed as the deals and trades happen. If a player wants to sell a hotel, but the bank does not have 4 houses, then the player must keep selling the houses and arranging them so that their count is evenly distributed in the end. As an example, let's say that a player owns a color set with 3 streets and has 4 houses on 2 streets and a hotel on one street. Assume that the bank has only one house. If the player wants to sell the hotel, he/she must sell the hotel and 1 house from each of the remaining streets on the color set. The player will end up having 3 houses on each of the streets.

## **2.10 Mortgaging**

Mortgaging a property is an action a player can take to increase their money. To be able to mortgage a property, all the buildings that are built on the same color set must be sold to the Bank at half their cost price. Once mortgaged, the corresponding Title Deed card is flipped over, and the mortgage value written on it is paid to the player by the bank.



The mortgaged properties can later be unmortgaged. To do so, the player must pay the value shown on the Title Deed card to the Bank. After that, the card is flipped again so that it is right face up.

Once the property is mortgaged, the player cannot collect rent from the player that lands on that property. Note that the benefits of owing full color set for streets, or increased rent for owning more Utilities and Railroads persist.

## **2.11 Player Out**

If the player is in debt and cannot earn enough money by deals, trades, and mortgaging, the player loses and is out of the game. The following subsections explain what happens depending on whom the player owes money to.

### **2.11.1 Owing to Another Player**

If the player owes to another player, first all the buildings the player owned is sold to the bank at their half price. Then, everything the player owned, including his properties, mortgages, and money is given to the player they owed. Mortgaged properties must be immediately unmortgaged or kept mortgaged according to the rules explained in the Deals & Trades section.

### **2.11.2 Owing to the Bank**

In the case of owing to the Bank, all the player's money is returned to the Bank. All mortgaged properties are canceled. All **Get Out of Jail Free** cards are returned to the bottom of the pile. Then, all of the player's properties are auctioned to the remaining players.

## **3 Functional requirements**

### **3.1 Menu Functionality**

#### **3.1.1 Start/Configure a New Lobby**

The player can choose to start a lobby for playing with other players, which can be private or public. The owner of the lobby can configure its settings such as its name, public/private type, and passcode.

#### **3.1.2 Search for a Lobby**

The users should be able to search for a lobby using its name.

#### **3.1.3 Enter a Lobby**

When trying to play a game, the players will have the choice of entering a public lobby or entering a lobby code to access a private lobby.

#### **3.1.4 Leave Game/Lobby**

The player will have the option of leaving a lobby before a game has started, or leaving an ongoing game.

### **3.1.5 Change In-Game Options**

The player can modify some in-game options regarding sound levels, text chat, and voice chat.

### **3.1.6 Adjust Game Settings**

The player can modify game settings outside of a match. These settings include webcam and microphone device selection, public username, and game sound level.

### **3.1.7 View Game Guide**

The player will be able to see the game's ruleset.

## **3.2 Gameplay Functionality**

### **3.2.1 Roll Dice**

During the player's turn, they will be able to roll dice and determine their move.

### **3.2.2 Manipulate a Tile**

After the player's move, they will have several options depending on the tile's type and condition. As an example, they will be able to buy a property if the property is not owned by anyone.

### **3.2.3 See Game State**

The player can see information about the current game state at any time during a game. This information includes property ownership, the Bank's condition, player token locations, money amount.

### **3.2.4 Create/Accept a Trade Offer**

The player can pick another player to make an offer to on their turn. Both parties can add anything they own except for buildings and they can accept or cancel the trade.

### **3.2.5 Vote to Kick Player**

During an ongoing game, a user can vote for another user to be kicked. If the majority votes to kick a player, the player will be kicked.

### **3.2.6 Chat via Text, Voice, and/or Video**

The players can access the text chat and open their microphones and webcams anytime they want.

### **3.2.7 Participate in an Ongoing Auction**

The player can increase the bid during an auction and win if s/he is the highest bidder.

### **3.2.8 Progress through the Game**

The player can buy streets, utilities, build houses/hotels, and make progress.

### **3.2.9 Mortgage / Unmortgage**

The player can mortgage a property s/he owns, and get some money from the bank. Later, s/he can unmortgage it with some additional cost, and get it back.

### **3.2.10 Go Bankrupt**

The player can go bankrupt during the game by running out of money.

### **3.2.11 Win**

The player can win the game by being the last player who has not gone bankrupt.

## **4 Nonfunctional requirements**

### **4.1 Usability**

Any English-speaking person should be able to play the game without asking for help.

A single game view should have no more than 7 buttons

While navigating from view to view, the process should take less than 3 clicks

The guide window should be accessible during the game with at most 1 click

In instances where a button does not have text on it, it should have a tooltip when the cursor hovers over it

### **4.2 Reliability**

During the game, any player's connection problem should not affect the game from being played by the other players. The player who faces a connection problem should be displayed as a bankrupt player to others.

The players should be notified in case of server connection problems.

### **4.3 Performance**

The video and voice chat latency should be less than 5 seconds.

The game should support Windows, Linux, and Mac computers.

The game should be playable on computers without a dedicated GPU.

The game should be playable on computers with more than 4GB RAM.

The game should take less than 5GB disk space.

It should take less than 3 seconds for user input to reach the server and be processed

### **4.4 Supportability**

Any client update should not require the server update.

## **5 Pseudo Requirements**

The programming language should be Java.

No library or framework that forces a particular design should be used.

## 6 System models

### 6.1 Use Case Model

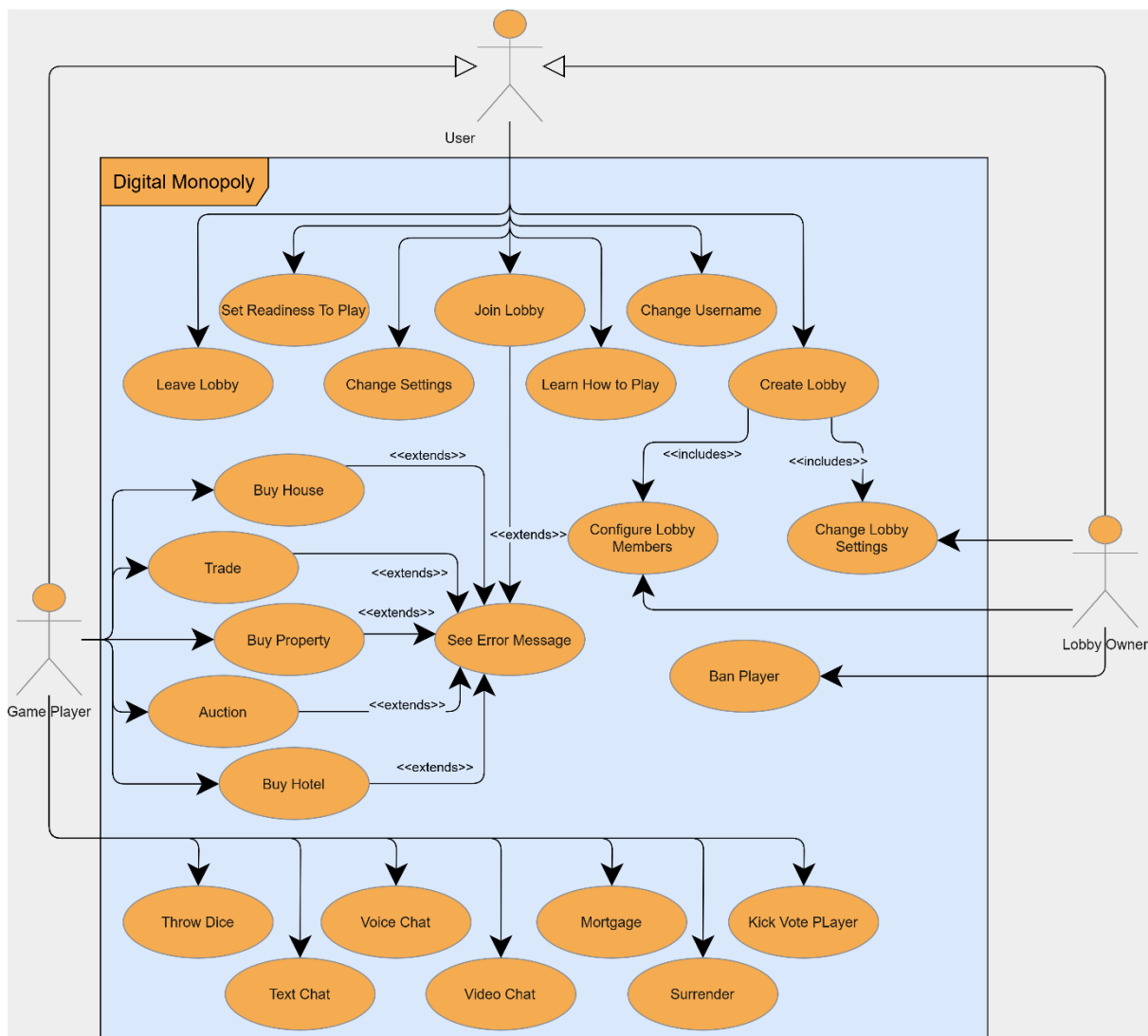


Figure 1. Use-Case Diagram

#### 6.1.1 Learn How to Play

Use Case Name: How to Play

Participating Actors: User

Entry Conditions: User clicks on how to play button or question mark icon

Exit Conditions: User closed the pop-up screen

The Flow of Events: 1. A pop-up screen with a game guide is shown

Quality Requirements: Any English-speaking person should understand the guide

### 6.1.2 Change Settings

Use Case Name: Change Settings

Participating Actors: User

Entry Conditions: User clicks change settings button or settings icon

Exit Conditions: User saves the changes or closes the screen

The Flow of Events:

1. User can change his/her personal information such as username, icon
2. User can alter the default audio settings (volume, effect sounds).
3. User can disable/enable camera, audio, and text chat
4. User approves the changes by clicking the save button or the changes are dismissed.
5. System updates the changes if the user chooses the save them.

### 6.1.3 Join Lobby

Use Case Name: Join Lobby

Participating Actors: User

Entry Conditions: The User clicks the join lobby button.  
There is at least one lobby created in advance.

Exit Conditions: User joins the lobby or cancels, and returns to the main screen.

The Flow of Events:

1. User enters the lobby credentials or selects the lobby from the lobby list.
2. User joins the lobby clicking to join button and joins the lobby

Quality Requirements: The server should respond within 3 seconds.

### 6.1.4 Create Lobby

Use Case Name: Create Lobby

Participating Actors: User

Entry Conditions: The User clicks the create lobby button.

Exit Conditions: New lobby is created.  
User is the owner of the new lobby.

The Flow of Events:

1. User is assigned to be an Owner of the created lobby.
2. Owner configures the lobby settings (id, name, size).
3. Lobby is created and the Lobby Owner is assigned to the lobby.
4. Owner can configure (remove, make admin) members of the lobby.

Quality Requirements: The server should respond within 3 seconds.

### 6.1.5 Set Readiness to Play

Use Case Name: Set Readiness to Play

Participating Actors: User

Entry Conditions: User must be in a lobby

Exit Conditions: User becomes ready to play, or exits the lobby.

The Flow of Events: 1. User toggles the readiness using a button and waits for other players to be ready.

Quality Requirements: The server should process the request within 3 seconds.

### 6.1.6 Auction

Use Case Name: Auction

Participating Actors: Game Player

Entry Conditions: A Game Player lands on an unowned property, and chooses to auction it

Or a Game Player has gone bankrupt due to owing to the Bank

Or a Game Player wants to buy the last available house/hotel from the Bank

Exit Conditions: The auctioned property is given to the highest bidder, or the auction ends with no one bidding

The Flow of Events: 1. Auctioned property is determined.  
2. The starting price is set to 10AA  
3. All players can increase the current bid by any amount  
4. All players mark themselves as satisfied with the current bidding state  
5. The property is given to the highest bidder, or the bank if no one bids

Quality Requirements: The server should respond within 3 seconds.

### 6.1.7 Mortgage

Use Case Name: Mortgage

Participating Actors: Game Player

Entry Conditions: Game Player owns at least one mortgageable property.

Game Player clicks the Mortgage button.

It is the Game Player's turn

Exit Conditions: The property becomes mortgaged.

Relevant game rules apply for the following turns.

The Flow of Events: 1. Game Player chooses which property he/she wants to mortgage.

2. System retains possession of the property from the player and labels the property as mortgaged.
3. System returns the half worth of the property to Game Player.

Quality Requirements: The server should respond within 3 seconds.

### 6.1.8 Trade

Use Case Name: Trade

Participating Actors: Game Player

Entry Conditions: Game Player clicks the Trade button.

It is the Game Player's turn

Game Player is not in an ongoing Trade

Exit Conditions: The system performs the trade operation successfully or one of the players cancels the operation.

The Flow of Events:

1. A trade offer is sent to the target Game Player.
2. Both sides add/remove tradable properties or money until agreement or refusal
3. If the Game Player and the targeted Game Player agree on a trade, the system exchanges traded belongings of each player.

Quality Requirements: The server should respond within 3 seconds.

### 6.1.9 Buy House

Use Case Name: Buy House

Participating Actors: Game Player

Entry Conditions: Game Player clicks the buy house button of the selected street.

Game Player owns the whole color set of the street.

Game Player has less than 4 houses on the street.

There is a house in the bank.

The number of houses and hotels on the color set will stay evenly distributed if another house is added to the selected street.

Exit Conditions: Game Player cancels the operation.

Or

The house is added to the street.

The cost of a house is taken from the player.

The Flow of Events:

1. Game Player specifies which street he/she wants to build a house on.

2. If the Game Player confirms the operation, the system decreases the cost of the house from the Game Player's balance.
3. If Game Player confirms the operation, the system increases the rent of the property to the predefined price of the street.

Quality Requirements: The server should respond within 3 seconds.

#### **6.1.10 Buy Hotel**

Use Case Name: Buy Hotel

Participating Actors: Game Player

Entry Conditions: Game Player clicks the buy hotel button of the selected street.

Game Player has 4 houses on the street.

There is a hotel in the bank.

The number of houses and hotels on the color set will stay evenly distributed If the hotel is added to the selected street.

Exit Conditions: Game Player cancels the operation.

The hotel is built on the street.

4 houses from the property are returned to the bank.

The cost of a hotel is taken from the player.

The Flow of Events: 1. Game Player chooses which street he/she wants to build a hotel on.

2. If Game Player confirms the operation, the system decreases the cost of the hotel from the Game Player's balance.

3. If Game Player confirms the operation, the system increases the rent of the property to the predefined price of the street.

Quality Requirements: The server should respond within 3 seconds.

#### **6.1.11 Buy Property**

Use Case Name: Buy Property

Participating Actors: Game Player

Entry Conditions: Game Player selects a property and clicks to buy button.

Game Player has the turn.

The property is not owned by another player.

Exit Conditions: Game Player buys the property

Money is withdrawn from the Game Player's account and the property is assigned.

Game Player cancels the action.



Game Player does not have enough money to buy the property.

The Flow of Events: 1. Game Player selects the property to buy.

2. If the player confirms the operation money is withdrawn from the Game Player's balance.

3. If the payment is successful property's ownership is assigned to the Game Player.

Quality Requirements: The server should process the request within 3 seconds.

## 6.2 Dynamic models

### 6.2.1 Sequence Diagrams

#### 6.2.1.1 Lobby Creation Sequence Diagram

**Scenario:** The user wishes to create a lobby

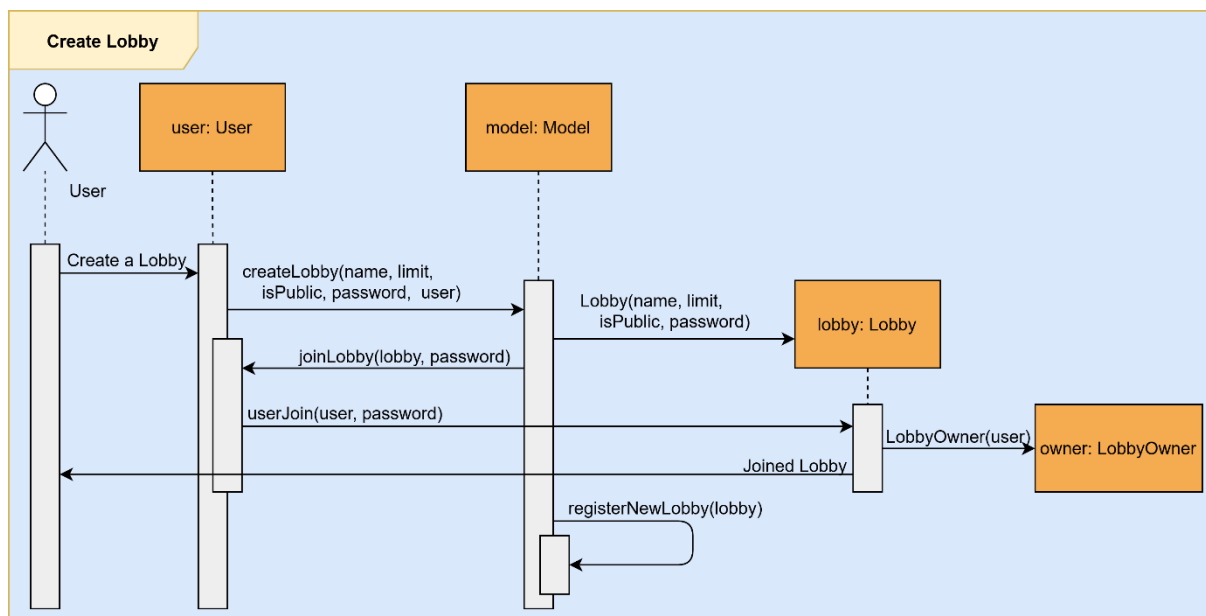


Figure 2. Create Lobby Sequence Diagram

The user wants to create a lobby, they then make the appropriate function call to the model instance which initializes a lobby object. The user who created the lobby is then made to join the newly created lobby, after which a "LobbyOwner" instance is created and assigned to be the user. Finally, the newly created lobby is added to the lobby list within the model instance.

### 6.2.1.2 Lobby Join Sequence Diagram

**Scenario:** The user wishes to join an existing lobby

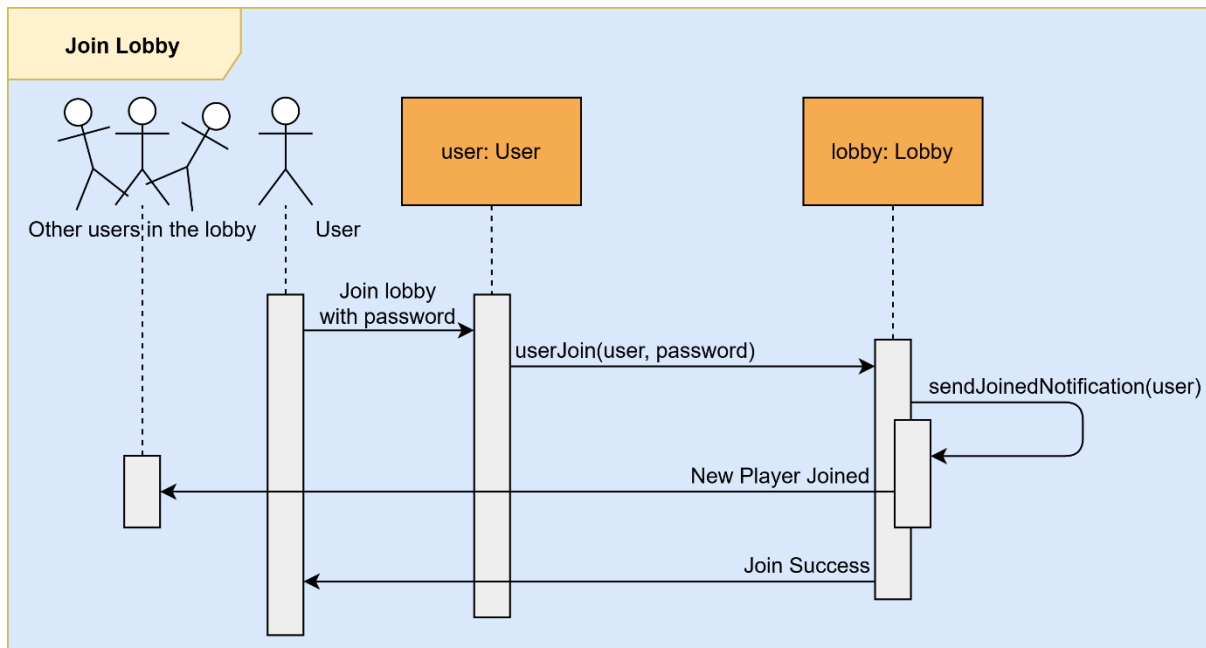


Figure 3. Join Lobby Sequence Diagram

The user wishes to join a lobby which they see on the lobby list, they then send the request to join with the appropriate password (if it exists) and the notification for a joining user is processed within the lobby instance. The users already in the lobby and the user joining then get an update in their client which shows them that a new user has joined the lobby.

### 6.2.1.3 Trade Sequence Diagram

**Scenario:** A player initiates a trade with another player

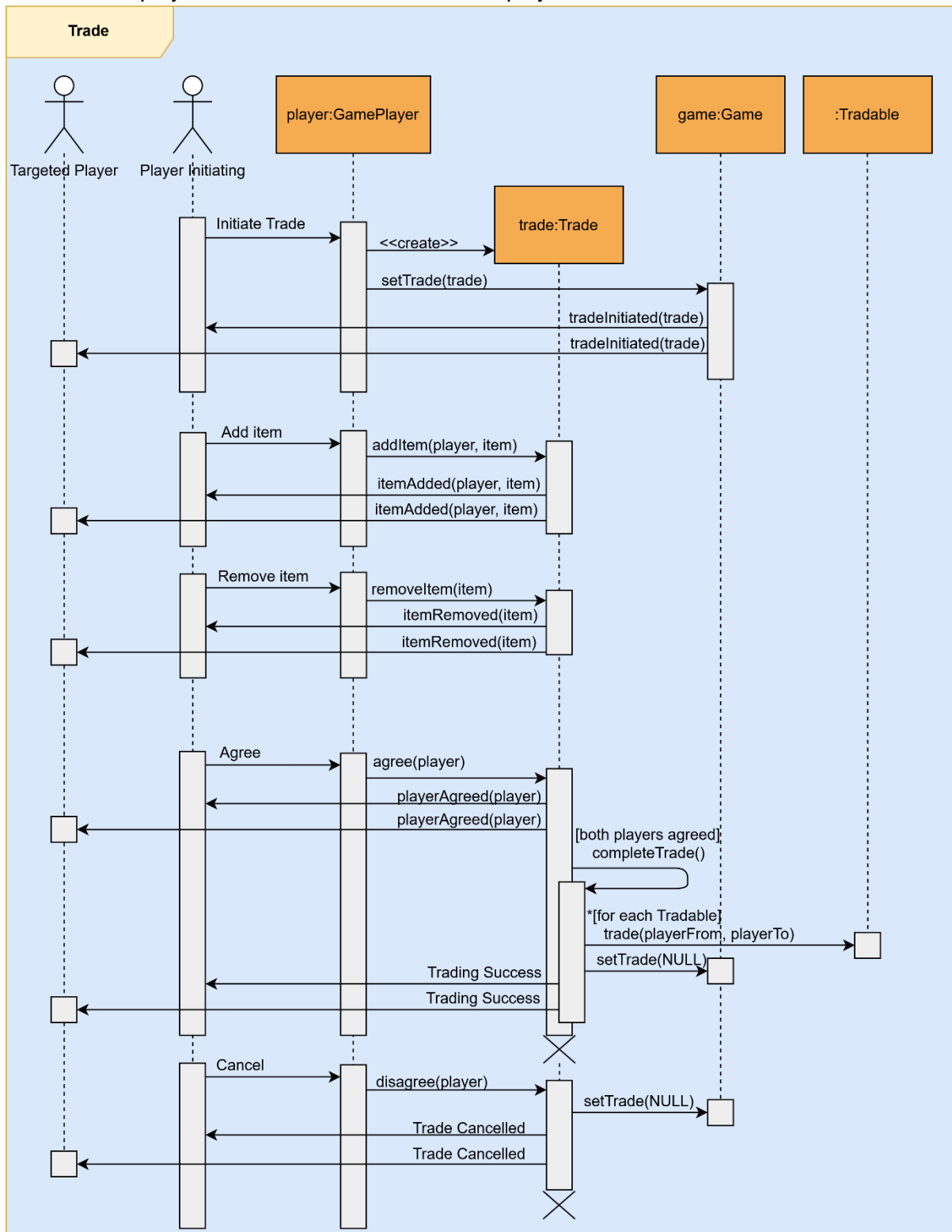


Figure 4. Trade Sequence Diagram

When a player chooses to start a trade with another player, a Trade instance is created and assigned to the appropriate variable in the existing Game instance. The notification for the

start of the trade is then sent to the player who started said trade, as well as the other player, and they can then take action accordingly.

The players in a trade can add and remove items from the existing trade, which will update their client GUI accordingly, as well as agree or disagree with the trade. A single player canceling a trade will cause the trade to be finished without any item exchange. If both players agree to the item exchange listed in the trade, the trade is executed and the tradeable items are transferred from one player to the other. When a trade is finished either due to agreement or canceling, the trade instance is set to null.

#### 6.2.1.4 Auction Sequence Diagram

**Scenario:** A player puts up an auction-able item for auction

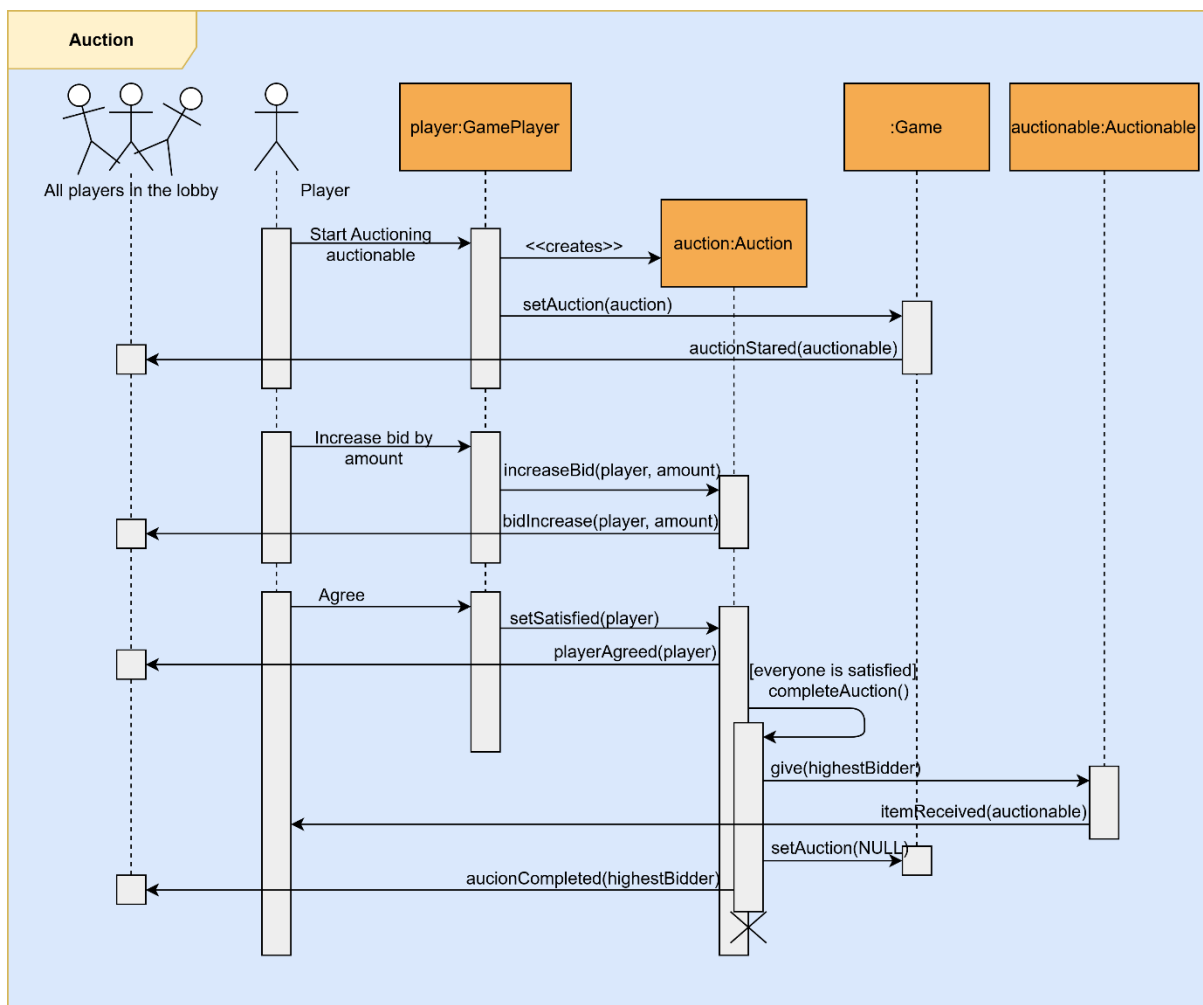


Figure 5. Auction Sequence Diagram

When a player starts an auction, an Auction object is instantiated, and this object is assigned to the Game instance. Afterward, the item that is being bid on is set by the GamePlayer instance and the auction starts.

During the auction, players can increase the bid amount for the item, and the last player who has done the said increase is saved into the Auction instance. At the end of the auction, if the player who started said auction agrees to the bid amount that is proposed, the “setSatisfied”

method is called and the item is given to the last bidder if they exist. The Auction instance is then destroyed and the reference inside the Game instance is now NULL.

### 6.2.1.5 Bankruptcy Sequence Diagram

**Scenario:** A player goes bankrupt

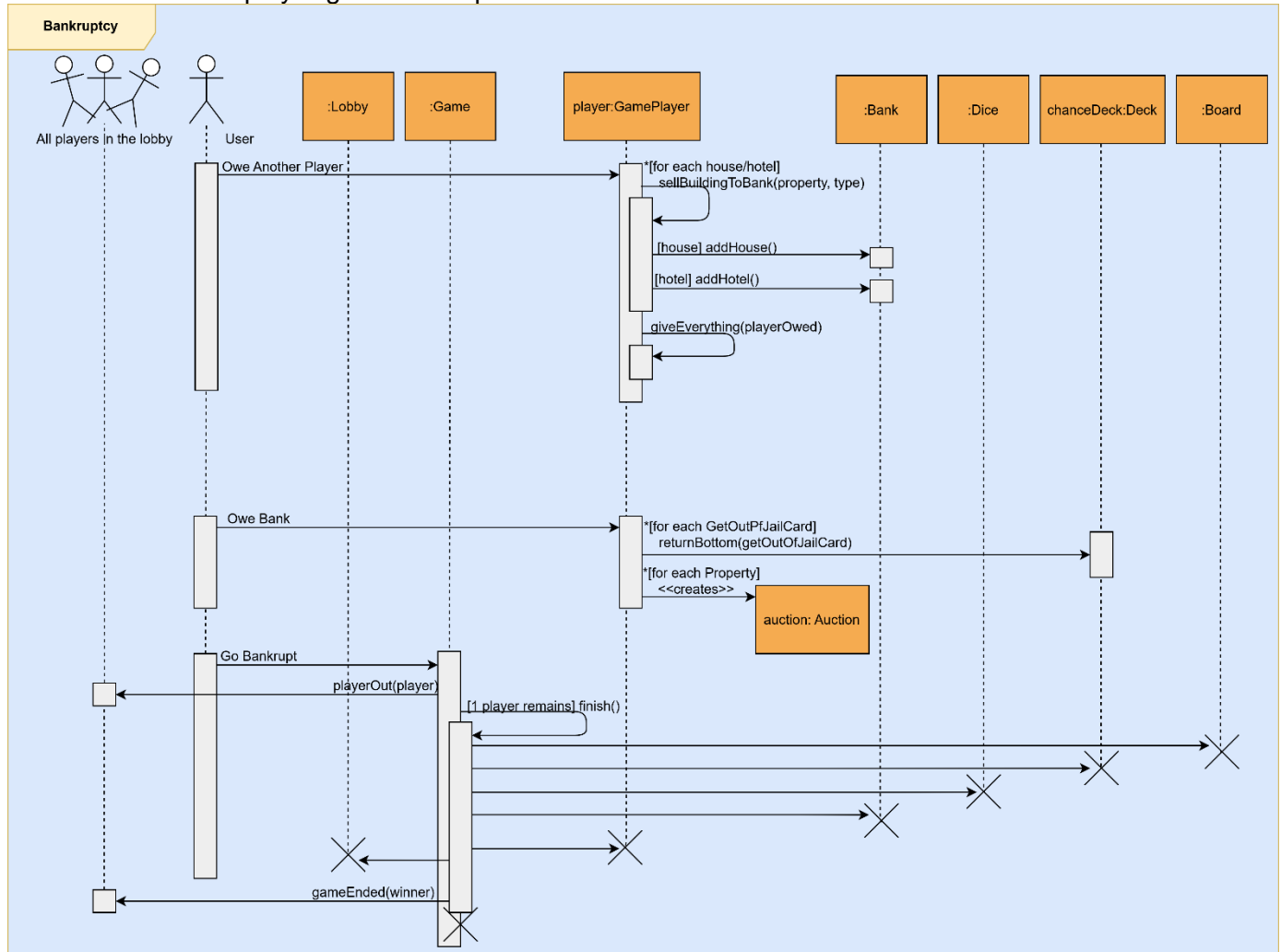


Figure 6. Bankruptcy Sequence Diagram

A bankruptcy occurs when a player owes an amount of money to the bank or another player that is larger than what they can payback. In that case, if the money is owed to the bank, all of the player's buildings and get out of jail cards are returned to their starting points and their properties are auctioned off by the bank. If the player instead owes another player, all of their buildings are sold to the bank at half the cost, and their money, as well as their properties, are transferred to the player that they are in debt to.

The player is then removed from the game, in which case, if more than 1 player is remaining the game continues as normal. In the case where only a single player remains, that player is declared the winner and the game is completed.

## 6.2.2 State Diagrams

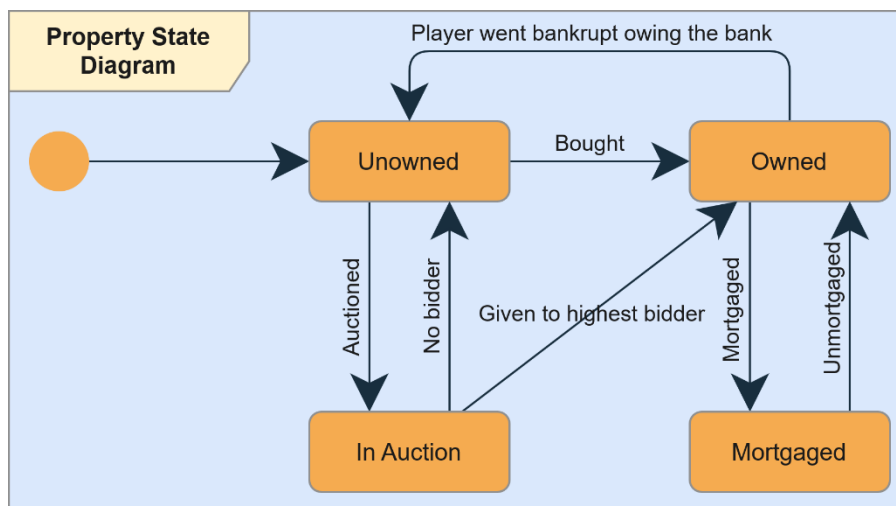


Figure 7. Property State Diagram

The above diagram illustrates the states that a property tile will go through during a typical game of monopoly. The tile always starts as unowned and can be either bought by a player to become an owned tile or can be put to auction where it will be given to the highest bidder. An owned property can then be mortgaged or unmortgaged by the player if they choose so, and the property will become unowned if the owner of the said property owes money to the bank.

## 6.2.3 Activity Diagrams

### 6.2.3.1 Lobby Activity Diagram

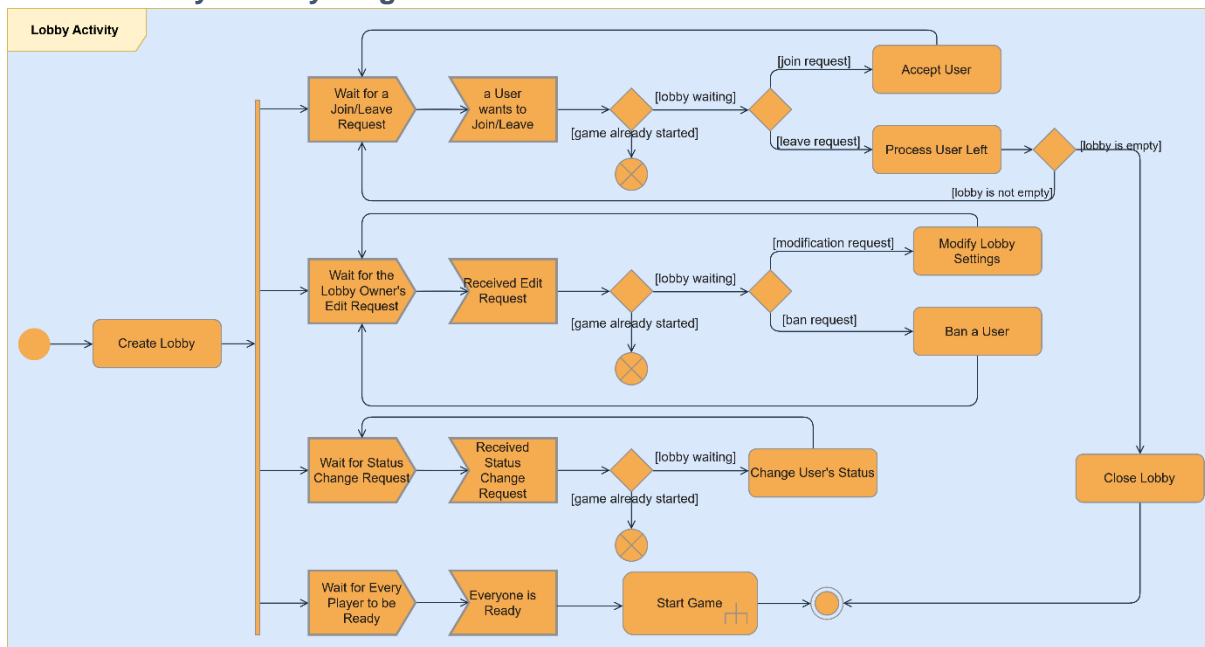


Figure 8. Lobby Activity Diagram

The above diagram illustrates how the lobby system is run. At the start, the lobby is created and there exists a lobby owner who can edit the lobby properties as well as ban players from joining the lobby, at which point said properties in the instance are changed and the lobby waits for other input. In the same vein, if a user joins or leaves the lobby, the request is processed and the user instance is added or removed from the lobby user list, if the list is empty then the lobby is closed.

Alternatively, the users in the lobby can choose to change their status, in this case, the request is processed and the user status becomes ready or not-ready. If all the player in the lobby has the ready status, then the game starts.

### 6.2.3.2 Gameplay Activity Diagram

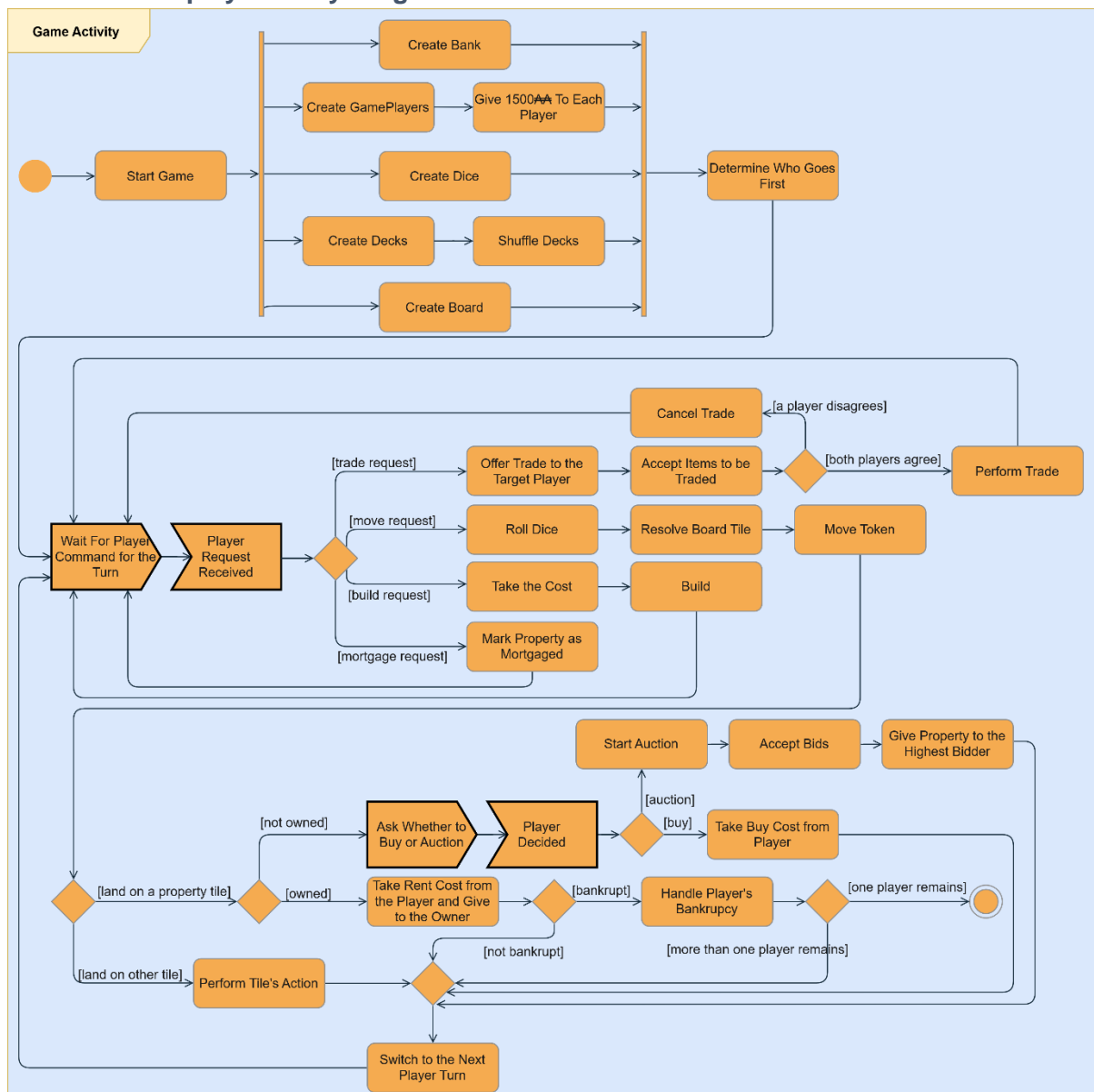


Figure 9. Gameplay Activity Diagram

The above diagram explains the series of action flows that would denote the gameplay portion of the monopoly game.

The start of the game immediately causes several instances of different objects to be created as per the gameplay sequence diagram. Afterward, the bank gives 1500~~AA~~ to each player and shuffles the card deck, following this, the turn order is established and the game starts for the users.

During each player's turn, said player is allowed to make 4 action requests: trade, move, build, and mortgage requests.

The trade request starts a trade action with another player, and when the trade action ends by either agreement or disagreement of both parties, the system goes back to waiting for a player request.

The build request allows the player to build on an owned tile if they have the funds to do so.

The mortgage request simply marks the property as mortgaged, from where the mortgage rules take effect for said property

Finally, the move request rolls 2 six-sided dice using the Dice class and moves the player a certain amount. If the tile which the player lands on is not a property tile, the tile's action is performed when the relevant method is called and the turn order progresses to the next player.

If the landed tile is an unowned property tile, the player is asked whether they would like to buy the tile or not, where depending on their answer the property is transferred to either the highest bidder or the buying player. If the tile is owned, the moving player pays the rent amount, which is taken directly from their account.

If the rent amount exceeds the player's account balance, they are declared bankrupt. The game continues normally unless only one player remains active, in which case they have won and the game ends.



6.3 Object and Class Models

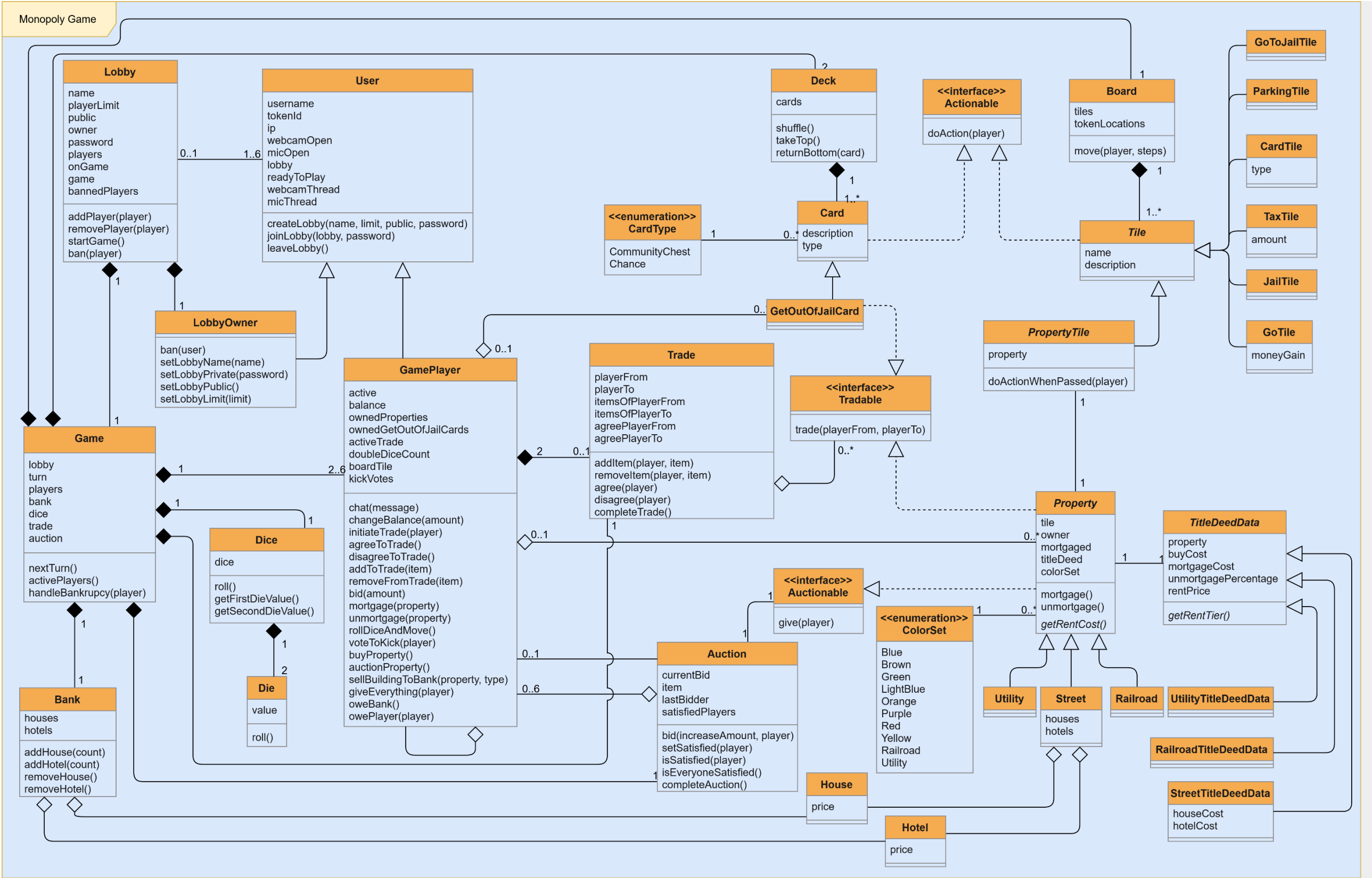


Figure 10. Application Domain Class Diagram

Illustrated above is the class diagram for our Monopoly game. Throughout the usage of the program, the Model class has the core of the program's functionality and is also in aggregation with other core classes.

### **6.3.1 Lobby Class**

The Lobby class represents the group of players that will be starting the game after a certain amount of user instances has given the ready signal. The class has a list of player instances (A List of “User” classes) and a list of banned players that are prohibited from accessing the lobby. It also has a “LobbyOwner”, which is a class that extends user. The lobby owner can control whether the lobby is a private or public one, and change the lobby properties such as the lobby password, the player limit, and the players who are banned from the lobby.

### **6.3.2 User Class**

The User class represents the players, and is extended by two other classes which are “LobbyOwner” and “GamePlayer”. The User class has the necessary functionality for joining and leaving lobbies as well as the related data such as the player token, user name, and IP address. The class also has boolean values that represent whether the webcam and microphone functions are being used, as well as the relevant threads. This is because the “GamePlayer” class utilizes these properties during the game as players speak with one another.

### **6.3.3 GamePlayer Class**

This class directly extends the User class and handles most of the player functionality during an ongoing game. It has data structures containing the properties and the get out of jail cards that the player has possession of, as well as the relevant properties to keep track of the player’s situation in the game, such as the tile they are on currently, their rolled dice value, a reference to Trade instance if it exists, and their monetary balance.

The methods in this class provide most of the player’s interaction with the game’s systems, for example, the methods that allow the player to trade, mortgage property, and buy/auction property are in this class, as well as simpler methods such as “chat” and “rollDiceAndMove”

### **6.3.4 Game Class**

The Game class controls and manages the functional aspects of the game, it has references to the “Bank” class as well as the current trade that is being performed. The class mainly handles the progression of the game, as it can call the “nextTurn” method, allowing the next player in the turn order to become active and perform their gameplay actions. The class is also able to cause a player to go bankrupt when the rules demand it, which is what advances the game to its completion

### **6.3.5 Trade and Auction Classes**

The trade class has most of the functionality for the trade mechanic in the game. It holds references to the player who initiated the trade sequence, the items that are proposed, and its conclusion. It’s methods exclusively allow players to add and remove properties from the trade agreement and agree/disagree on the trade itself.

The Auction class, in a similar vein only exists to facilitate the auction mechanic, it has fields for the current item and the bid, it allows players to bid on an item so it can be auctioned off.

### **6.3.6 Enumerations and Interfaces**

The enumerations in the diagram represent the type of card that can be yielded when a draw card tile is activated, and the color sets that a property can be a part of. This color set is

important in establishing the logic behind building placement, and the card types each have different functionalities as well as different card instance pools they can draw from. Therefore, these enumeration distinctions are needed.

The interfaces are there to facilitate joint functionality between differing objects, such as the Actionable interface, which has a doAction method. There also exists the Tradeable interface that has the “trade” method.

### **6.3.7 Tiles, Properties, Buildings, and Extras**

The tile and property classes, as well as the card class, have very little functionality. Properties implement the “Tradeable” interface, as well as the “Auctionable” interface, and the cards, as well as the tiles, implement the “Actionable” interface, which gives them their primary functionalities. The properties also have distinct owners as well as an assigned tile, they can also be mortgaged and have a rent cost. House and hotel classes represent buildings that can be bought and placed by players.

All of these classes extend to many children of a different type that share more or less the same function and only add one or two variables.

There also exists the “TileDeedData” class, which along with its children holds the relevant data for rent price, buy cost, and other such values.

## 6.4 User Interface

### 6.4.1 Navigational Path

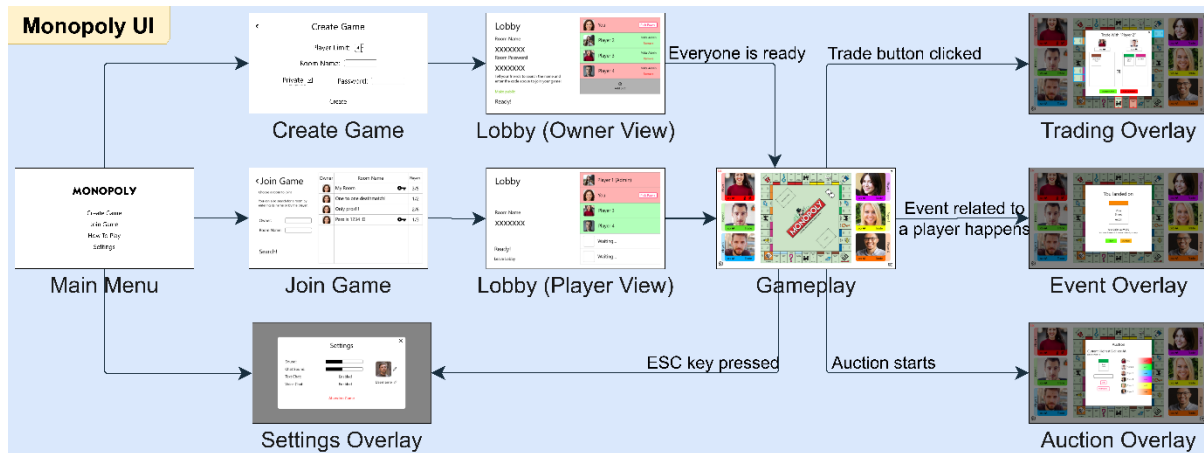


Figure 11. UI Navigational Flow Diagram

The UI navigational flow of the project is illustrated above. The path provided starts from the main menu screen and continues up to the start of a game session, ending at several different screens showcasing gameplay functionalities such as trading, buying property, and auctioning. The settings screen also can be accessed from both in-game and the main menu.

Live animated demo of the UI can be found at [\[2\]](#).

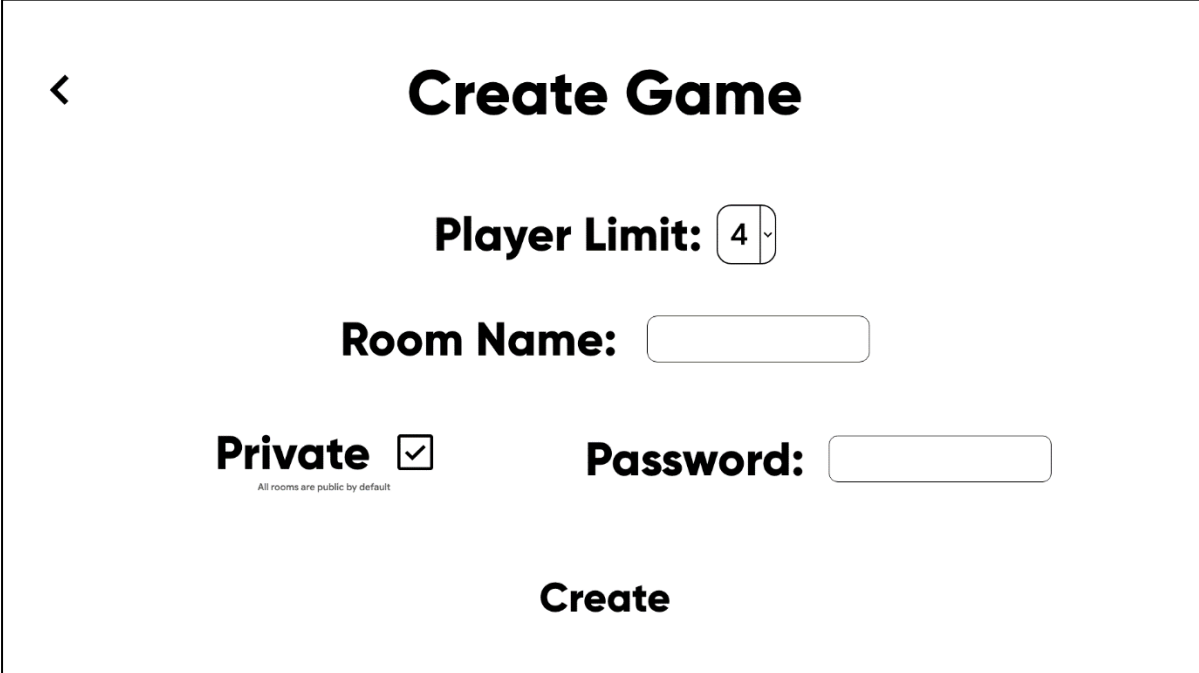
### 6.4.2 Menu



Figure 12. Main Menu Screen

The main menu screen hosts four buttons. The “Create Game” button leads to the “Create Game” screen while the “Join Game” menu leads to the “Lobbies” screen. The “How To Play” button leads to a simple screen where the rules of Monopoly are briefly discussed. Clicking the “Settings” button will summon the “Settings” popup.

### 6.4.3 Create a Game

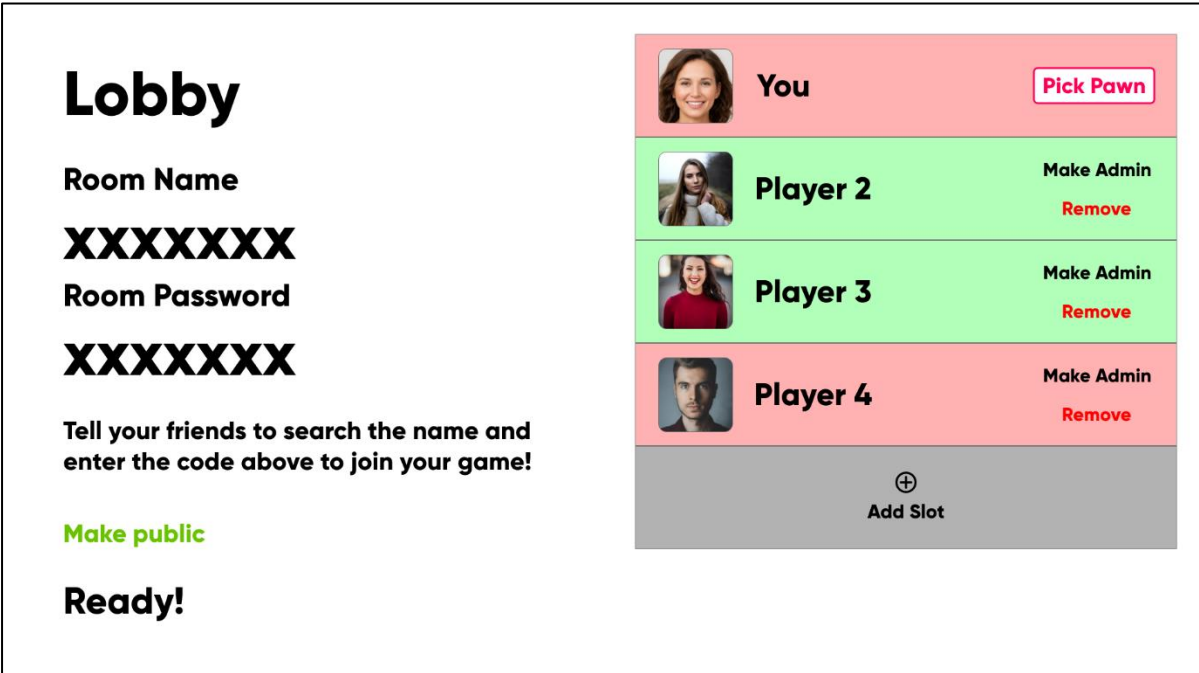


The 'Create Game' screen features a back arrow in the top left corner. The title 'Create Game' is centered at the top. Below the title, there is a 'Player Limit' section with a dropdown menu currently showing '4'. Underneath is a 'Room Name' label followed by a text input field. The 'Private' option is selected, indicated by a checked checkbox, with a small note below it stating 'All rooms are public by default'. To the right of this is a 'Password' label followed by another text input field. At the bottom center, there is a large 'Create' button.

Figure 13. Create Game Screen

Within this screen, the player can determine the number of players through a dropdown menu that maxes at 6. The player can also determine the room name and whether the room will be public or private by checking the checkbox provided. Consequently, they define a password and create the game, proceeding to screen 6.4.4.

### 6.4.4 Lobby: Owner View



The 'Lobby: Owner View' screen is divided into two main sections. The left section, titled 'Lobby', contains the 'Room Name' (displayed as 'XXXXXXXX'), the 'Room Password' (displayed as 'XXXXXXXX'), and a message: 'Tell your friends to search the name and enter the code above to join your game!'. Below this message are two buttons: 'Make public' (in green) and 'Ready!' (in bold). The right section displays a list of players in a table-like format. The first row is for 'You' (the owner) with a 'Pick Pawn' button. The following three rows are for 'Player 2', 'Player 3', and 'Player 4', each with 'Make Admin' and 'Remove' buttons. At the bottom of this list is a grey button with a plus icon and the text 'Add Slot'.

Figure 14. Lobby Screen - Owner View

If you are the lobby owner, i.e., if you have created a lobby by clicking the “Create Lobby” button in the main menu, you have the options to make other players admins, make the lobby public or private, add an extra slot for more players if possible and finally, create the game once everyone has joined the lobby.

#### 6.4.5 Join Game

## < Join Game

Choose a room to join!

You can also search for a room by entering its name or by the player.

Owner:

Room Name:

**Search!**







Owner	Room Name	Players
	<b>My Room</b> 	<b>3/5</b>
	<b>One to one deathmatch!</b>	<b>1/2</b>
	<b>Only pros!!!</b>	<b>2/6</b>
	<b>Pass is 1234 :D</b> 	<b>1/3</b>

Figure 15. Join Game Screen

The “Join Game” screen is summoned when a player clicks the “Join Game” in the main menu. Here, on the right, they can see the private (indicated by the key icon) and public lobbies that are awaiting a member. From here, the player can either click on a lobby to join it or they can search a lobby either by the player name or the lobby name.

#### 6.4.6 Lobby: Player View

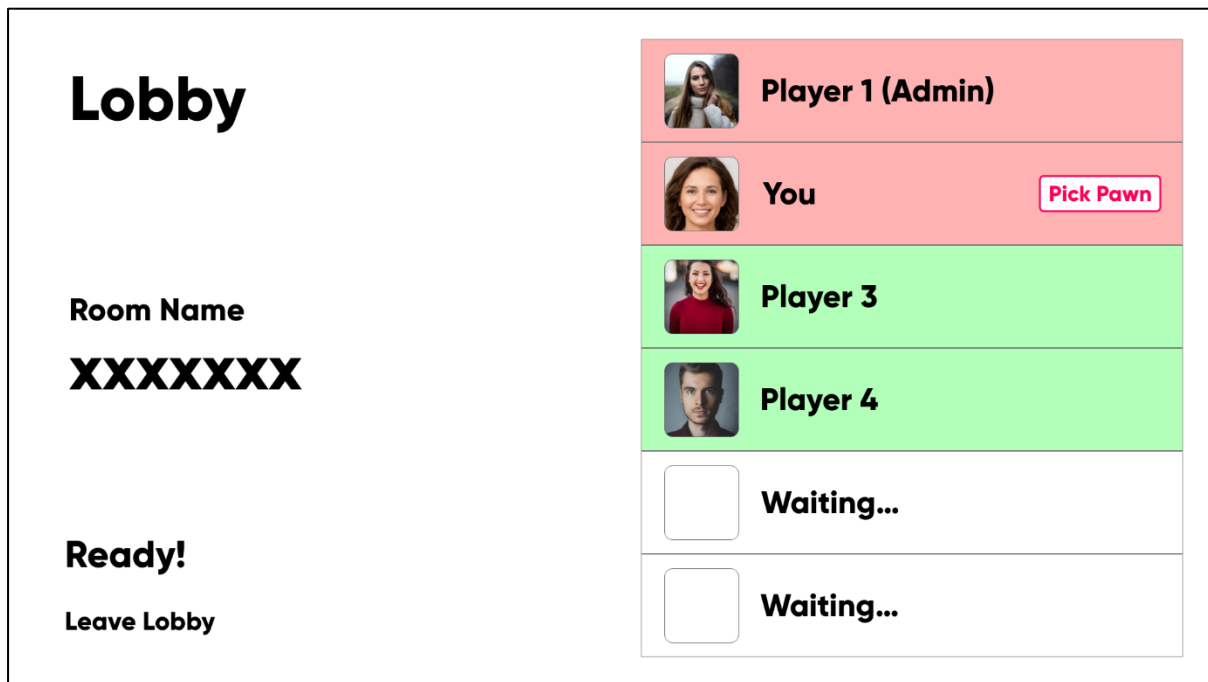


Figure 16. Lobby Screen - Player View

When you click “Join Lobby” in the “Join Game” screen, you enter this view in which you can leave the lobby and pick your pawn.

#### 6.4.7 Settings

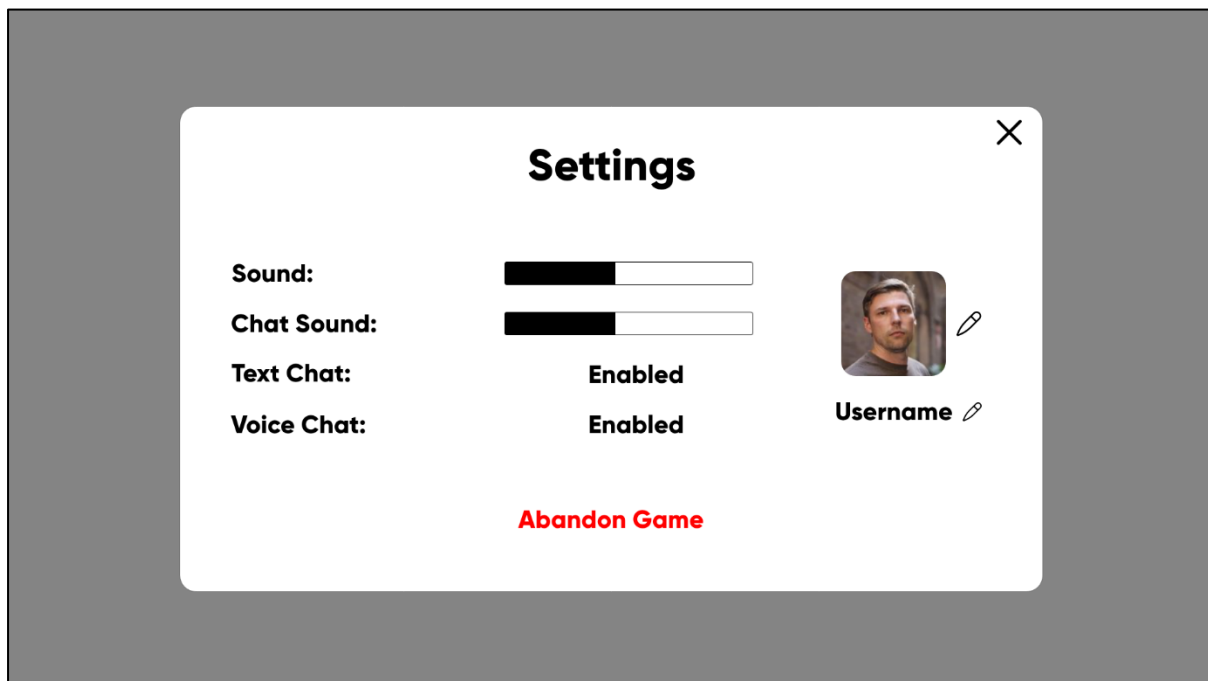


Figure 17. Settings Overlay

In the settings page, the player can change the master sound of the game (i.e. music, effect, voice, and video chat volumes), they can change the video chat sound specifically, enable

and disable text chat and voice chat and change their username and profile picture. Also, if they are in a game, they can abandon it.

#### 6.4.8 Main Game View

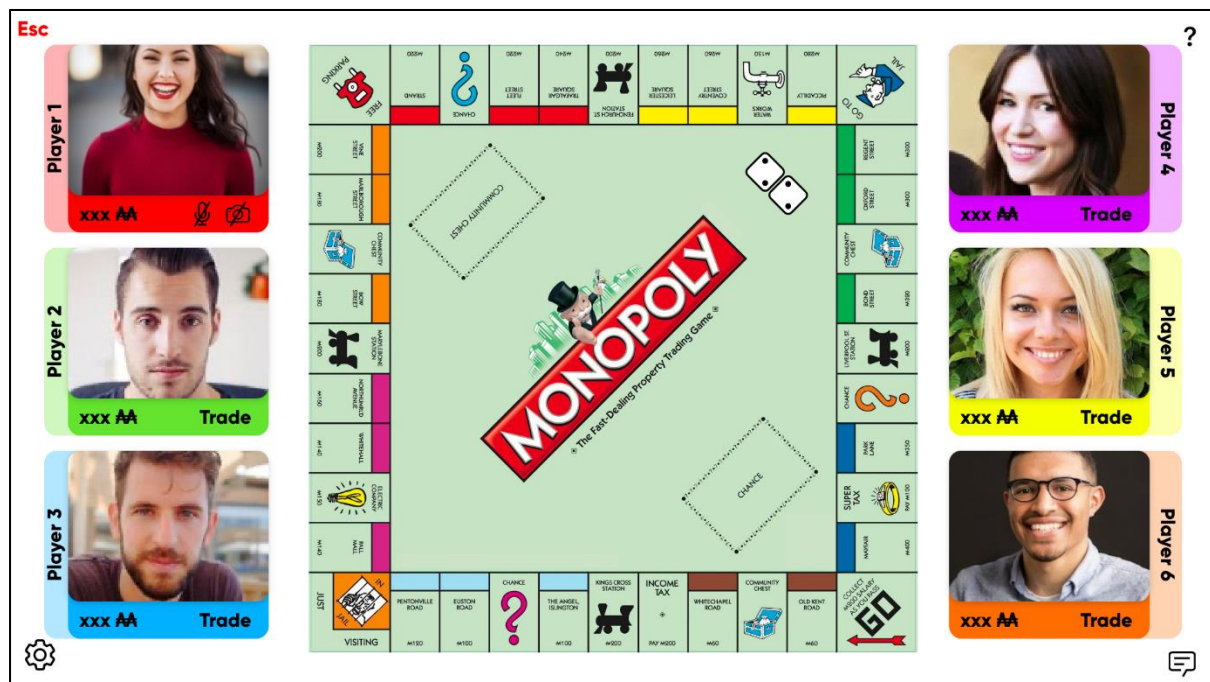


Figure 18. Main Game Screen

This is the main game screen. The player can roll dice, trade with other players, see their balances, video chat, and text chat with them. By clicking the question mark at the top right, they can open up the help menu of the game, they can open up settings by clicking the button at the bottom left or by clicking the escape key. They can also close and open their microphone and camera with the buttons situated at the bottom right of their player avatar.



## 6.4.9 Text Chat



Figure 19. Main Game Screen with Text Chat Overlay

This is the game screen when the chat button is clicked. The players can text chat from this overlay.

## 6.4.10 Popups

### 6.4.10.1 Auction



Figure 20. Auction Overlay

If a player lands on a property they cannot buy because they have insufficient funds or they simply do not want to buy that property, they can sell it at auction. This is the screen for that functionality. The players can enter higher and higher amounts of balance to try to acquire the property and the auction is concluded when all players withdraw from the auction.

#### 6.4.10.2 Trade



Figure 21. Trading Overlay

Trading is a big part of our game. A player can initiate a trading process by clicking the “Trade” button situated at the bottom right color of every other player’s avatar. This will highlight the properties owned by the two players in the trade and put a border around them corresponding to the owner player’s color. Here, we can see that Player 1 has initiated the trade with Player 2 and their properties have been highlighted (Red for Player 1 and blue for Player 2).

Once Player 1 also puts something to give (i.e., money, property, or a “Get Out of Jail Free” card), they can make the offer.

The player who is being offered the trade also adds new assets and when both parties are happy, they can both click “Accept Offer” and the trade is successful. The trade is unsuccessful if either party clicks “Reject Offer”.

### 6.4.10.3 Landing on a Property

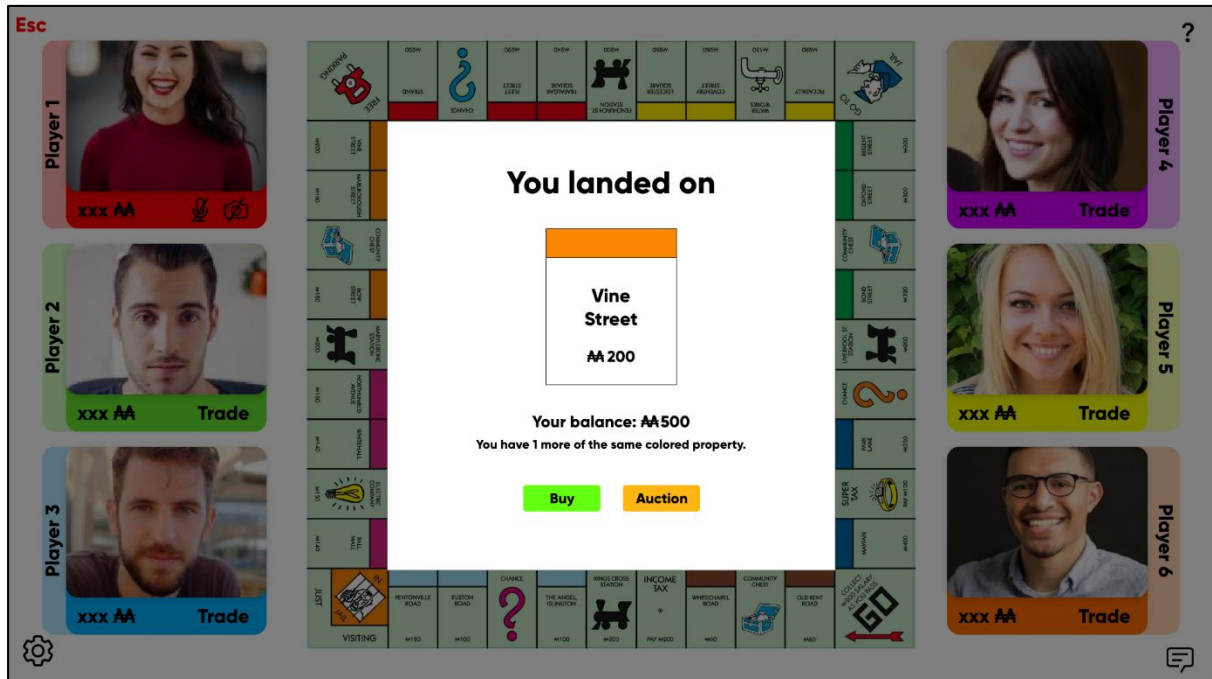


Figure 22. Landing on a Property Overlay

If the player lands on a property, they either have to buy it or auction it.

### 6.4.10.4 Miscellaneous

<p><b>CHANCE</b></p> <p>You have payed to renew the sound systems of your karaoke bar in Oxford Street. Pay 50 AA.</p>	<p><b>COMMUNITY CHEST</b></p> <p>You won a ticket to get out of jail. You can use it or trade it.</p>	<p><b>GO TO JAIL</b></p> <p>Oh no! It looks like you have committed a crime...</p>	<p><b>INCOME</b></p> <p>Salary due! Collect 200 AA</p>
<p>Used when player lands on "Chance"</p>	<p>Used when player lands on "Community Chest"</p>	<p>Used when player lands on "Go to Jail"</p>	<p>Used when player passes "Go"</p>
<p><b>INCOME TAX</b></p> <p>You payed 200 AA as a luxury tax.</p>	<p><b>LUXURY TAX</b></p> <p>You payed 100 AA as a luxury tax.</p>	<p><b>IT'S YOUR TURN!</b></p> <p>Click on the die to roll them!</p> 	<p><b>You Are In Jail!</b></p> <p>Roll Double Pay 50 AA Use "Get Out of Jail Free" Card</p>
<p>Used when player lands on "Income Tax"</p>	<p>Used when player lands on "Luxury Tax"</p>	<p>Used when it is the player's turn</p>	<p>Used when the player is in jail</p>

Figure 23. Miscellaneous Overlays

## 7 Improvement Summary

After the previous iteration, the following changes were made:

- Added Section 2.1 “Changes from the Original Board Game”: The summary of the additions to the original board game
- Improved Use Case Diagram: The diagram detailing the use cases of our game has been improved.
- Sequence Diagram Improvements: The diagrams describing sequence diagrams now have figures representing the remaining players other than the acting user, as well as data flow from the active instance back to the users. Diagrams are now more specific.
- Class Diagram: Previous class diagram is changed to the application domain class diagram since that is more appropriate for the analysis report. The old diagram is moved to the design report.
- Added Section 6.2.2 “State Diagrams”: Contains a new state diagram for a property in Monopoly.
- Added new gameplay functional requirements that include the game end conditions
- Added captions to all figures
- Other small “bug-fixes”

## 8 Glossary & references

[1] “Monopoly Classic Guide”.

<https://www.hasbro.com/common/documents/A0AFE3A69EC745EBA77B9A7950BBCA44/AD7742057B1D43609B53D24D75E9CA9B.pdf>. [Accessed: Nov 1, 2020]

[2] “Adobe XD – Live UI Demo”.

<https://xd.adobe.com/view/8b255e68-92e0-42f1-857e-9d397d388a97-de76/screen/bcdcf8ff-5f63-4b97-b012-ad1d3000e6b3>. [Accessed: Nov 1, 2020]