Bilkent University

Department of Computer Engineering

CS319 Term Project

Project short-name: Scrolls of Estatia

Analysis Report

Atakan Sağlam, Sarp Ulaş Kaya, Berk Kerem Berçin, Oğulcan Çetinkaya, Furkan Başkaya

Instructor: Eray Tüzün
Teaching Assistant(s): Barış Ardıç, Emre Sülün and Elgun Jabrayilzade

Iteration 1/Project Analysis Report
Dec 13, 2020

This report is submitted to the Department of Computer Engineering of Bilkent University in partial fulfillment of the requirements of the Senior Design Project course CS319.

Contents

Analysis Report

Project short-name: project title

1 Introduction

Monopoly is a board game which can be played with 2-8 players. Each player has their own token which they can move through the board by Rolling two 6-sided die. As the game progress, players can buy neighborhoods, build houses and hotels on them, buy railroads and utilities, pay taxes, draw community chests and chance cards. The main objective of the game is to buy as many properties as possible and achieve total monopoly while bankrupting the remaining players. When a player's token visits a property, which belongs to another player, the player needs to pay a rent in order to continue the game.

Our twist on the original game is that there will be classes and each class will have different advantages and disadvantages, thus encouraging the player to pick a class based on their preferred strategy and playstyle. Therefore, multiple players will be able to choose the same class, so that every player has the same options. Another major change from the original game is that the Community Chest and Chance cards are combined as a single deck named Fortune Cards, and there is a new deck named Scrolls which can be kept in the player's hand and be used anytime during their turn. Our game will also have a medieval fantasy theme with twists on Monopoly's original mechanics to fit the theme.

2 Proposed System

2.1 Overview

2.1.1 Changes to Existing Mechanics

- Community Chest and Chance Cards are combined into a single deck called Fortune Cards.
- Neighborhoods are now called Towns.
- Houses and Hotels are now called Inns and Mansions.
- Railroads are now called Stables and Docks.
- Electric Company and Water Works are now called Weaponsmith and Armorsmith.
- Free Parking is now called Feast and it now has a function that is detailed in New Mechanics.
- Go to Jail and Jail squares are now called Go to Dungeon and Dungeon.

These replacements will be placed on the map the same way as their counterparts except the Fortune Cards deck. The squares that will make the players draw from the Fortune Cards deck will replace the old Chance squares, while the squares that will make the players draw from the Scrolls deck will replace the old Community Chest squares.

2.1.2 New Mechanics

The tokens are now classes with their own unique advantages:

- Traveler: Once every two turns, moves extra squares.
- Noble: Starts with extra money and pays less tax.
- Knight: Pays less rent to other players.
- Treasure Hunter: Once every three turns, draws a Fortune card at the start of their turn.

- Wizard: Fills Mana based on the number of squares travelled. Once Mana is full, draws a Scroll.
- Fortune Teller: When drawing a Fortune Card or Scroll, looks at the top 2 cards of the deck, picks one and shuffles the other one randomly back into the deck.
- Thief: Upon arriving at the same square as another player after rolling, steals some of their money. Escapes from the jail in one less turn.
- Builder: Erects buildings for cheaper.
- Cardinal: Builds Churches and Cathedrals instead of Inns and Mansions, which collect more rent than their counterparts.

Feast is a special square that has different effects for each class.

- Traveler: Moves extra squares for the next four turns.
- Noble: Earns money.
- Knight: Will evade the next rent he would have to pay after leaving Feast.
- Treasure Hunter: Draws a Scroll and a Fortune Card.
- Wizard: The amount of Mana regenerated will be permanently multiplied by a factor of 1.15, e.g. 1 --> 1.15 --> 1,3225 --> 1,520875.
- Fortune Teller: Steals a Scroll from a random player if possible.
- Thief: Steals from every player.
- Builder: Earns a certain amount of money for each Inn they built. Earns triple this amount for each Mansion they built.
- Cardinal: Randomly teleports to one of his towns that has a Church or a Cathedral if possible.

Scrolls are a new deck of cards that can be kept in the player's hand after drawn and used anytime during one of their turns. They can have effects that can change the course of the game drastically unlike Fortune Cards. Each player begins the game with two scrolls.

2.1.3 Gameplay

Each player starts at the same square. Two dice are used for moving tokens on the board and putting in order player turns. The turn order will be set randomly. Each player then rolls the two dice and moves a number of squares based on the result in their turns. If a player arrives at a property that hasn't been purchased, they can purchase this property to make it theirs. Any other player who arrives at this property has to pay rent which will increase depending on the number of Inns/Churches and Mansions/Cathedrals if the property in question is a town. It will also increase based on the number of properties of the same type owned by the player if the property in question is a Stable, Dock, Armorsmith or Weaponsmith. The players can trade with each other to buy or sell property. A player loses when they have no money left. When a player loses, all of their properties lose their ownership and can be bought by the other players again. If the bankruptcy was caused by rent, the bank will pay the rent instead of the bankrupt player. The last player remaining in the game wins.

2.2 Functional Requirements

2.2.1 Initializing a Game

- The System should allow initializing a new game with a unique ID.
- The System should allow an interested group of players to join using the game's unique ID.
- The System should be able to provide a chatbox where the users can communicate.
- The System should allow the players to select a class.

- The System should distribute money and Scrolls to each player based on the classes.
- The System should set the order of turns randomly.

2.2.2 Playing the Game

- The System should allow the players to roll a pair of dice on their turn.
- The System should move the player's token based on the result of their dice roll.
- The System should distinguish the type of squares on the map i.e. whether the square is a property square, a Fortune Card square, a Tax square, a Scroll square etc.
- The System should keep track of each property square's ownership, i.e. if it is bought and whom it belongs to.
- The System should allow the player to draw from a deck, purchase a property or pay to the bank or to a player based on the square they reach after moving.
- The System should keep track of each player's position, properties, money and Scrolls.
- The System should allow players to send trade requests.
- The System should allow the players to accept or decline trade requests.
- The System should allow the player to end their turn.
- The System should keep track of the remaining time until a turn ends.
- The System should acknowledge the end of a turn.

- The System should remove the bankrupt players from the turn cycle.
- The System should return the bankrupt players' possessions to the bank and the card decks.
- The System should make the bank pay a bankrupt player's rent if the bankruptcy was caused by the inability to pay rent to another player.
- The System should announce the last remaining player as the winner and end the game.
- The System should play music and sound effects throughout the game.

2.2.3 Main Menu

- The System should display the credits.
- The System should a tutorial on how to play the game.
- The System should allow the user to exit the game.
- The System should allow the user to configure the game's settings.

2.3 Non-functional Requirements

2.3.1 Performance Characteristics

- Response time of the system should not exceed 1 second.
- There shouldn't be any data loss between the server-client communications.

2.3.2 User Interface

- A player should be able to read the current state of the game in under 10 seconds.

- The game should accept inputs from mouse and keyboard.
- Users should be restricted from making illegal moves.

2.3.3 Error Handling and Extreme Conditions

- The System should not allow more than the maximum allowed number of players into a game room.
- The System should display an error message if a user tries to enter an invalid game ID while joining a game.

2.3.4 Quality Issues

- There shouldn't be any downtime in the game's services.

2.3.5 Security Issues

- A player shouldn't be able to see another player's Scrolls unless it is intended, i.e. Fortune Teller stealing scrolls.

2.3.6 Supportability

- A player shouldn't be able to see another player's Scrolls unless it is intended, i.e. Fortune Teller stealing scrolls.

2.4 Pseudo Requirements

1. The game will be made in Java
2. The game's trailer will be made in Adobe Premier Pro
3. The game's soundtrack will be made in MuseScore 3
4. GitHub will be used for version control during development.

2.5 System Models

2.5.1 Use-Case Model

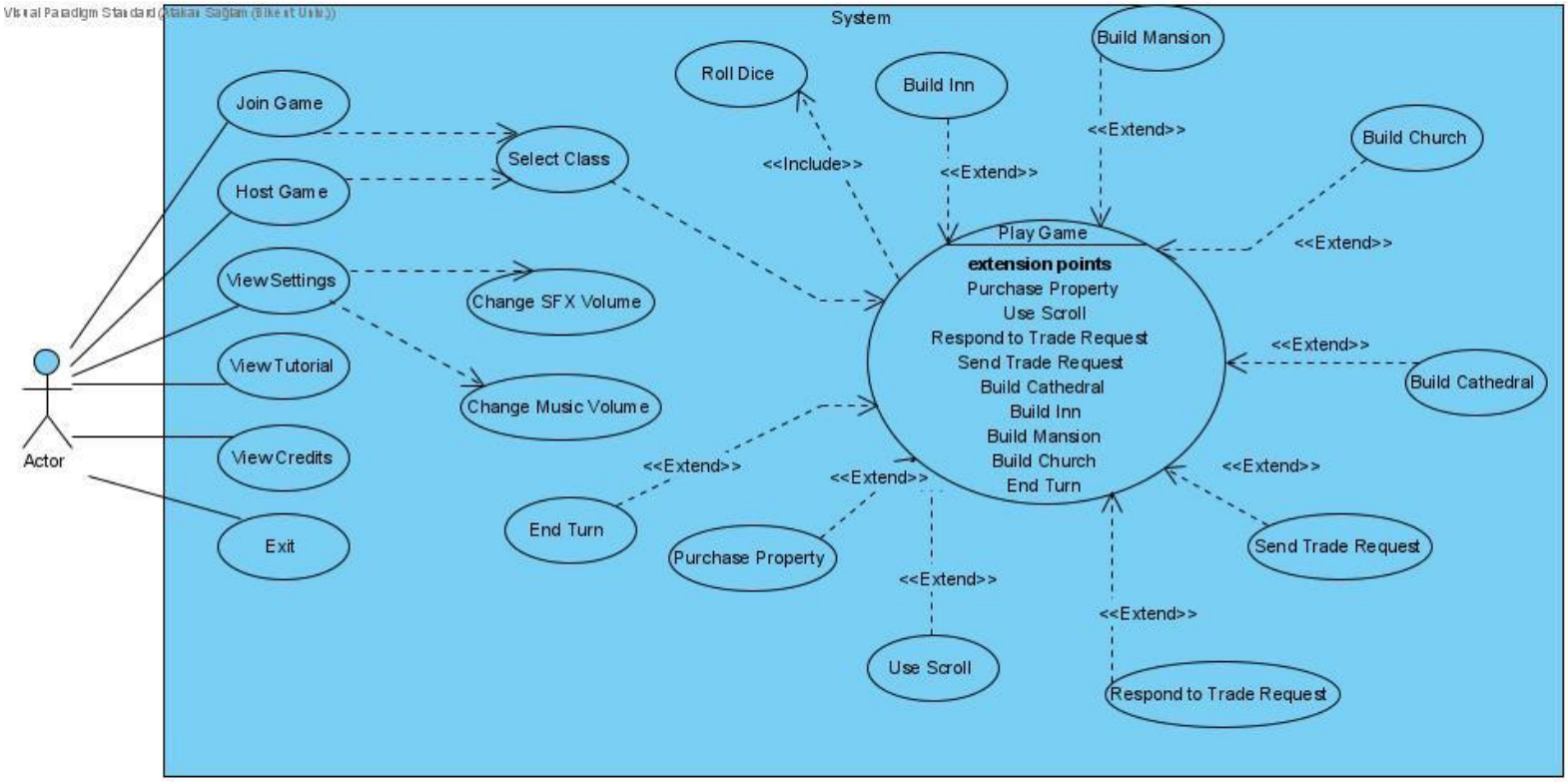

Figure 1 Use-Case Diagram

2.5.2 Object and Class Model

Figure 2 Object and Class Diagram

2.5.3 Dynamic Models

2.5.3.1 State Diagrams

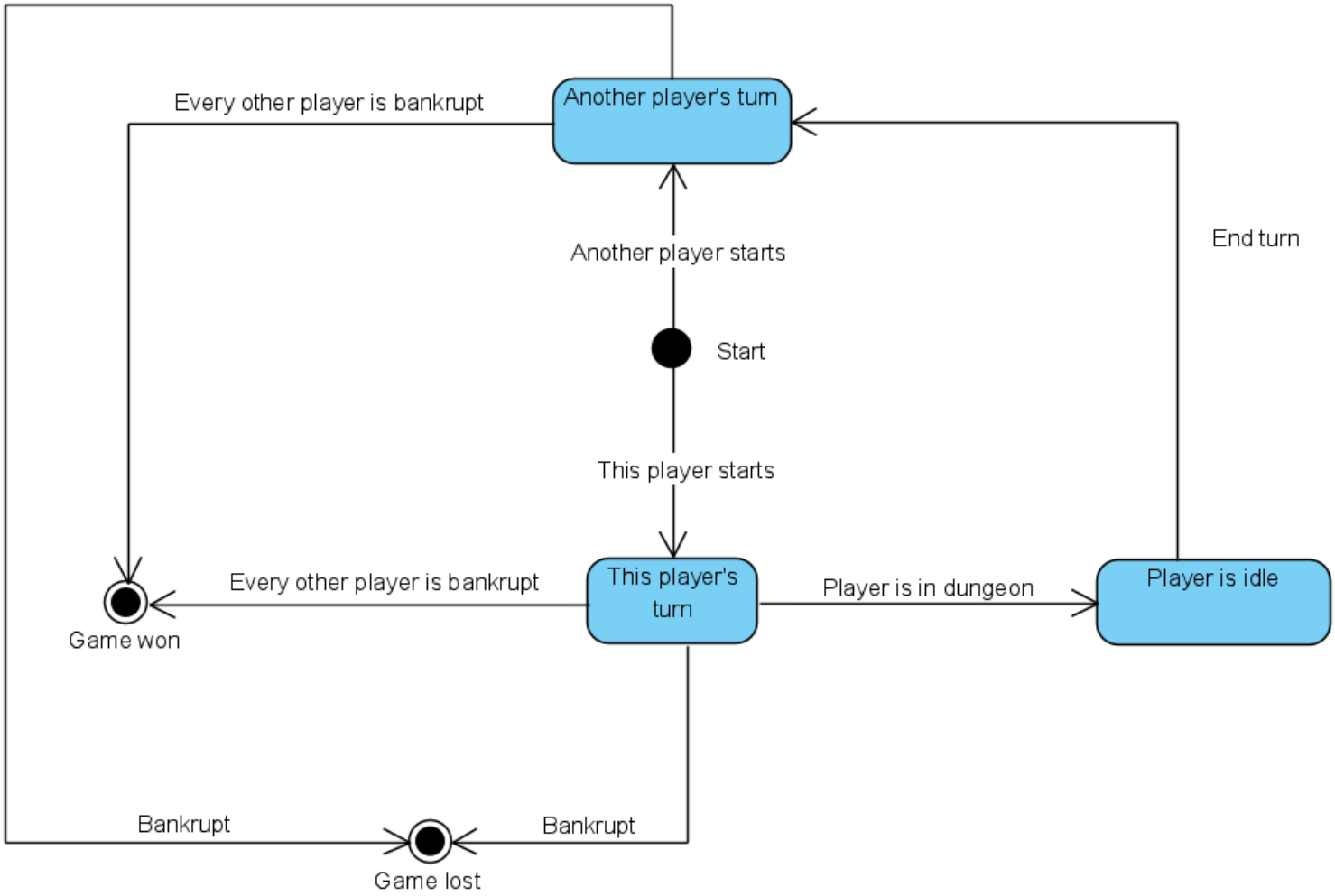

Figure 3 State Diagram for main turn loop

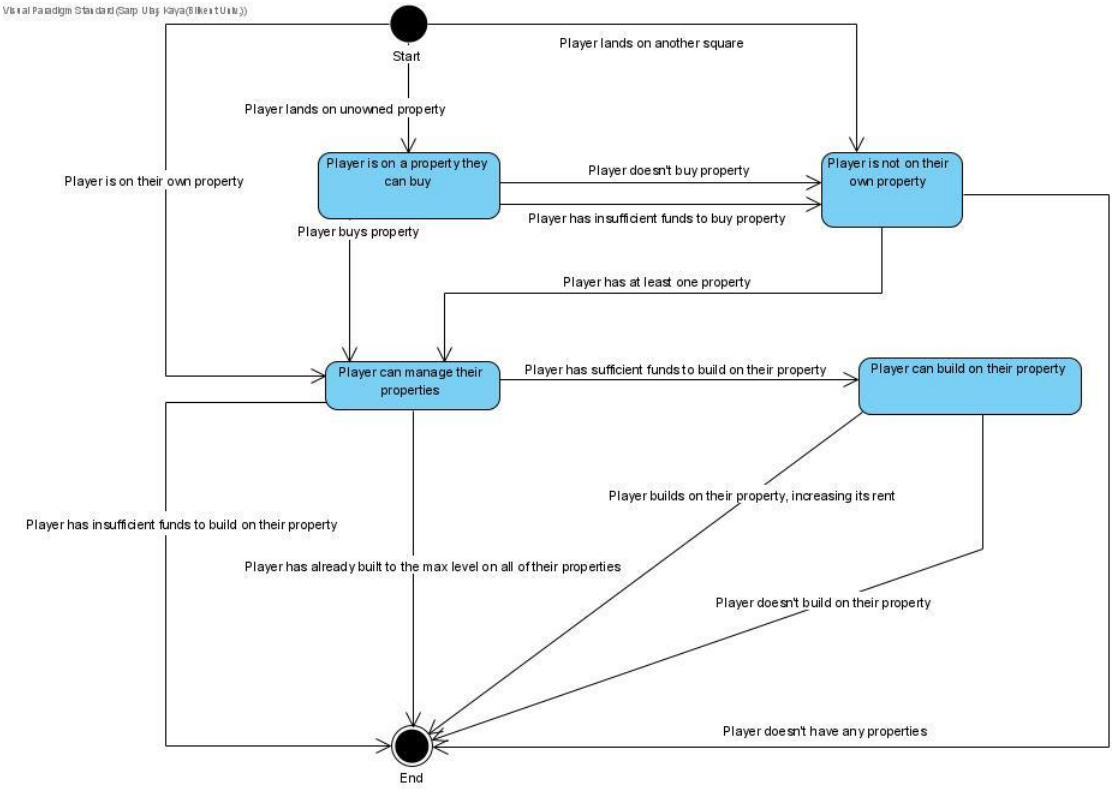

Figure 4 State Diagram for property management

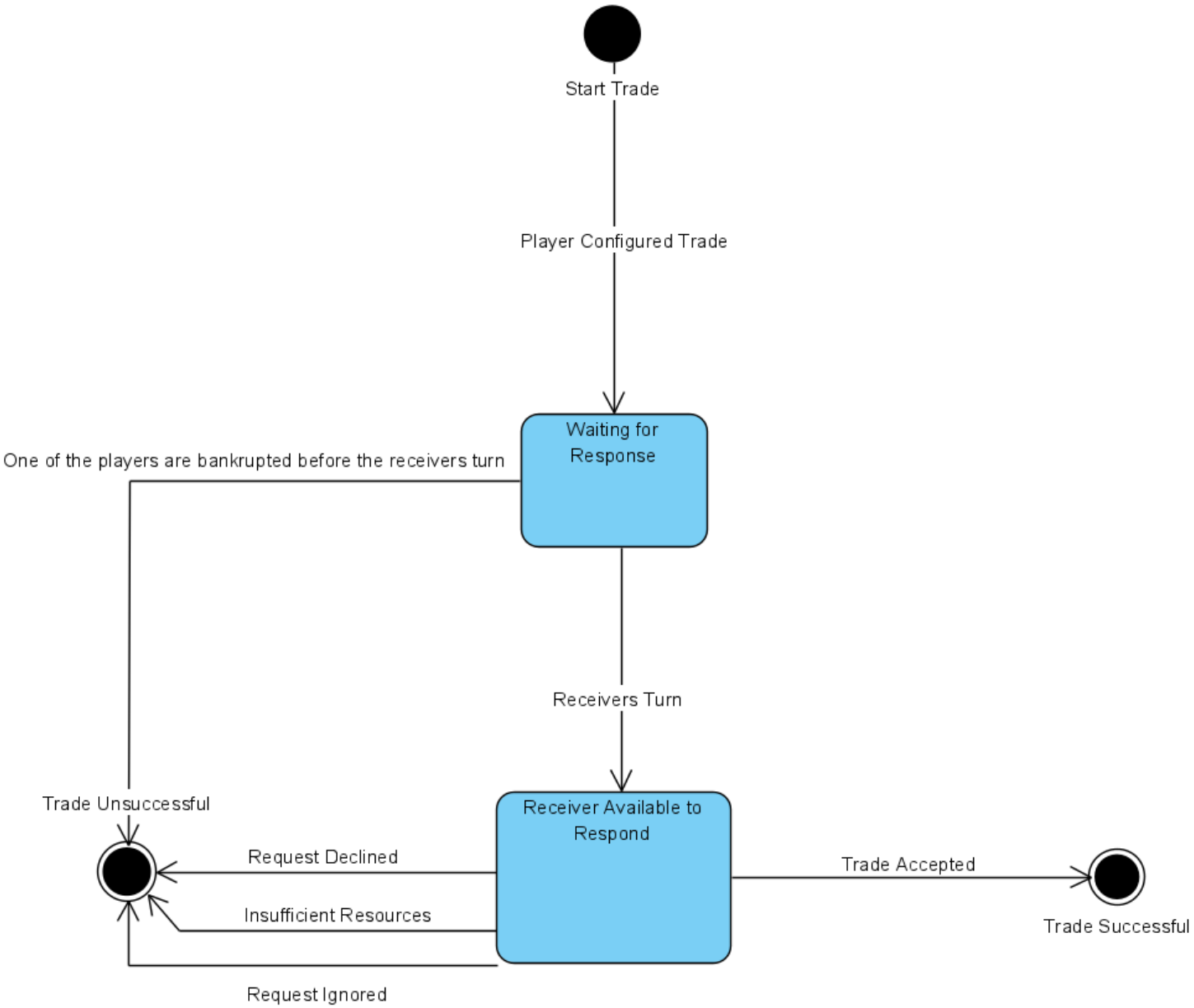

Figure 5 State Diagram for trade operation

2.5.3.2 Sequence Diagrams

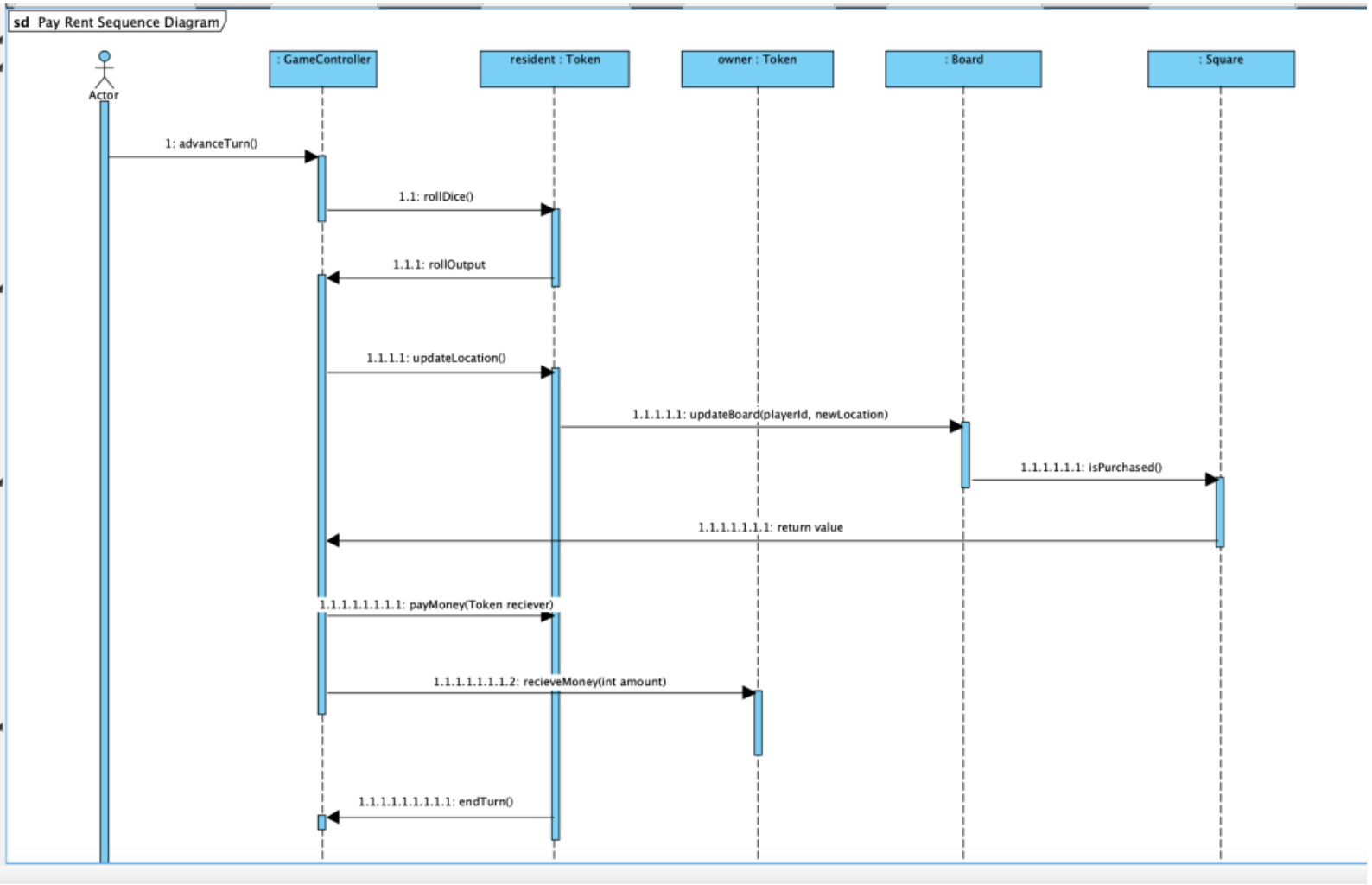

Figure 6 Pay Rent Sequence Diagram

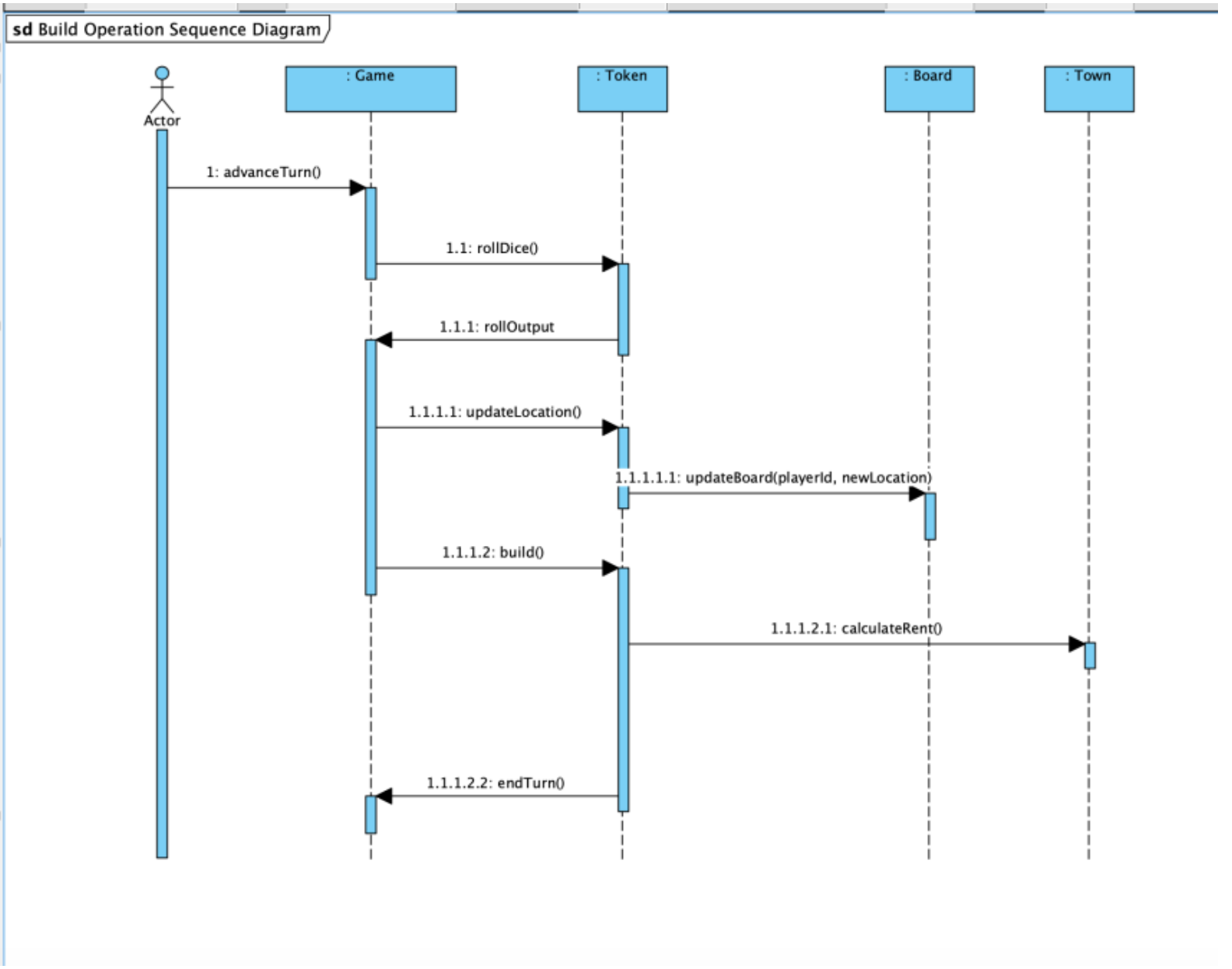

Figure 7 Build Operation Sequence Diagram

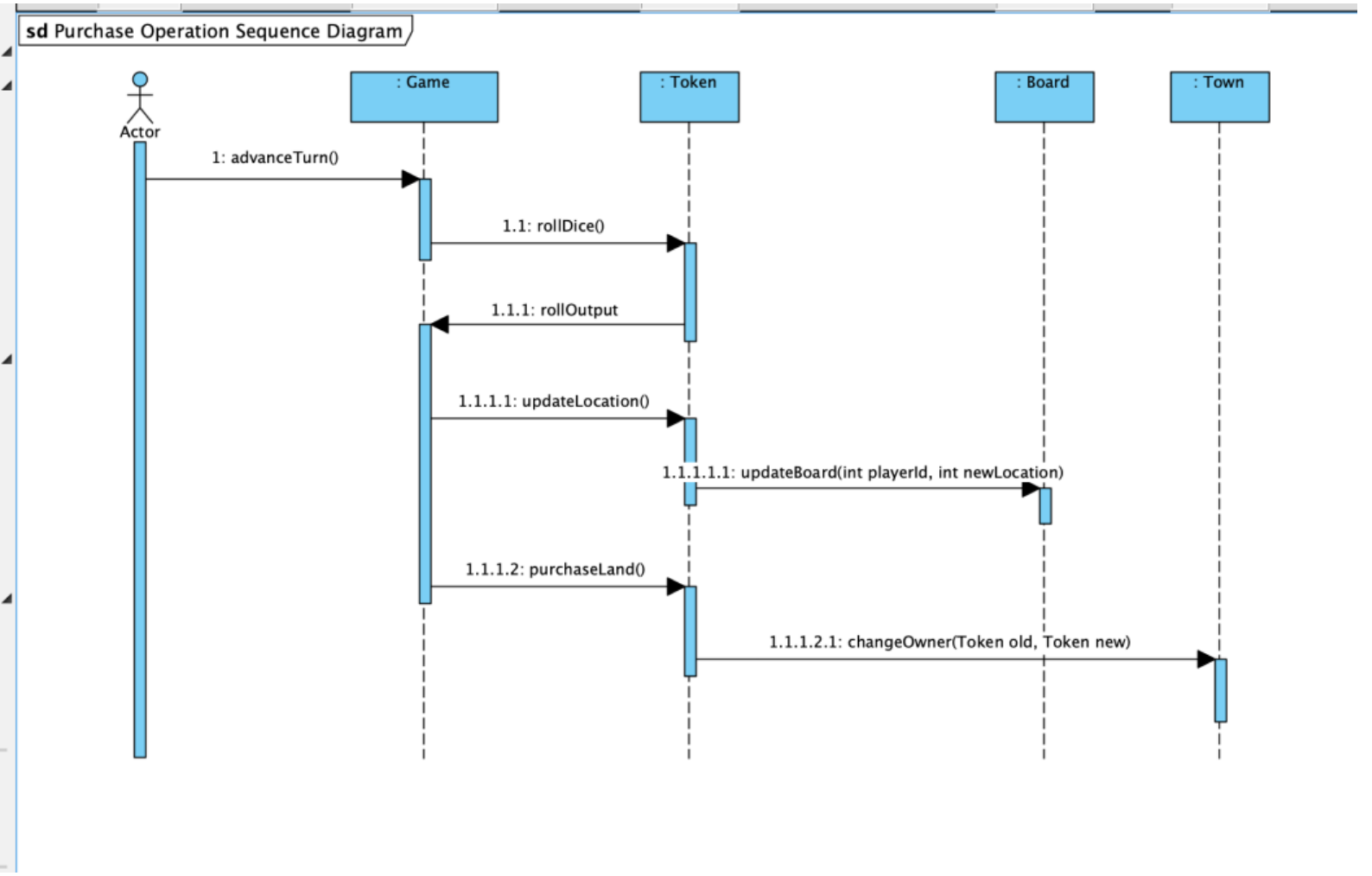

Figure 8 Purchase Operation Sequence Diagram

2.5.3.3 Activity Diagram

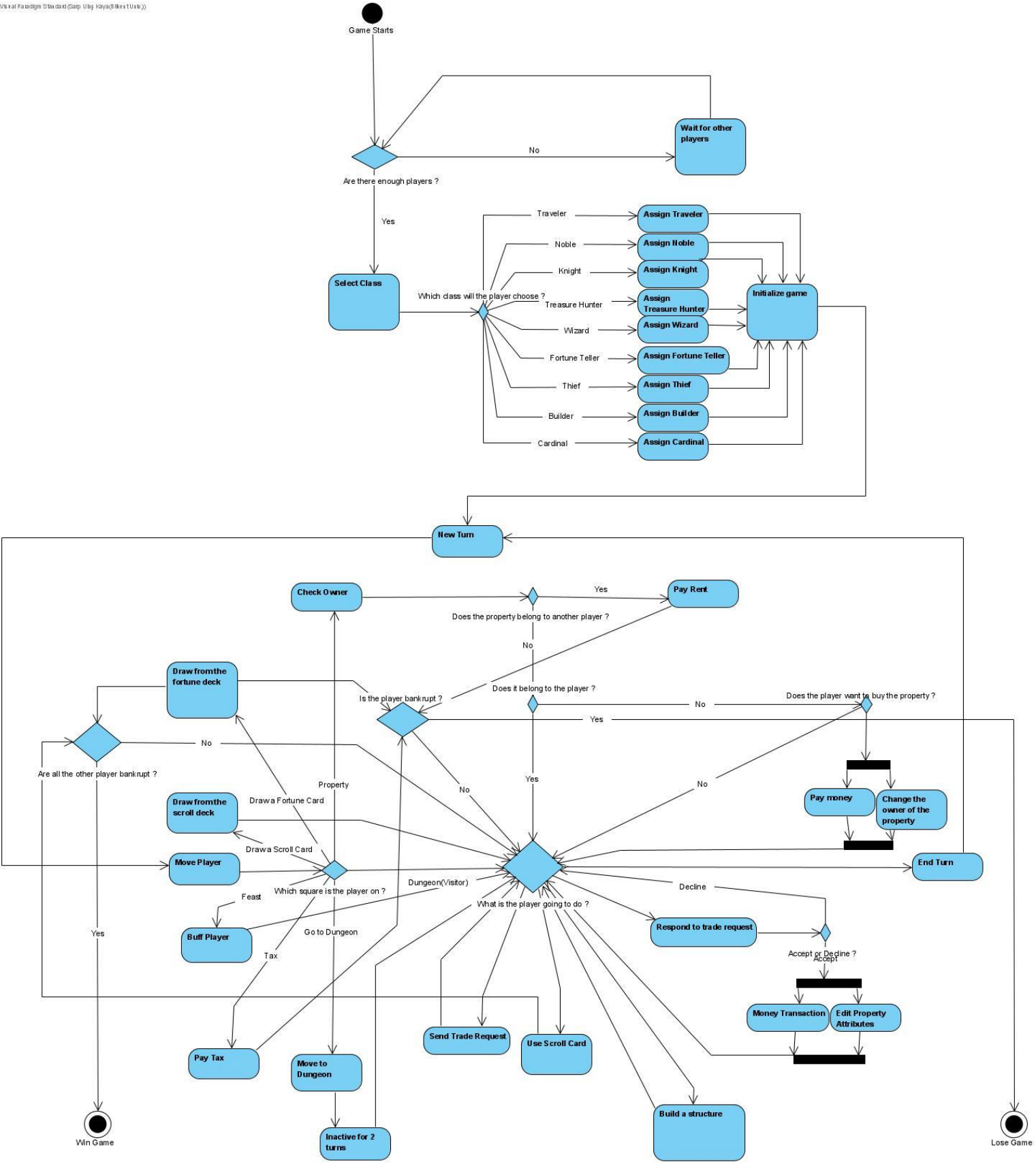

Figure 9 Activity Diagram

2.5.4 User Interface

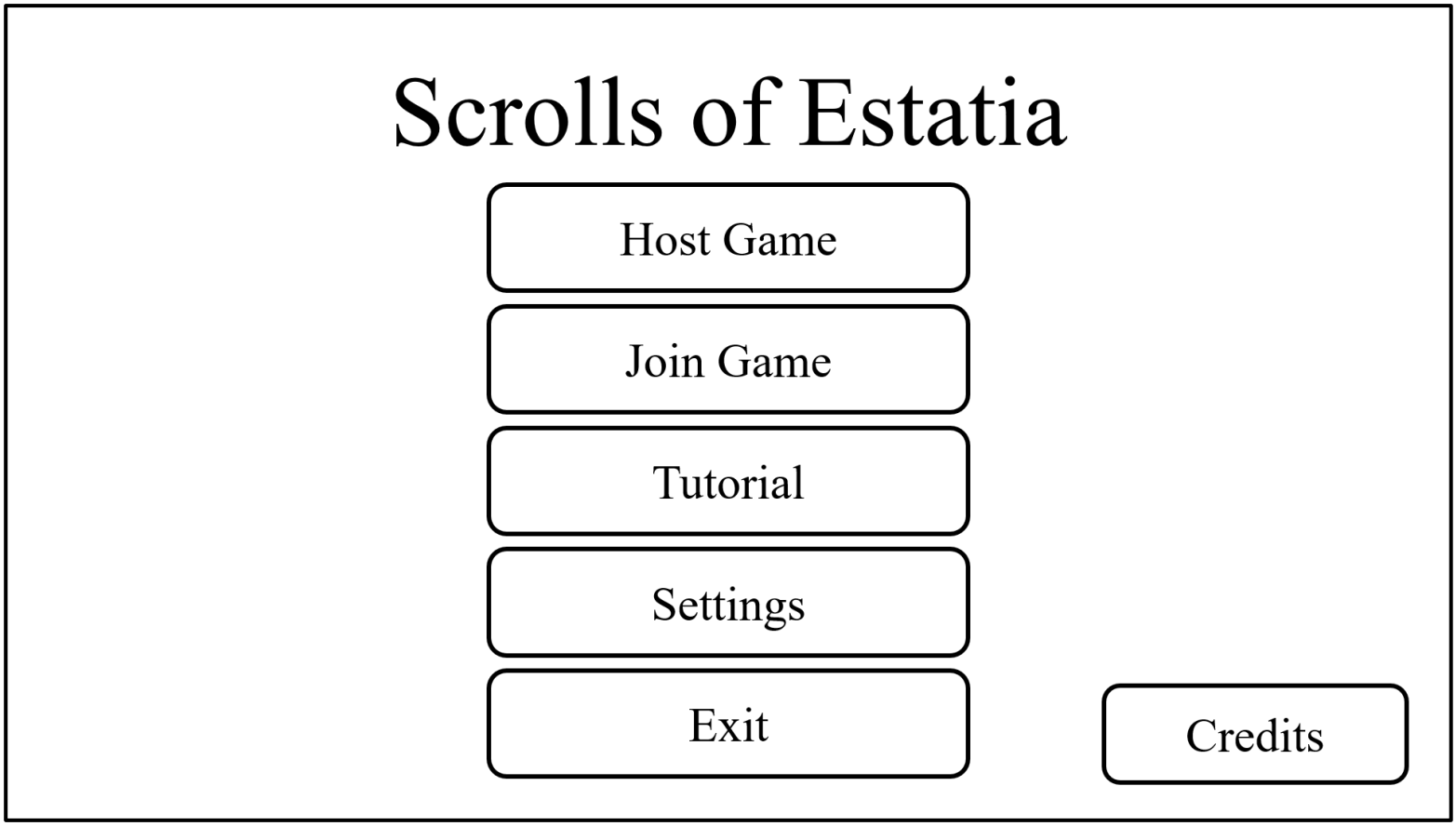

Figure 10 Main Menu

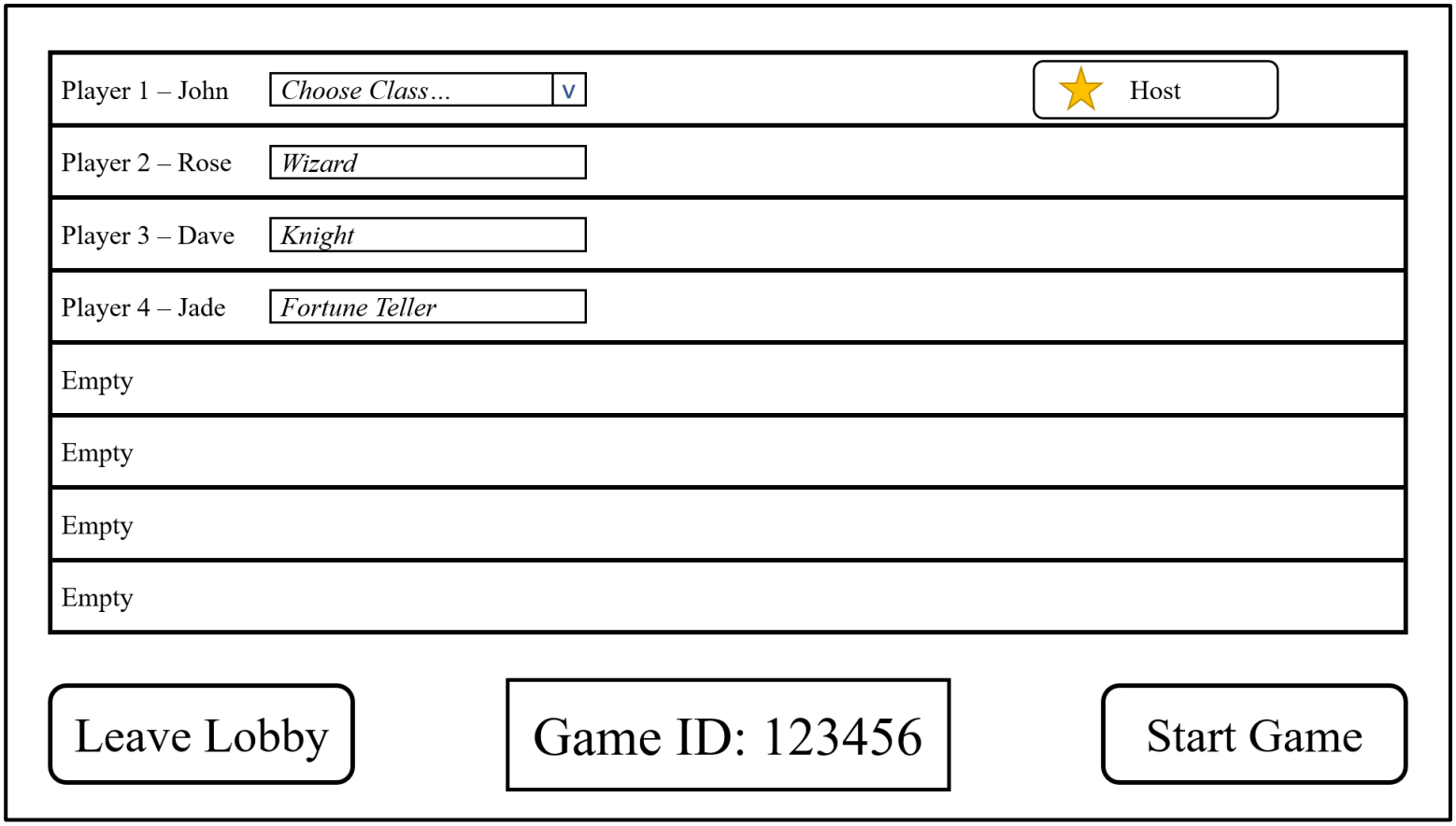

Figure 11 Host Lobby Screen

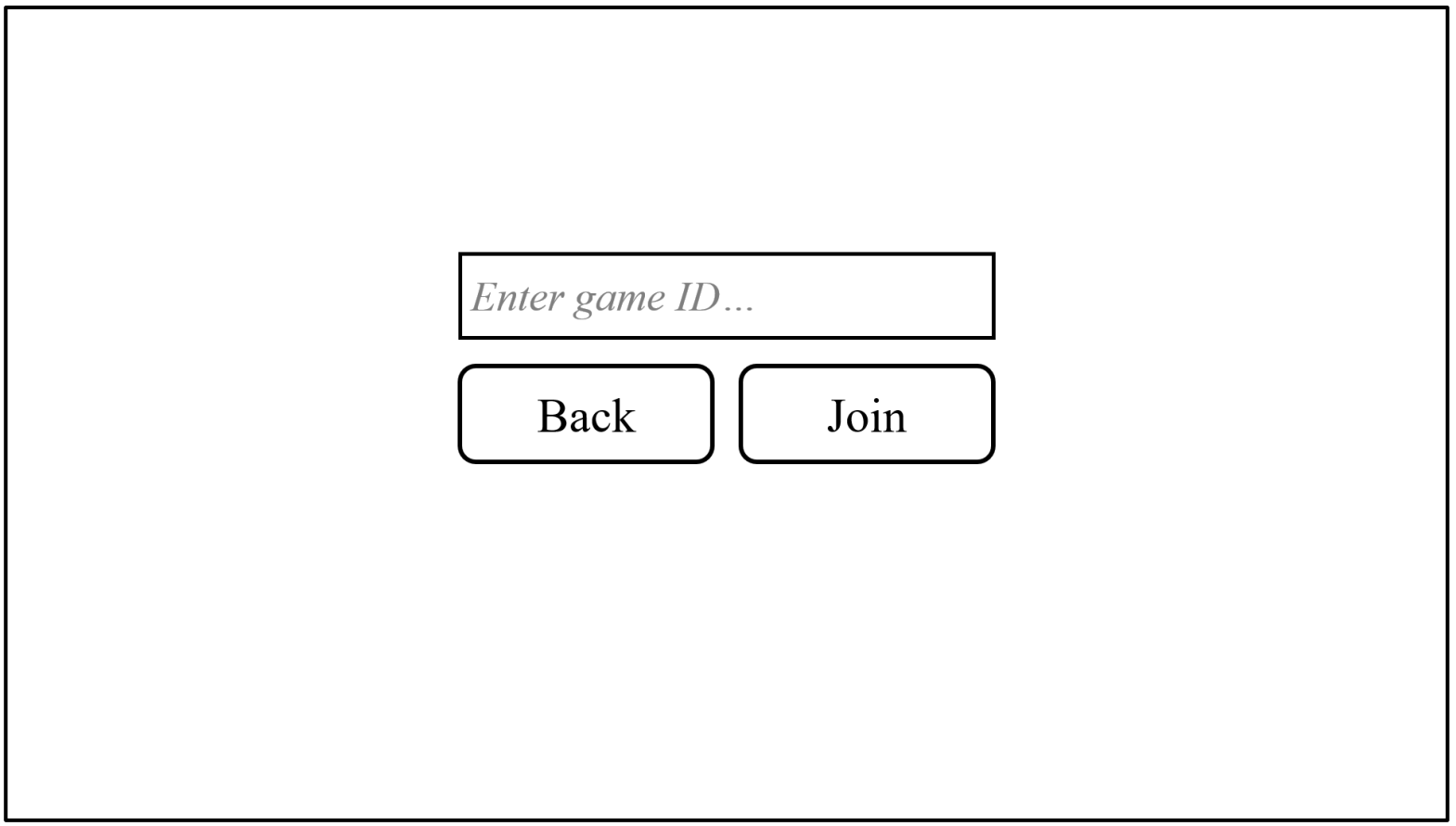

Figure 12 Join Game Screen

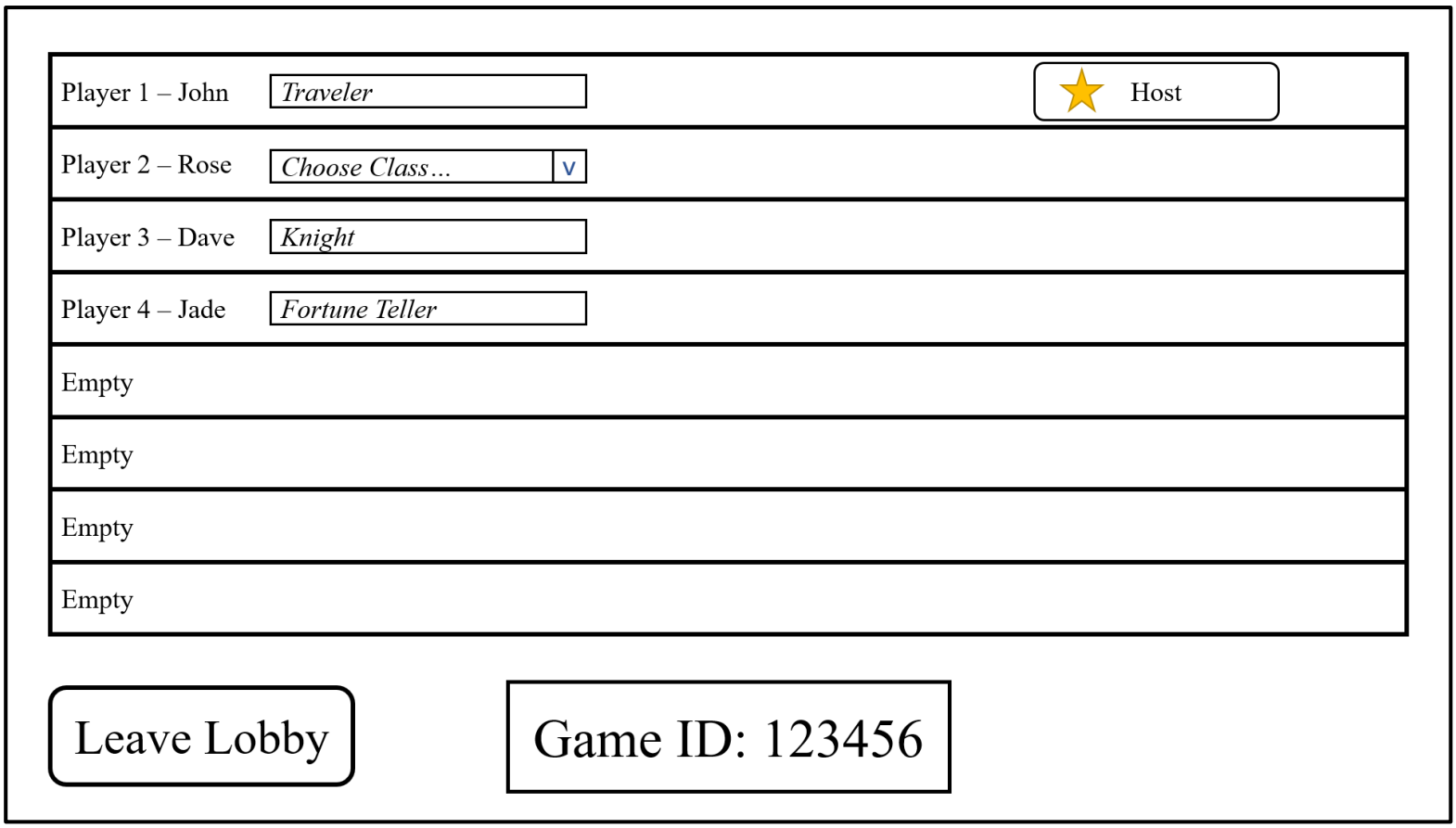

Figure 13 Non-Host Lobby Screen

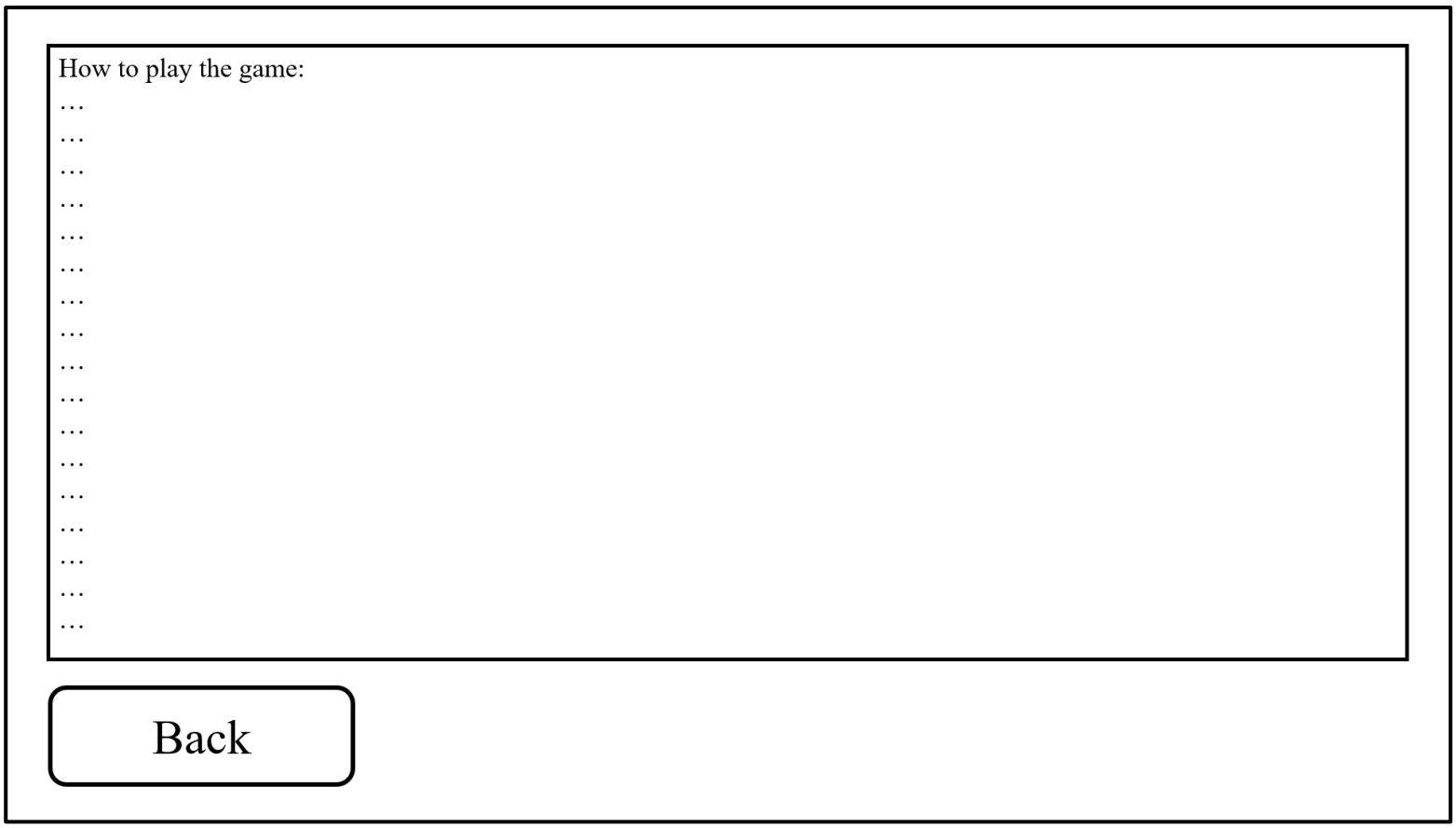

Figure 14 Tutorial Screen

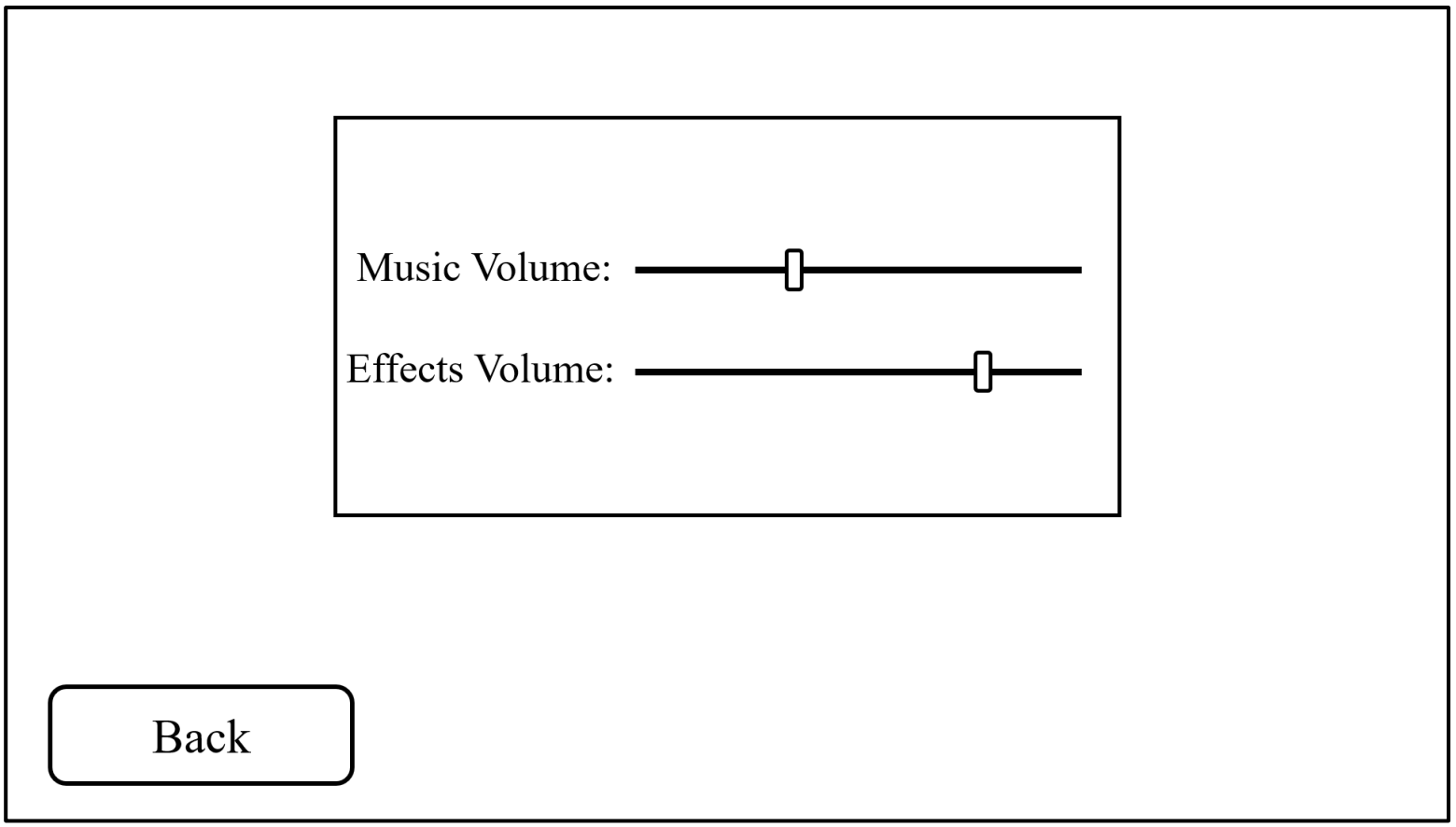

Figure 15 Settings Screen

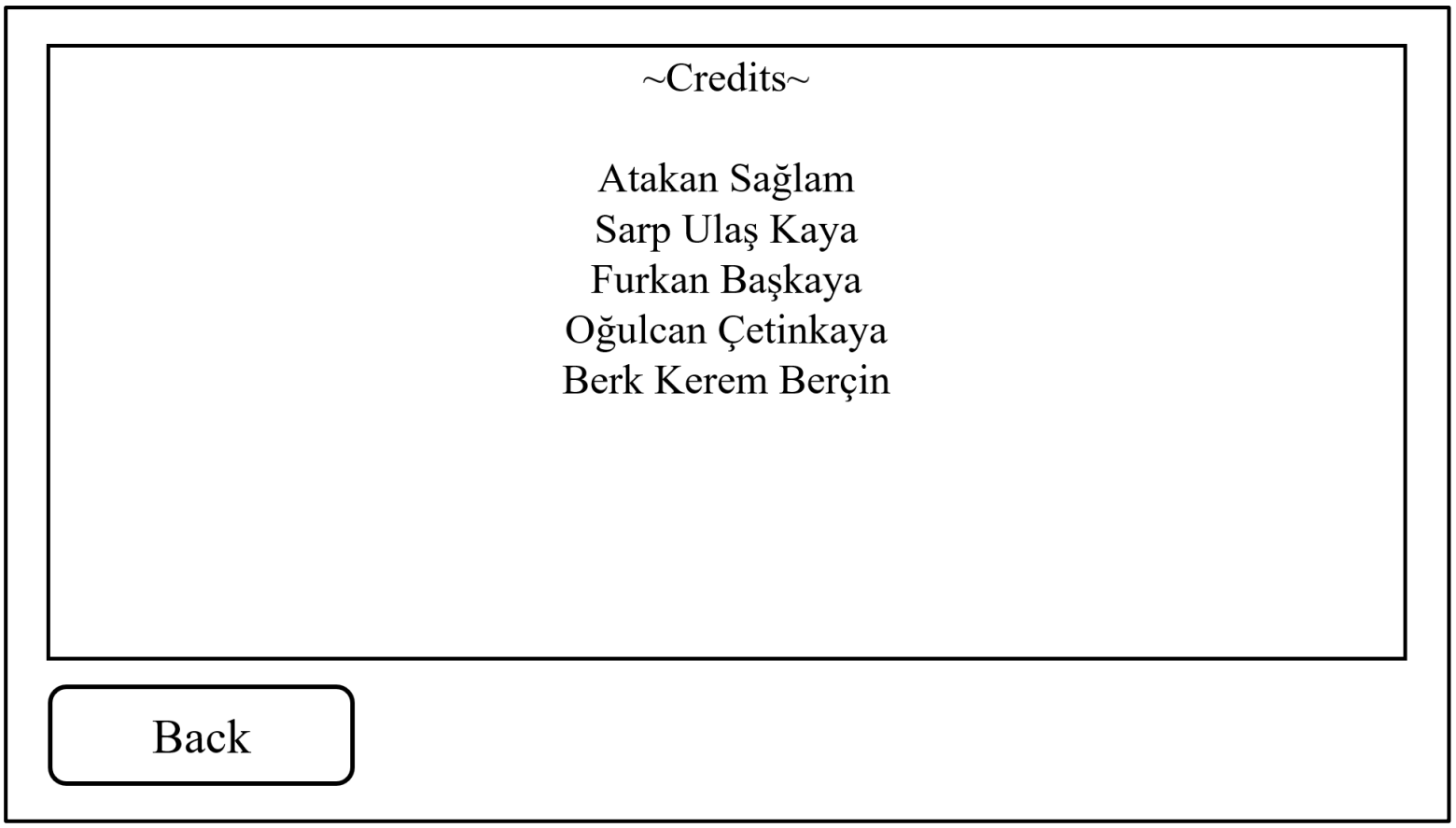

Figure 16 Credits Screen

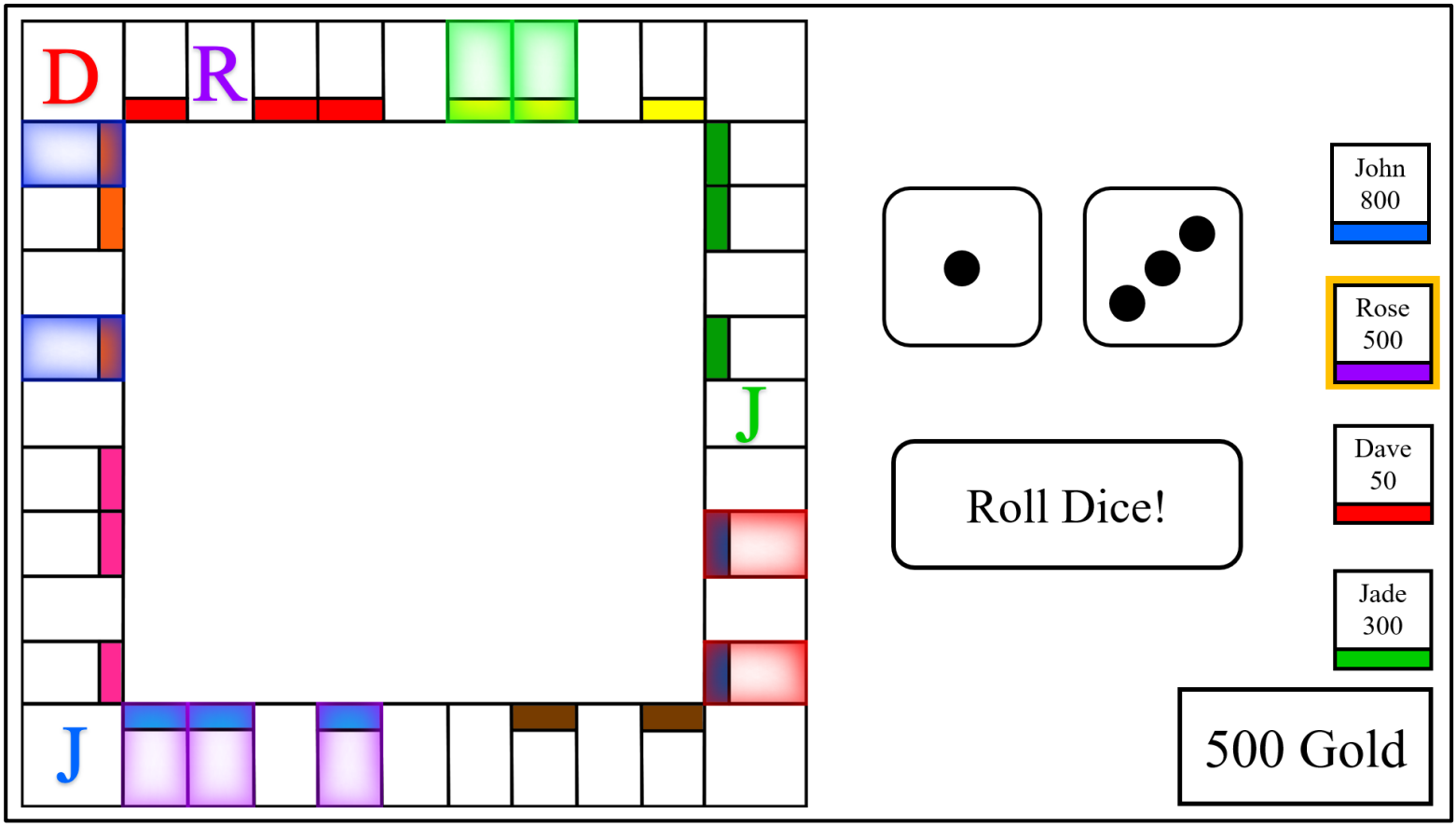

Figure 17 Game Screen When Throwing Dice
(Board Template from Pinterest [1])

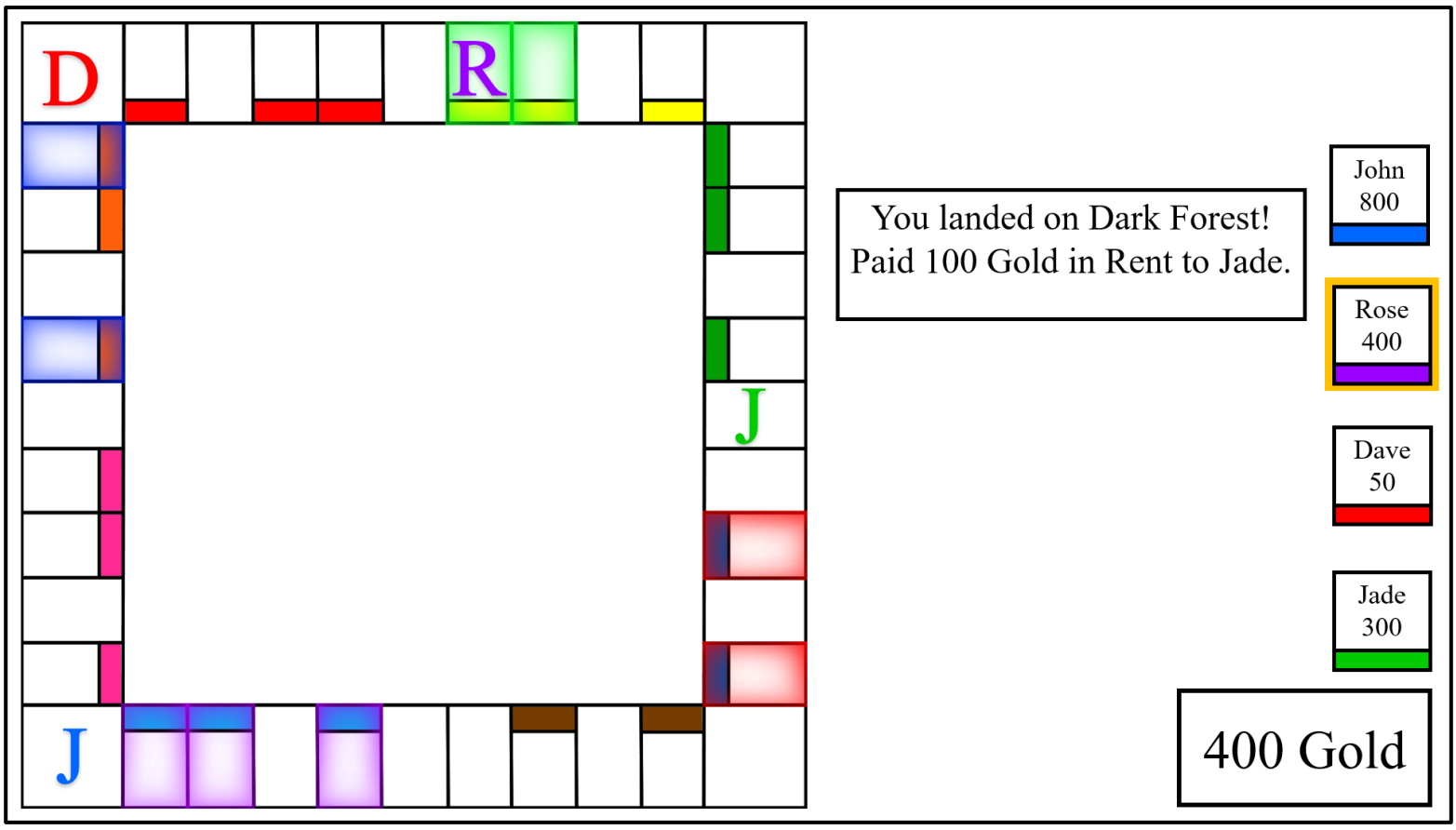

Figure 18 Game Screen When Paying Rent

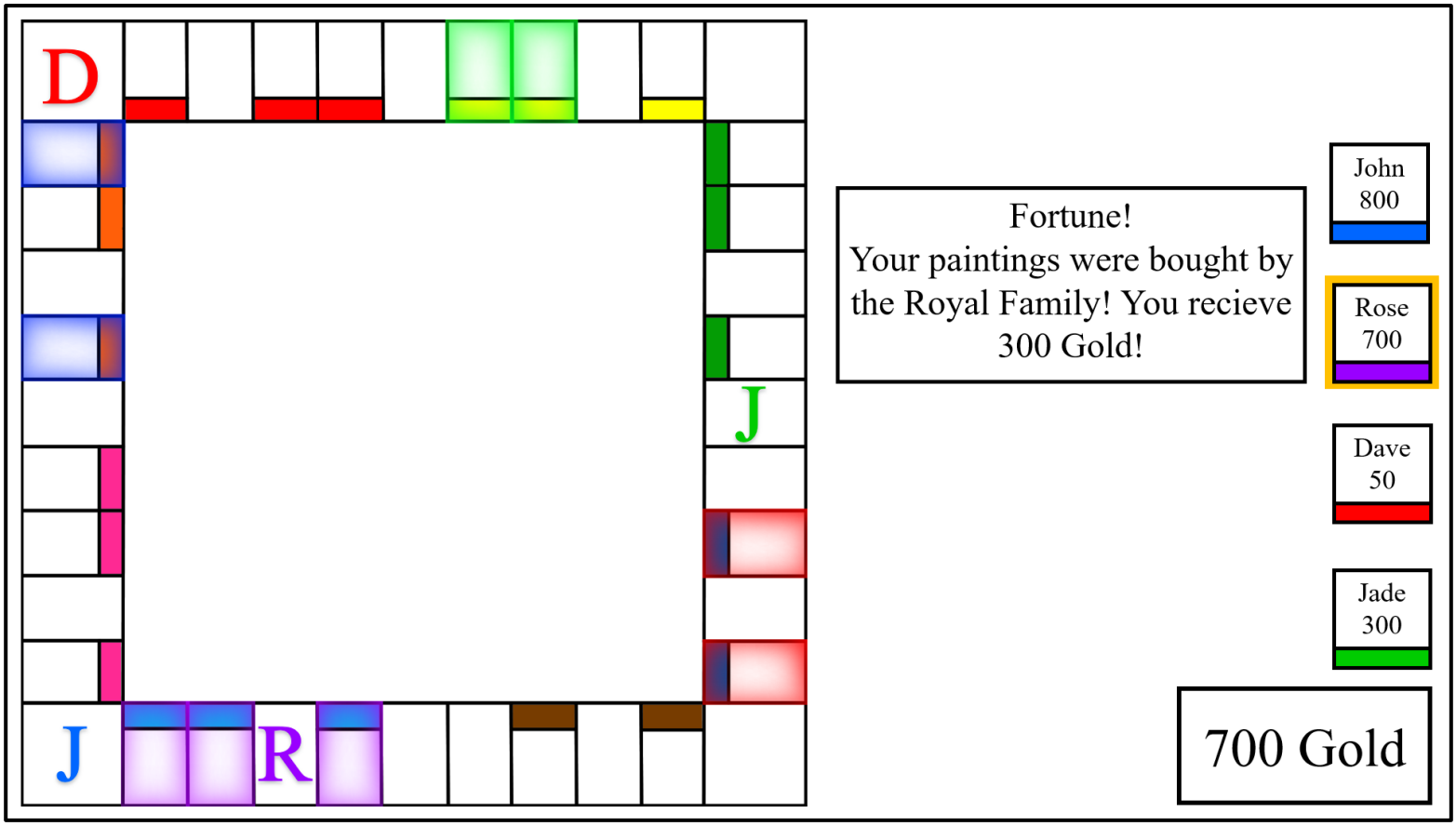

Figure 19 Game Screen When Drawing Fortune Card

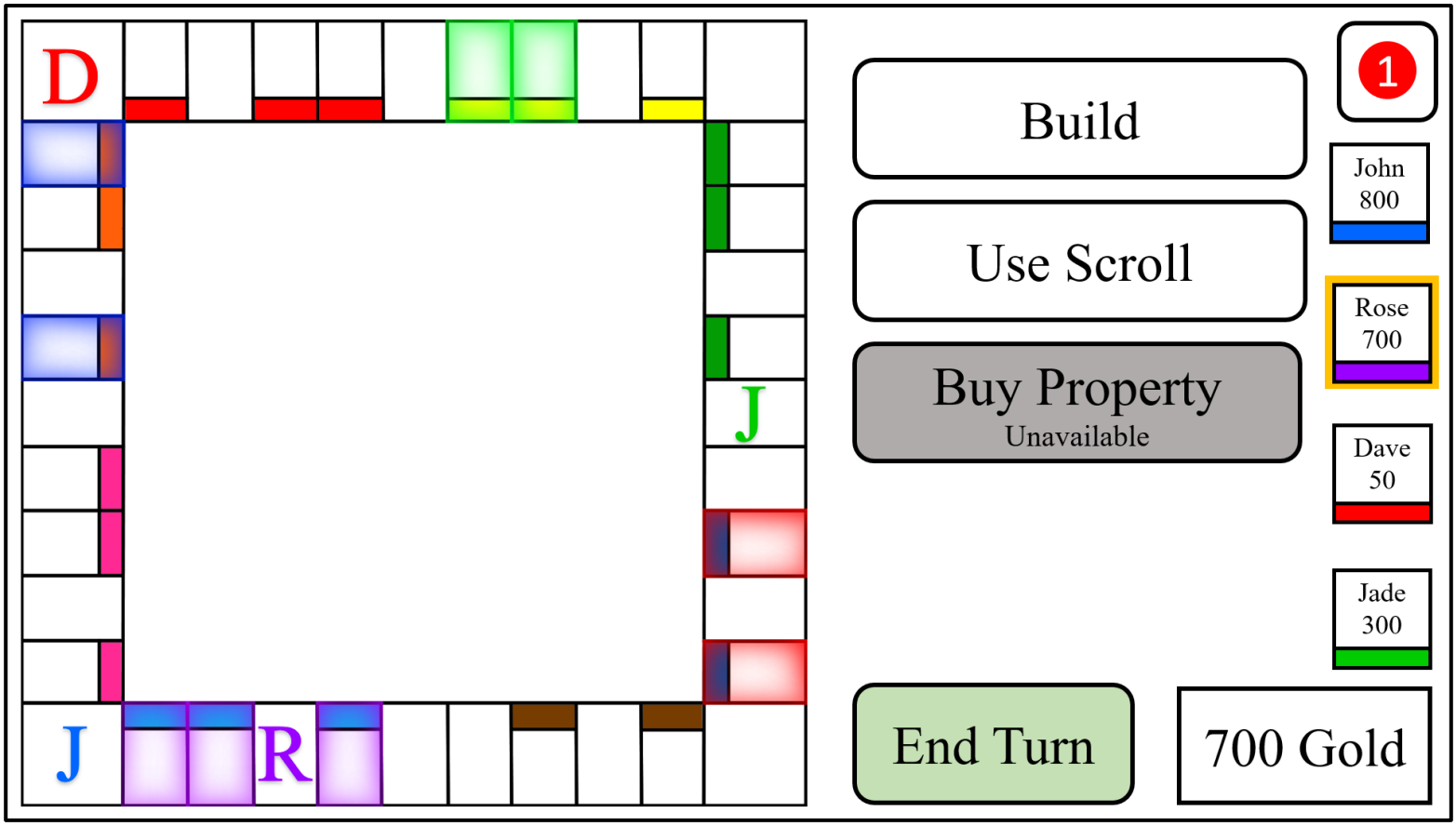

Figure 20 Game Screen When Landed on Non-Purchasable Square

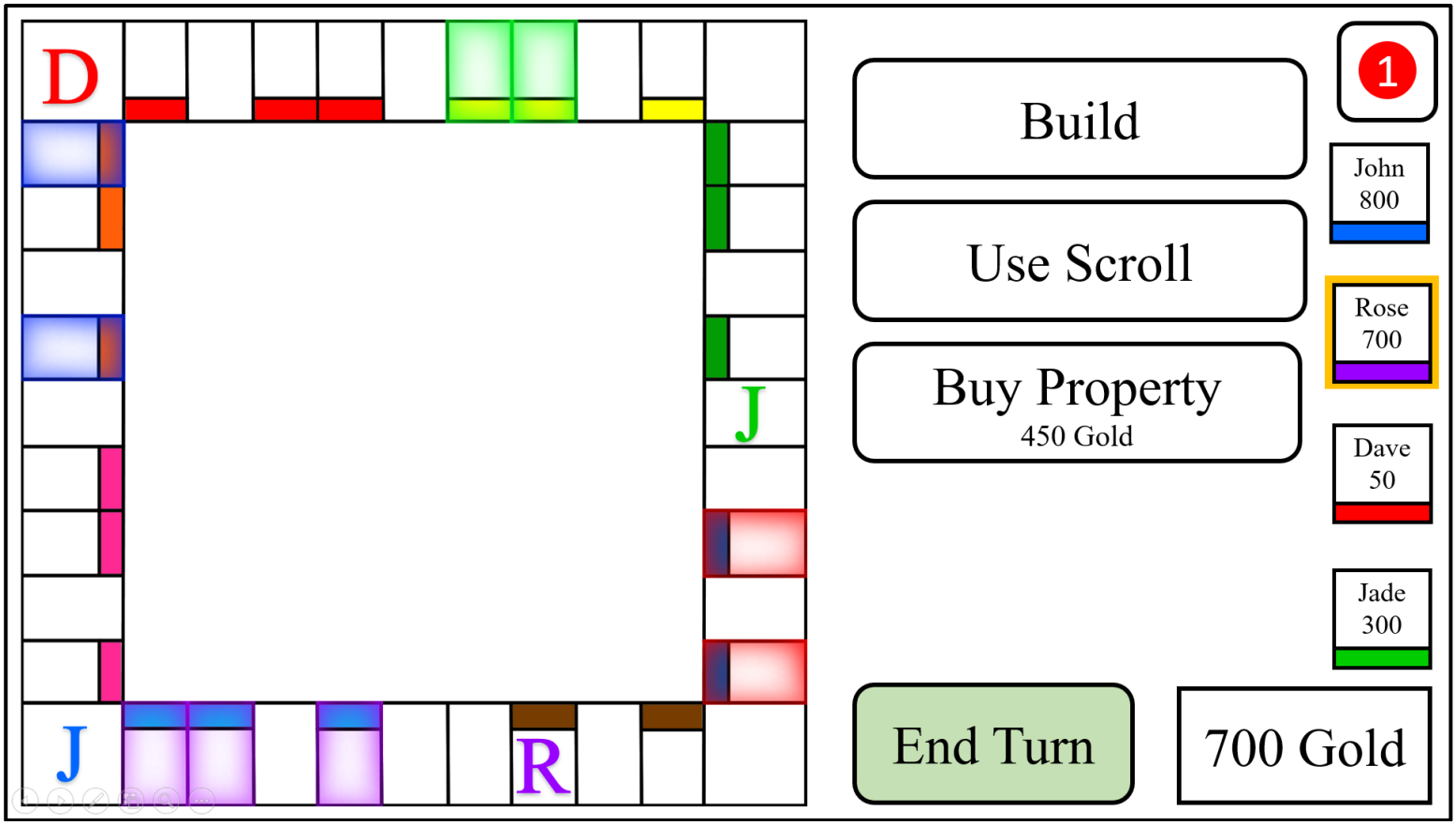

Figure 21 Game Screen When Landed on Purchasable Square

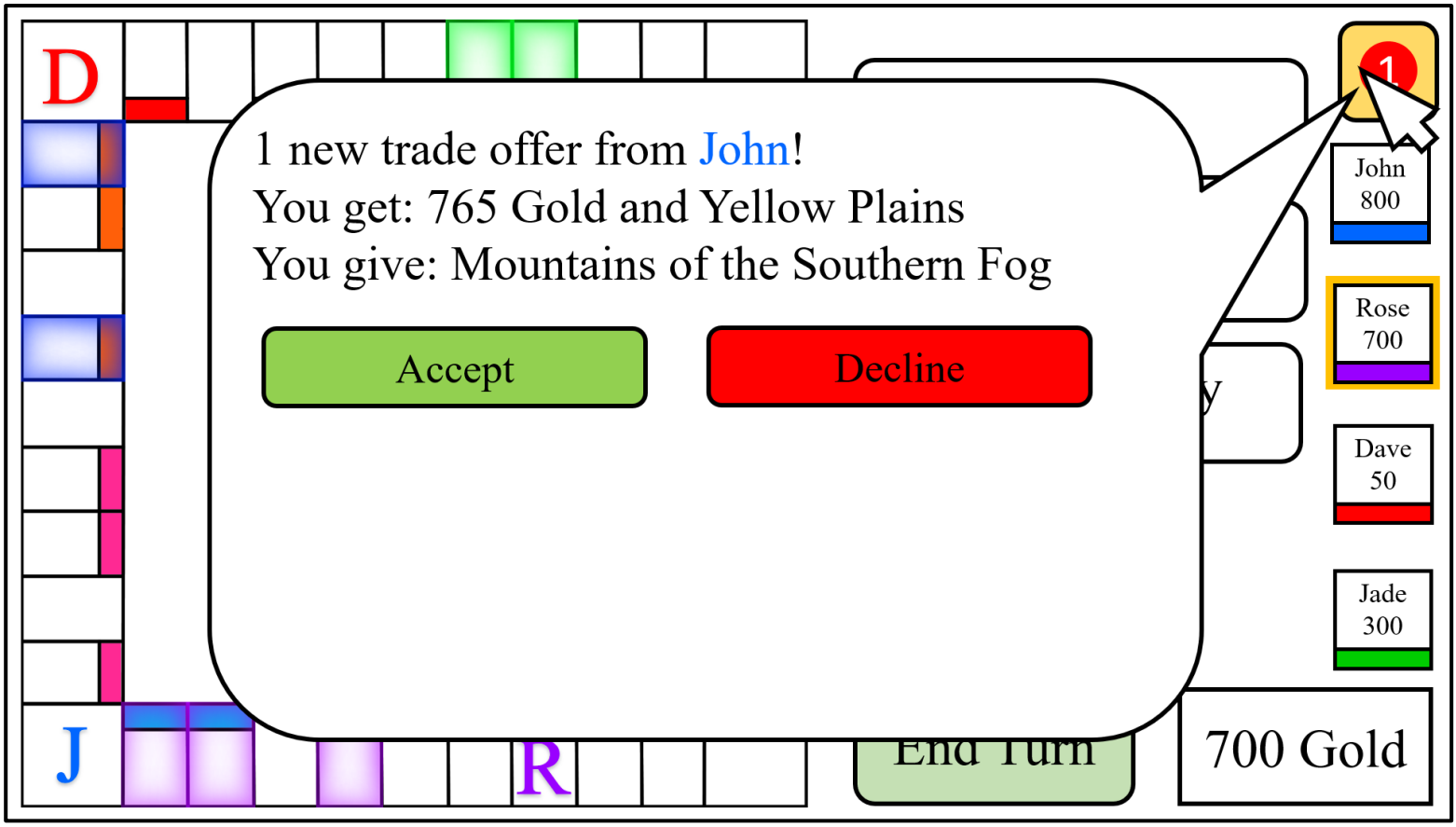

Figure 22 Game Screen When Reviewing Incoming Trade Request

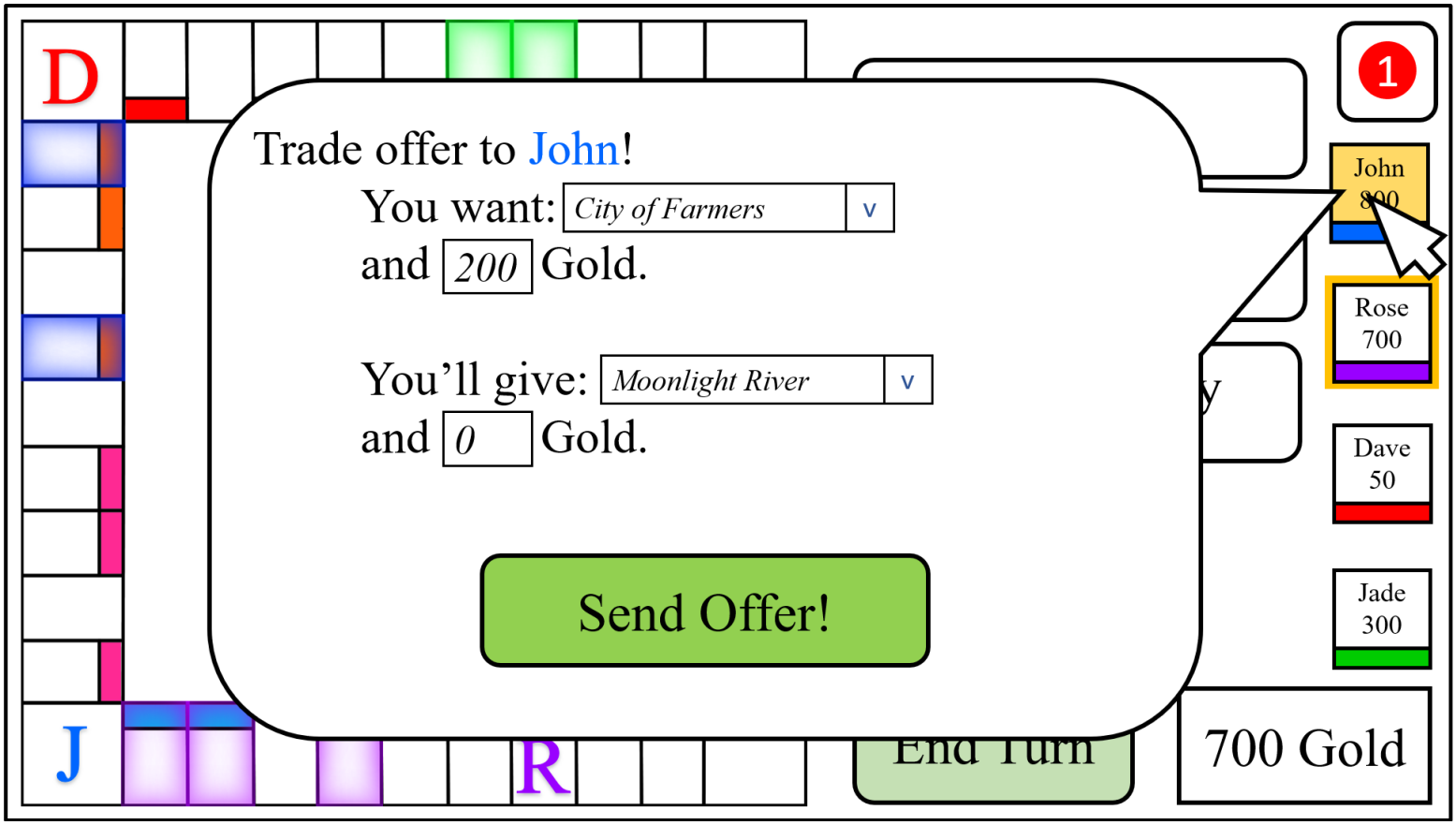

Figure 23 Game Screen When Sending Trade Offer

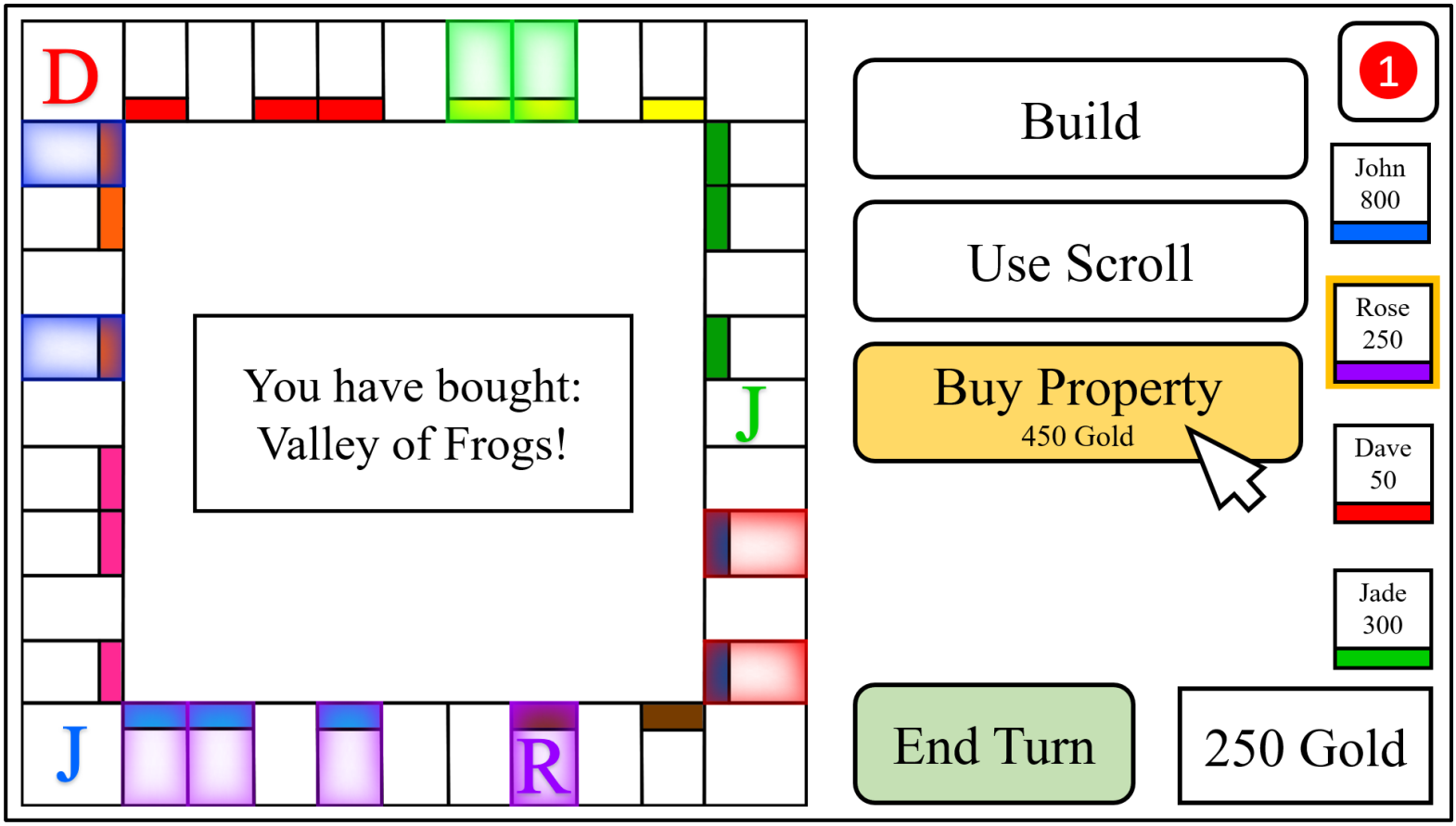

Figure 24 Game Screen When Buying Property

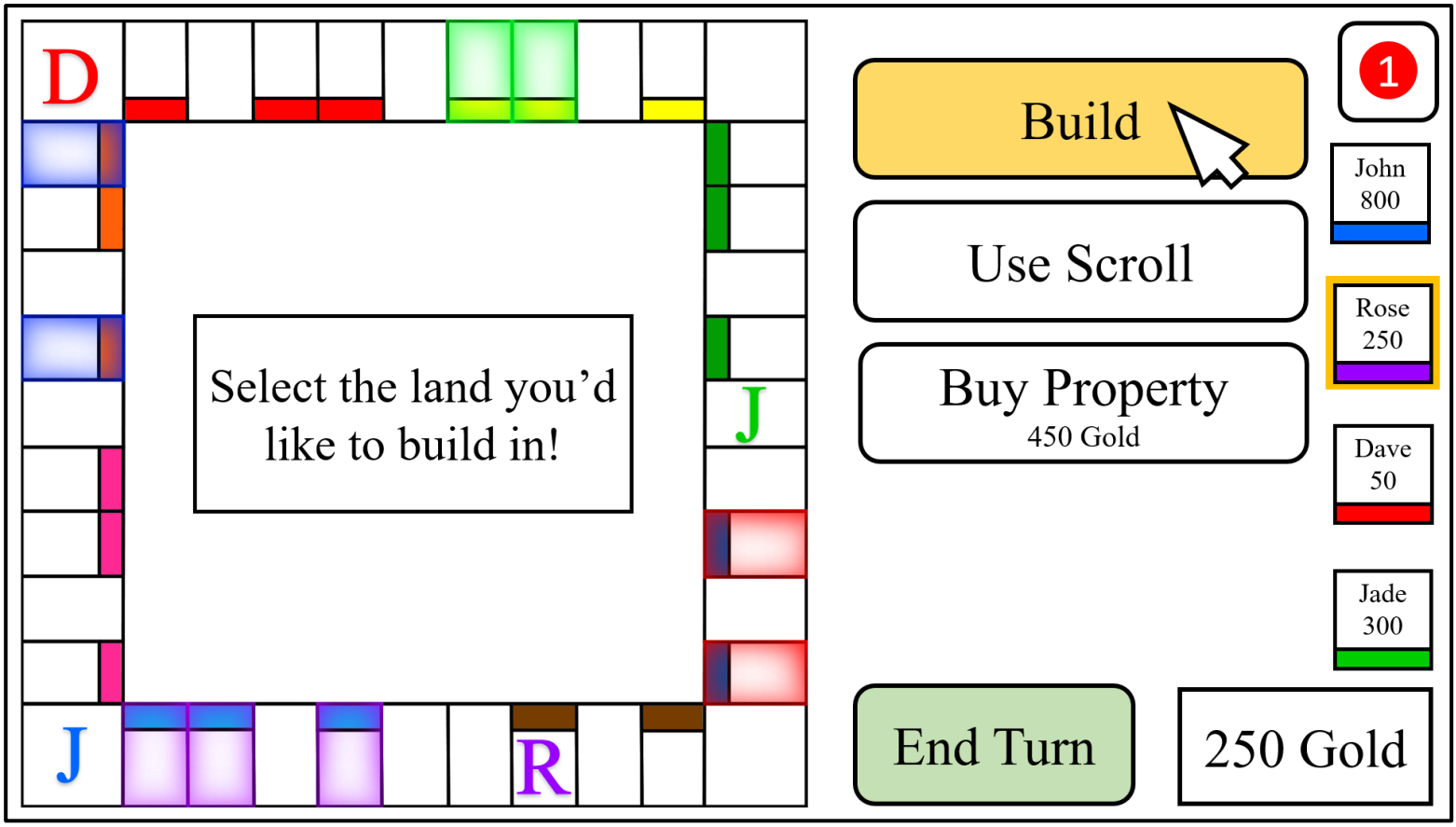

Figure 25 Game Screen When Entering Build Mode

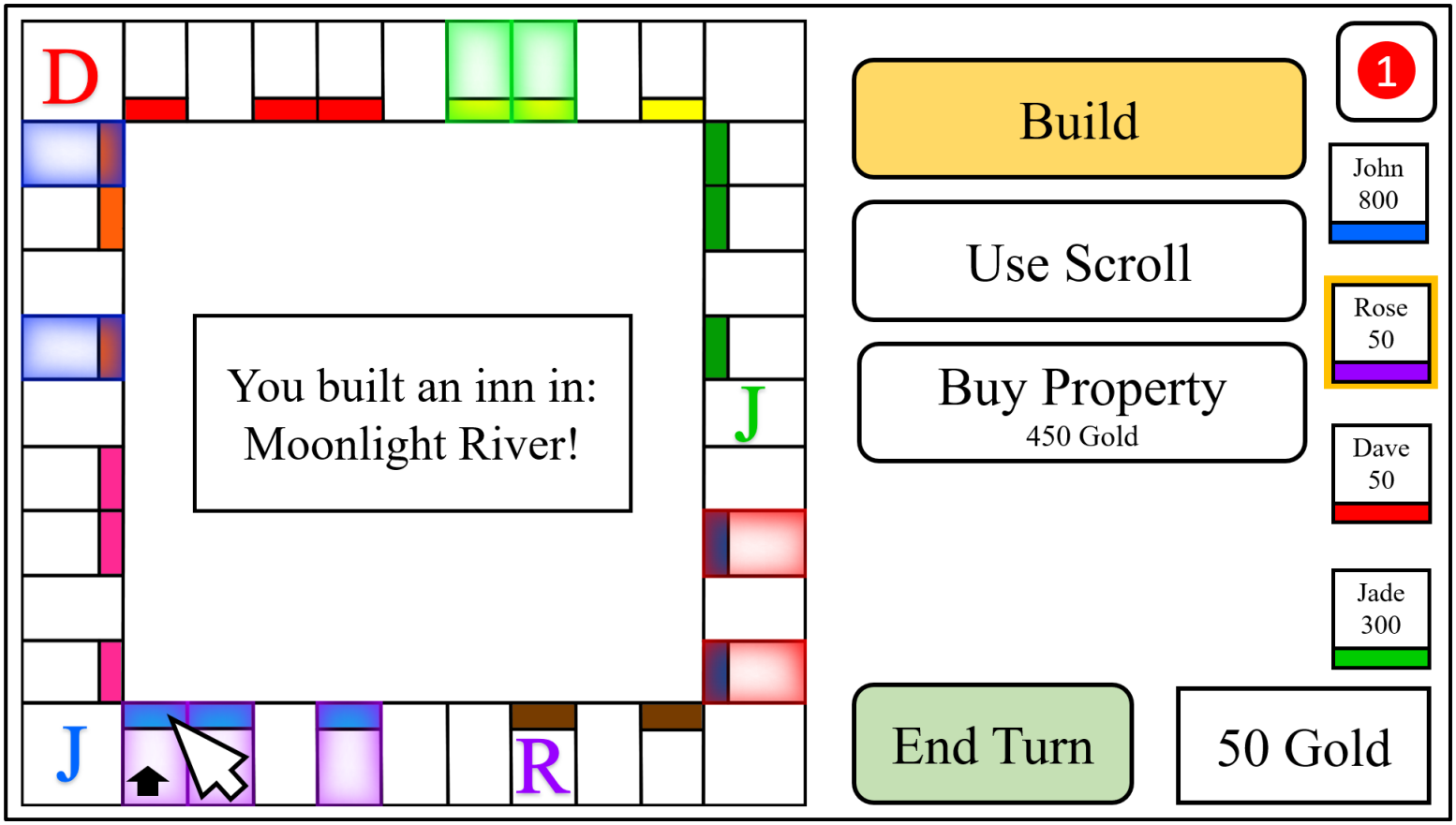

Figure 26 Game Screen When Building

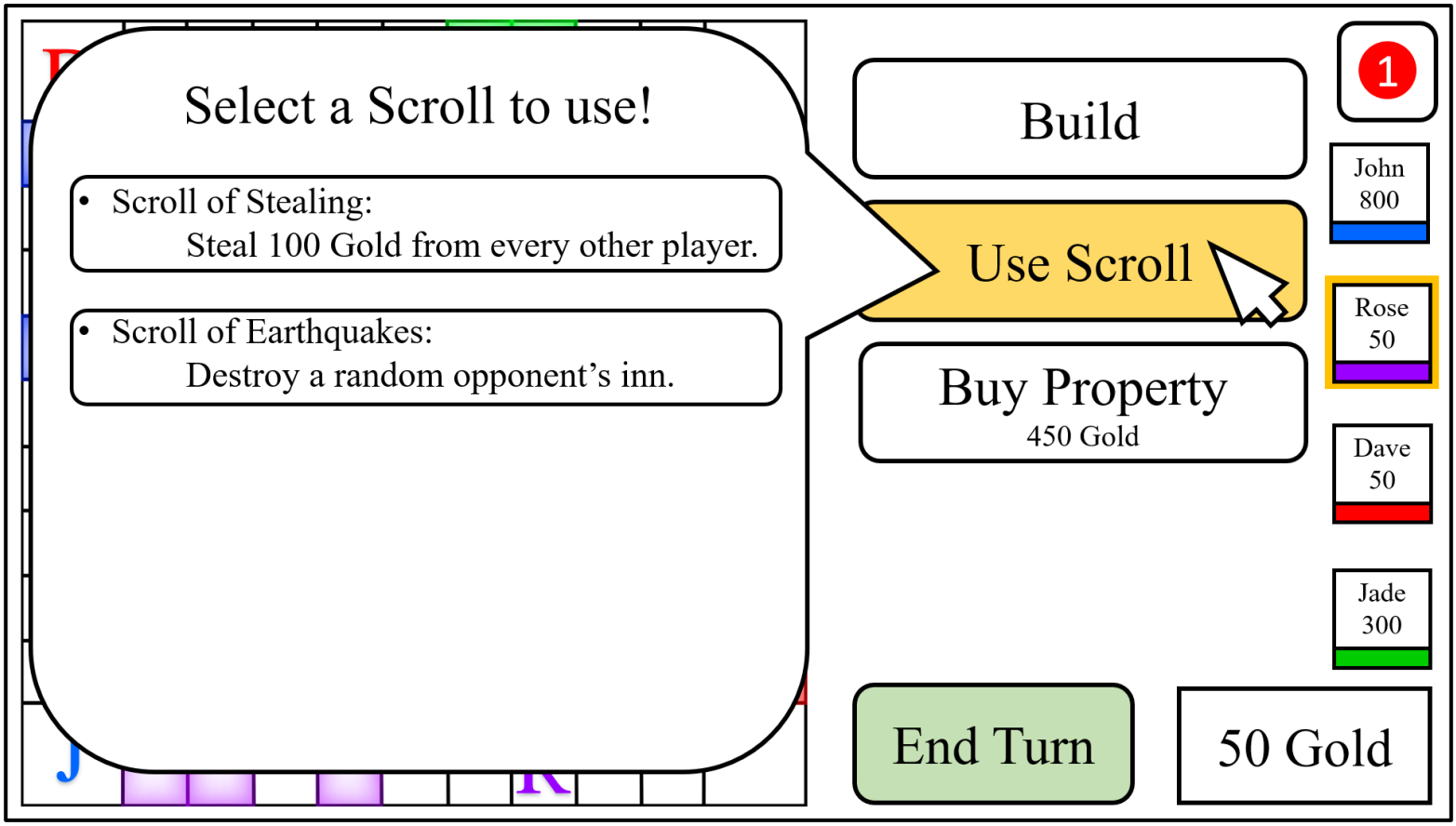

Figure 27 Game Screen When Using Scroll

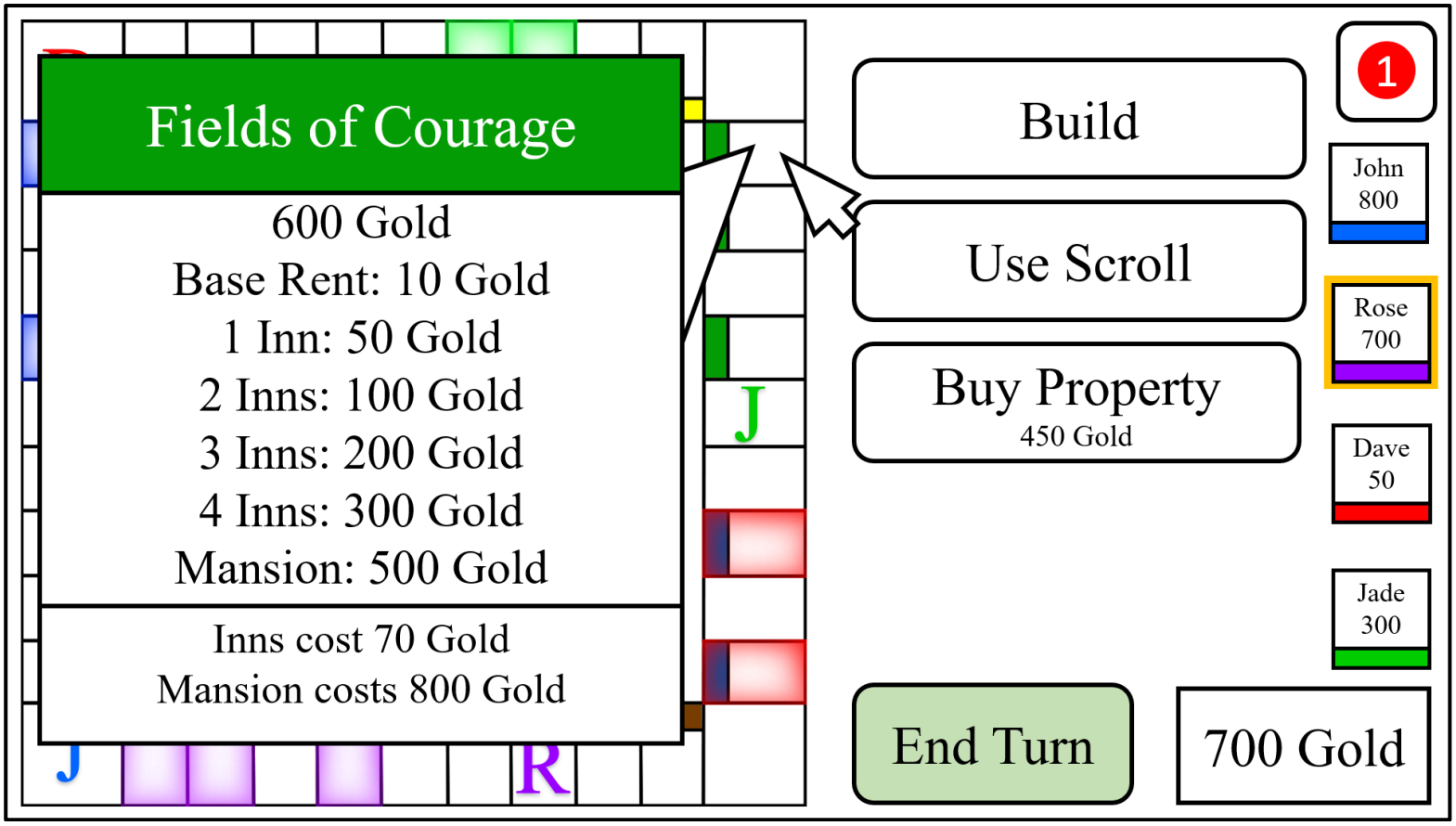

Figure 28 Game Screen When Looking at Property Title Card

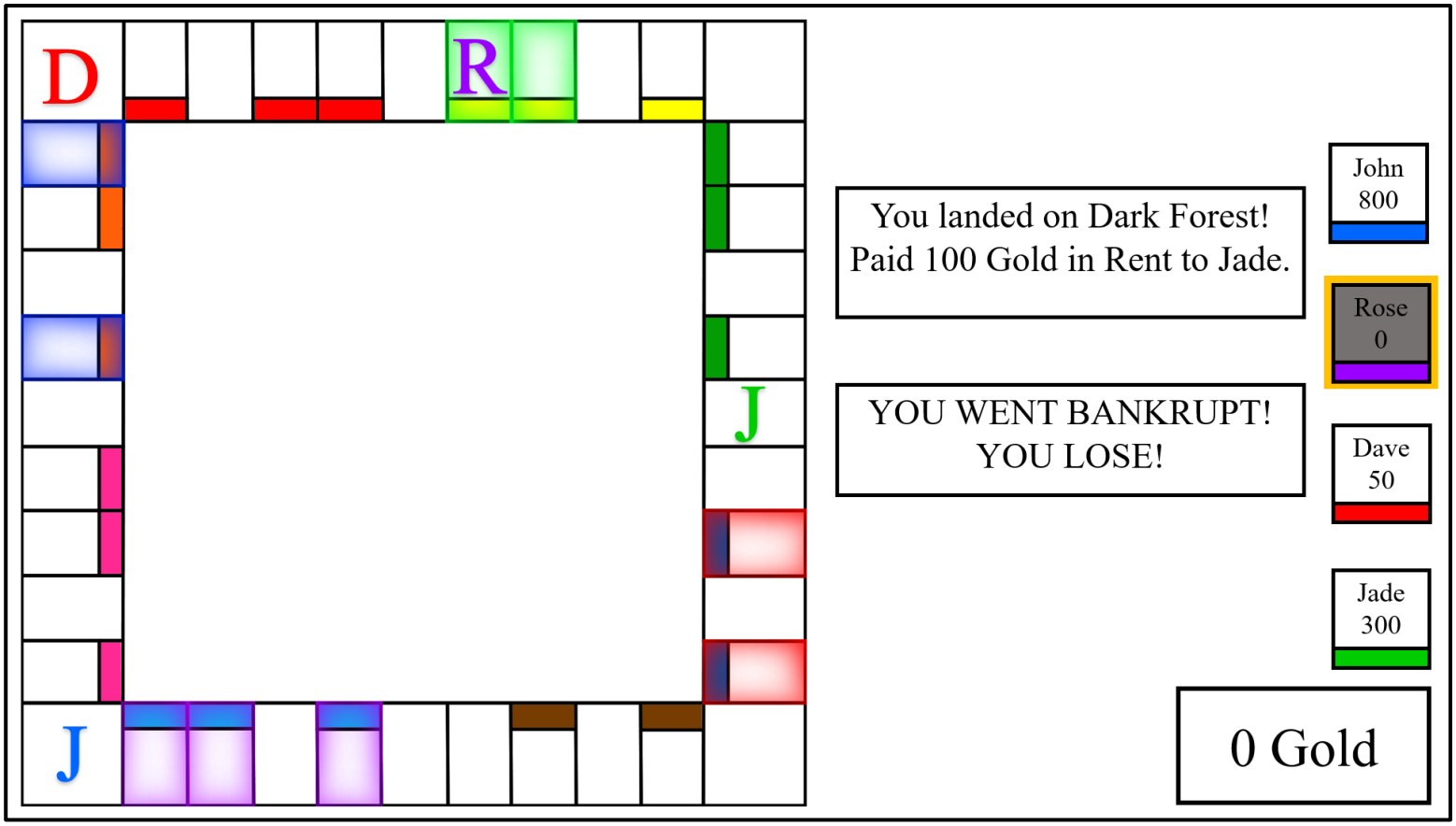

Figure 29 Game Screen When Losing Game

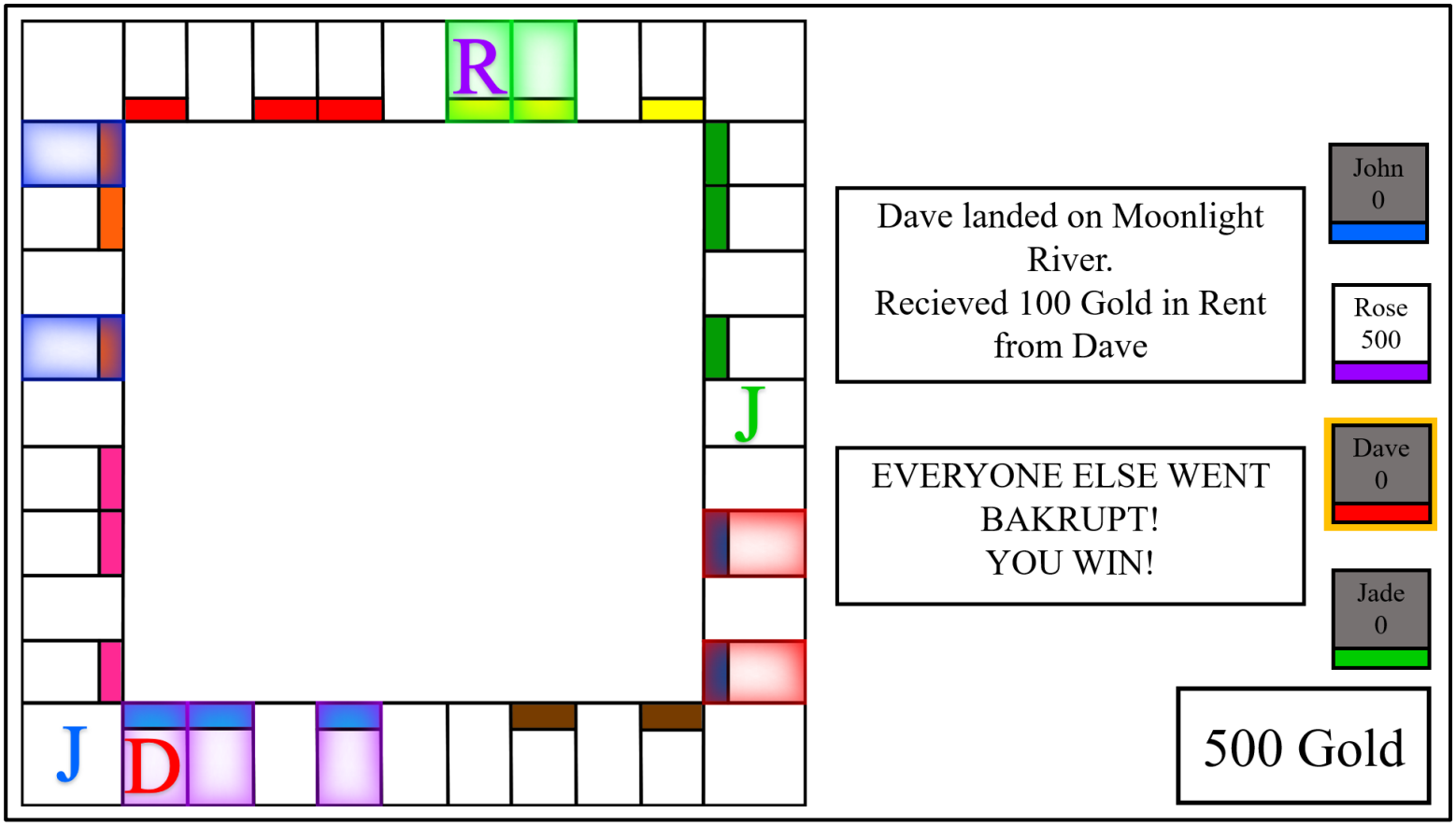

Figure 30 Game Screen When Losing Game

3 Glossary

- *Classes:* Not to be confused with classes in programming, the classes mentioned in this report, the classes in-game are the specializations of player characters, such as a Knight or a Wizard.
- *Inns:* A building that can be built on a town if all other towns of the same color are also owned by the same player. Increases the amount of rent visiting players have to pay. In order to build another Inn on a town, all the other towns of the same color have to have at least the same number of Inns already built as the current town in which an Inn construction is currently being attempted.
- *Mansions:* After four Inns are already built on a town, the player can turn in those four Inns and build a Mansion, which increases

the amount of rent visiting players have to pay even more than inns.

- *Fortune Cards:* Cards that have an immediate effect on the game when drawn. Can have both positive or negative effects to the player that drew it or other players.
- *Scrolls:* Cards that can be kept in the player's hand indefinitely and played during their turn. They have drastic effects on the game.
- *Tokens:* The tiny pieces that represent the players and show which square they are currently on.
- *Mana:* A special resource exclusive to Wizards. Fills up as they roll dice. Once full, allows the Wizard to draw a scroll before depleting again.
- *Churches and Cathedrals:* Advanced versions of Inns and Mansions. Provides the Cardinal with more rent than their regular counterparts.

4 Improvement Summary

We tried to implement all the feedback from the TA's and course coordinator. Firstly, to make our non-functional requirements more testable we changed adjective promises into numeric promises. In order to better describe user interactions with the system, we improved our Use-Case Model by discarding some unnecessary use-case scenarios and fixed mismatched associations in between these states. In the object model we tried to increase and detail our attributes for most of the classes. We added new Inn class and Mansion class since they are entity objects for Town class. Another addition is Server, ServerSideConnection classes and Player, clientSideConnection classes. These classes are further explaining our

client-server connection protocol. We also added multiplicity information to our object model. To improve our Dynamic model, we detailed our state diagrams. We focused on single events and described them thoroughly. We drew different state diagrams for different events. In our sequence diagrams parameter names and parentheses for used methods were missing so we added them. Activity diagram is simplified and dead-ends are fixed. We updated one of our class Interface. We are using Serializable instead of JSON object to send and receive game state data between the server and clients.

5 References

[1] "make your own monopoly board". https://sk.pinterest.com/pin/456130268489111725/. [Accessed: October 31, 2020]