Bilkent University

Department of Computer Engineering

Group 2G: Hasbrothers

Project short-name: Monopoly Bilkent

Analysis Report Iteration 2

- Mehmet Yaylacı
- Vural Doğan Akoğlu
- Mert Laleci
- Barış Tiftik
- İlhan Koç

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

Contents

Analysis Report

Group 2G: Monopoly Bilkent Edition

1. Introduction

Monopoly is a board game that can be played with 2 - 6 people. Players of the game simulate real estate business. The aim of the game is to purchase or invest in real estate properties as much as possible. Players throw dice to move through estates and complete the cycle. When a player lands on a property which is owned by another player he will lose money. If the player lands on a card property he will draw a card and he has to do whatever is written on the card. If the player lands on a property which is not owned by any other player, he can choose to buy this property or pass. If the players have chosen a time limit when the time limit is reached, whoever has the most real estate or cash wins. Bankrupting other players is another way of winning the game.

2. Current System

2.1. Game Elements

Board: The game board of the box version. It has locations for players to move through.
Properties: Properties that players can purchase
Houses and Hotels: Buildings that players can build on a property
Pawns: Pawns that players can select one to represent them inside the game
Chance and Community Chest Cards: Cards that players draw when they stop by a card location
Bank: One of the players is chosen to represent the bank. He or she is responsible for the money flow.
Money: Money that players start with and try to save or accumulate afterward.
Dice: At every turn, players roll dice to move on the board among locations.

2.2. Game Set-up

2-6 players can start a new game. Property Cards, Bank, Community Chest and Chance Cards and Pawns must be prepared in advance. A pair of dice is used to proceed in the game.

2.3. Gameplay

Players play their turn respectively. They complete the lap and start a new lap concurrently. They buy properties and build houses and hotels upon them. They earn the rent incomes from what they own as properties. At some locations, they pay tax and draw some cards that can make them certain things. Players who run

out of their money lose. The game loop continues until the end of the game. The last standing player is said to be qualified as a winner.

2.3.1. Throw the Dice and Move on the Board

Players should be able to throw the dice but their tokens will be moved automatically by the system**.** Each player throws the dice after he/she passes the “offers” screen. After the throw, the player's token will be moved forward the number of spaces shown by the thrown dice. The player can make different actions according to the place where the token comes, such as buy, pay rent, pick card, go jail.

2.3.2. Buy a property

The player comes to a property which has no owner, can select a buy option for this property with a certain amount of money. Then, the system automatically decreases the money in the player’s bank account.

2.3.3. Buy houses/hotels

At the beginning of the player's turn, if he/she has all properties of the same color, can select build the houses/hotels option and decide the properties to be built. If the number of houses of all properties in the same color has reached 4, the player can build a hotel instead of 4 houses. Also, if the player comes with the 3rd property which is the same color which he/she already owns, and the player buys it, has the option to build immediately.

2.3.4. Pay rent

The player comes to properties which are bought by opponent players, he/she needs to pay rent. The system automatically transfers the amount of rent from the player’s bank account to the opponent player’s one.This rent price can change by certain conditions. These conditions are as follows:

1. If the owner of the property owns one or two properties of the same color, the rent price increases.
2. If there are buildings (houses/hotel) on the property, the rent price increases.
3. If the property is chosen for the Mayfest area, the rent price is double.

2.3.5. Pick Mayfest area

The player comes to “pick a property for Mayfest” cell. By clicking, he/she decides the Mayfest area on one of the owned properties. The system marks this property as mayfest area and now it’s rent price is double.

2.3.6. Go to jail

The player comes to the “Go to jail” cell or picks a chance card which has the action as “Go to jail”. The system automatically moves the token of the player to the prison cell.

2.3.7. Exit the prison

When the player’s turn starts, if the player is already in jail, he/she rolls dice if the dice is double, and can move on the board according to dice. Otherwise, the system gives the player the option as “pay for exit” or use community card as “prison break” or “wait”, if the player has not the card this option is invalid. If the player chooses the “pay for exit” option, a certain amount of money is decreased from the player's account. If the player chooses the “wait” option, he/she waits until the next turn and tries to roll double again. After 3 turns the player in jail, the system let the player free.

2.3.8. Pick chance or community card

The game has two different types of card deck one of them is “chance” the other is “community” cards. Each deck has 15 cards, each card demands the players to do different actions. In general, community cards contain more player-friendly actions but what will come from the chance cards is total gambling because chance cards’ actions include high amounts of money but the players have the possibility to receive it as well as to pay. Community cards’ actions include the average amount of money and the players have the most possibility to receive it. In addition to giving and receiving money, these cards can take the players to another part of the board, give them a prison break card, let the players choose the mayfest area and more such options.

2.3.9. Make an offer

In the beginning of the player's turn, if the player wants to take a property which has an owner, can make an offer to the opponent player who owns the property. This offer can be made by multiple choice such as money for property, property for property, money + property for property. In the same way if the player wants to dispose of own property can make an offer to opponent players.

2.3.10. Accept an offer

In the beginning of the player's turn, if there are any offers for the player, the system checks for conflicting offers and displays them in the same color so that the user can choose only one of the conflicting offers. He/she can see all offers at the same time and decide to accept or decline for each offer. According to offer, the system rearranges the ownership of the properties and money transfers. Then, the player continues to his/her turn with rolling dice.

2.3.11. Borrow credit from the Bank

If the player has financial difficulties, he/she can demand a certain percentage of his/her financial assets from the bank at the beginning of his turn, but the player has to pay this debt with next starting point, if he/she does not pay, the interest rate increases, if the non-payment situation is repeated 3 times, the bank forces to the player to pay this total debt. If the player can not afford it, the bank explains the user's bankruptcy. Also, the bank does not give any new debt to the player if he/she has unpaid debt.

2.3.12. Pay tax

Whenever the player comes to the "pay tax" cell on the board, need to pay taxes to the bank according to a certain percentage of total money he/she has in the bank account. The system decreases this amount of taxes in the player's bank account.

2.3.13. Complete turn on the board

Whenever the player passes the starting point, the bank pays a salary for the player and the system increases the money of the player's account.

3. Proposed System

3.1. Current System vs Proposed System

In the current version, there is a bank inside of players. However, in the proposed system, no one is charged to serve as a bank. It is handled inside the game itself. There will be new features in the proposed system such as a Mayfest event in the car parking location. Also, players can take loans.

3.2. Game Mode

This game is designed to be finished in multiple modes. The lobby maker can choose the desired game mode(it can be multiple) before setting up the lobby. These modes are following:

1-Time limit (starting from 30 and continuing in 30 minute increments)
2-East Campus Strike (If one player buys all properties on east campus, game ends)
3-Athletic win (If one player buys all properties with sporting activities, game ends)
4-Strike of one side (If player buys all property contained in one of the sides on the game board, game ends)

3.3. Functional Requirements

3.3.1. Sign up & Sign In

Users have to sign up and sign in to the game. This allows them to save their settings and avatar.

3.3.2. Making Avatar

Each player can create their own avatar by selecting simple choices while registering for the game, such as gender, hair and eye color, glasses, beard.

3.3.3. Creation of Lobby and Joining Into a Lobby

Users can create a game lobby with the desired settings. After the creation, the creator can share the key of the lobby with the other participants so they can join the same lobby.

3.3.4. Making and Accepting Offers

At the start of each turn, players should be able to accept or make an offer. The details of these actions are explained in the game play section.

3.3.5. Throw the Dice and Move on the Board

Players should be able to throw the dice but their tokens will be moved automatically by the system.

3.3.6. Selecting Game-Settings

Players can decide game settings while creating a lobby. There will be two game modes for the settings. One of them decreases the time that player spends in the game and fastens the game-play by increasing rents.

3.3.7. Sound Effect

The sound effects of the game can change according to the actions in the game. For example, if a player comes to the mayfest area, a festival music will be active for a short time and whenever another player comes to the property that is chosen by the player, this feature will be activated again.

3.3.8. Mortgage Property

If the player is in an economically difficult situation, they can click and mortgage any of the properties they own, and the color of the property mortgaged on the board automatically changes and the system does not receive rent from the opposing players who return to this property.

3.3.9. Buying Houses and Hotels

Players who want to buy a house should buy them one by one. Before buying the first house on properties with the same color group, a player cannot buy a second house. To build a hotel, a player needs fourth houses on each property. When players buy a house or hotel, a visual building will appear on the property.

3.3.10. Pick card

Players who came to the chance cell on the board, will see 3 different closed cards and have to pick one of those by clicking. By clicking one, the card will be revealed and apply what is written in the card.

3.3.11. Go to jail

If players arrive in the cell called "Go to jail" or choose "Go to jail" from the chance and community cards or throw dice double three times in a row in their turn, their avatar automatically goes to the jail cell on the board.

3.3.12. Exit from jail

If a player is in prison when his/her turn starts, he/she throws the dice first, if he/she does not roll a double and if he/she does not have a break-out card he/she can get out of jail by giving money. If the player does not give money, these options are repeated every time he/she turns starts, and after 3 rows he/she is released from prison without giving money.

3.4. Non-functional Requirements

3.4.1. Usability

The opening page will provide a help button that explains all the buttons that are used in the game and explains the game-play of the game.

3.4.2. Reliability

The system will show error messages to the user so the user will be informed about problems and its solutions. If it was a connection error that is caused by a user, the system will wait 1 minutes for the player. Privacy of players information such as id will be provided. The system must be available 24 hours.

3.4.3. Performance

This game will react to user decisions in a limited time and perform game actions within a time that the players cannot see the delay.

- The launching of the will take less than 30 seconds.
- Sign up & Sign in part will take less than 1 seconds if the player enters the correct user name and password combination.
- Opening the settings and strolling between different avatars will take less than 1 second.
- The creation of the lobby will take less than 1 second.
- The joining operation to the existing lobby will take less than 5 second.
- Turn passes between players will take less than 0.5 second when the player's turn is over.
- The mayfest effects will be invoked less than 0.5 second.
- The system should support hundreds of players within the restrictions above.

3.4.4. Supportability

This Monopoly Bilkent will be adaptable to the new versions of the operating systems. Also, The Monopoly Bilkent's graphical user interface will be suitable.

3.5. System Models

3.5.1. Use-Case Diagram

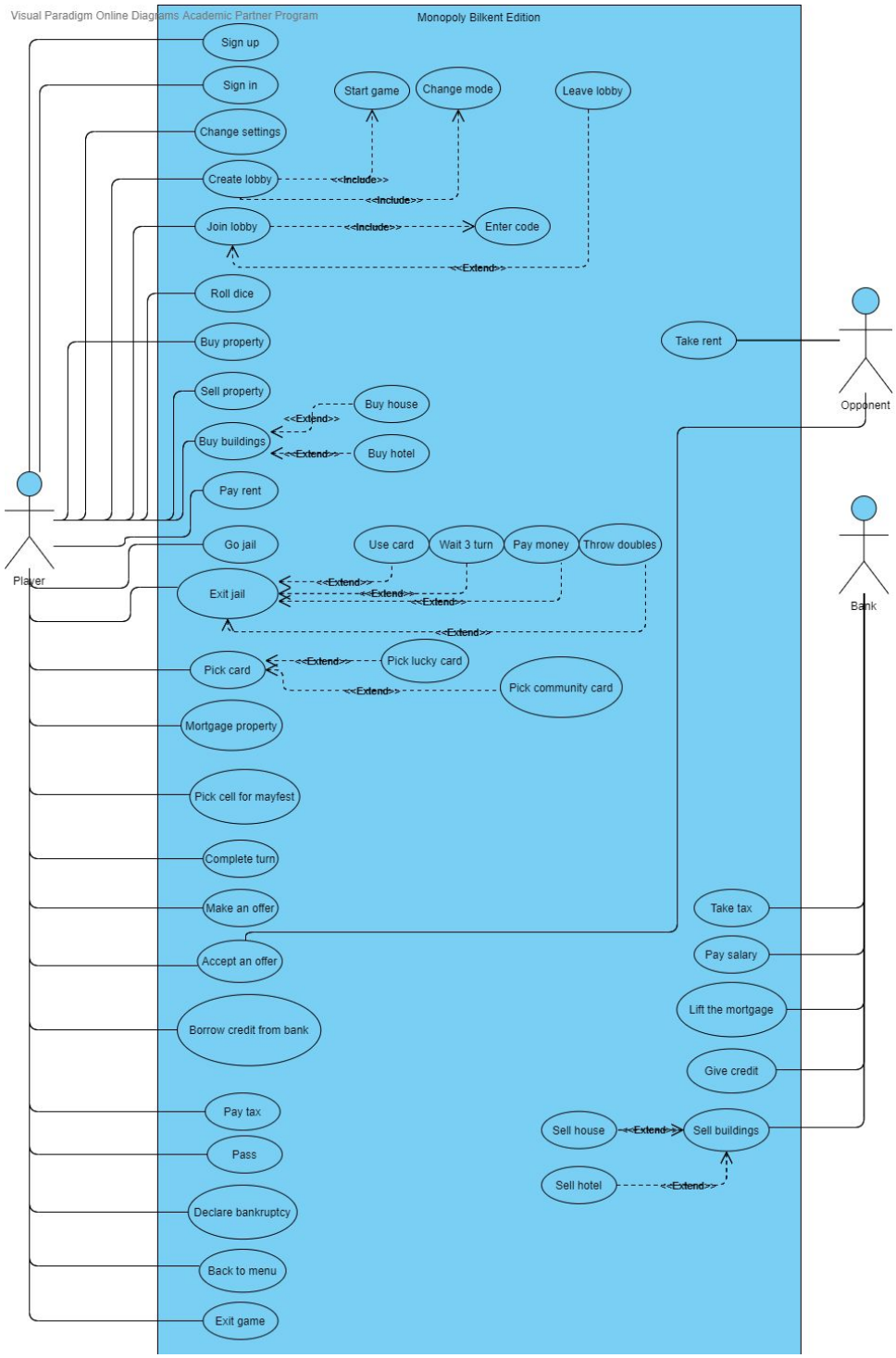


Figure 1: Use-case diagram for Monopoly Bilkent

Use case name	Change Settings
Participating actors	Initiated by User
Flow of events	1. If the user wants to change his/her username, he/she rewrites the new username he/she wants. 2. If the user wants to change his/her in-game avatar, he/she can choose from different avatar options. 3. If the user wants to change the current settings of audio description and/or sound effects, he/she selects the corresponding setting. 4. If the user wants to apply a change in any setting of the application he/she saves the settings, otherwise cancels the changes. 5. If the user saves his/her settings, the system updates the settings.
Entry condition	The user invokes the “Change Settings” use case
Exit conditions	The user has saved the chosen settings.

Use case name	Create Lobby
Participating actors	Initiated by User
Flow of events	1. If the user wants to create a lobby he/she has to decide in-game settings first. 2. Then if there are more than three players in the lobby, the creator can start the game.
Entry condition	The user invokes the “Change Settings” use case
Exit conditions	The user has saved or canceled the settings
Quality requirements	The user must select a unique username and valid password.

Use case name	Join Lobby
Participating actors	Initiated by user
Flow of events	1. The user enters the lobby code (include use case Enter Code). 2. If the user does not want to join the room, the user cancels the use case. 3. If the user selects to join the lobby, the system makes the player join the lobby. (includes the use case Enter Code) 4. If the user successfully joins to lobby, user can choose to leave the lobby as well.(include use case Leave Lobby)
Entry condition	The user invokes the “Join Lobby” use case
Exit conditions	The user has joined the game or left the lobby.

Use case name	Roll dice
Participating actors	Initiated by user
Flow of events	1. The user's turn has been started. 2. The user rolls the dice.
Entry condition	The user automatically invokes the "Roll Dice" use case when his/her turn starts.
Exit conditions	The user has rolled the dice.

Use case name	Buy Property
Participating actors	Initiated by user
Flow of events	1. The user wants to buy a property. 2. If the user has enough money in his/her bank account. 3. The amount is decreased from the user's bank account.
Entry condition	The user a invokes the "Buy Property" use case.
Exit conditions	The user has bought the property.

Use case name	Sell Property
Participating actors	Initiated by user
Flow of events	1. The user sells his/her property by making an offer. 2. The opponent accepts the offer.
Entry condition	The user makes an offer to sell his/her property.
Exit conditions	The opponent accepts the offer.

Use case name	Buy Buildings
Participating actors	Initiated by user
Flow of events	1. The system places the buildings into the board.
Entry condition	The user invokes the "Buy Buildings" use case.
Exit conditions	The system has been placed into the board.

Use case name	Pay Rent
Primary actor	Player
Flow of events	1. Player comes to property which belongs opponent player 2. The system transfers the amount of rent from player's account to opponent's account
Entry condition	Player rolled dice
Exit conditions	The system transferred rent

Use case name	Take Rent
Primary actor	Opponent Player
Flow of events	1. Player comes to property which belongs opponent player 2. The system transfers the amount of rent from player's account to opponent's account
Entry condition	Player rolled dice
Exit conditions	The system transferred rent

Use case name	Go Jail
Primary actor	Player
Flow of events	1. Player comes to cell which is "Go to jail 2. Player's token automatically moved to prison by the system
Entry condition	Player rolled dice
Exit conditions	The system moved Player's token

Use case name	Exit Jail
Primary actor	Player
Flow of events	1. If player has the Community Chest card which calls "prison break" can use this card to escape from prison 2. Player tries to roll double in order to exit from prison 3. If dice is not double, player can pay money to exit from prison 4. If player already has been prison for 3 turn, the system lets player free
Entry condition	When player's turn starts, he/she already in jail

Exit conditions	Player used “prison break card” or rolled double or payed money or waited 3 turns

---

Use case name	Pick Card
Primary actor	Player
Flow of events	1. Player comes to cell which is pick “Chance” or “Community Chest” card 2. Player clicks the top of the card deck 3. The system shows the message in the card which player picked 4. The system automatically does the action given by card
Entry condition	Player rolled dice
Exit conditions	The system did the actions according to message given by card

---

Use case name	Mortgage Property
Primary actor	Player
Flow of events	1. Player clicks mortgage property option 2. Player selects his/her own property 3. The bank increases money in player’s account 4. The bank mortgages the property
Entry condition	Player clicks mortgage property option
Exit conditions	The bank mortgaged the property

Use case name	Pick cell for Mayfest
Primary actor	Player
Flow of events	1. Player comes to cell which is pick “Mayfest area” 2. Player clicks one of the his/her property for Mayfest area 3. The system marks the property for “Mayfest area”
Entry condition	Player rolled dice
Exit conditions	The system marked the property for “Mayfest area”

Use case name	Complete turn
Primary actor	Player

Flow of events	1. Player passes to cell which is “Start” 2. The banks increases money in player’s account
Entry condition	Player rolled dice
Exit conditions	The bank increased money in player’s account

Use case name	Make an offer
Primary actor	Player
Flow of events	1. Player clicks to make an offer option and decide to opponent he/she want to offer 2. Player clicks “property” or “money” or “both” option which he/she will offer 3. Player selects the property and amount of money he/she will offer 4. Player clicks “property” or “money” or “both” option which he/she will want to take 5. Player selects the property and amount of money he/she will want to take 6. The system record the offer to show the opponent player
Entry condition	Player clicks to make an offer option
Exit conditions	The system recorded the offer

Use case name	Accept an offer
Primary actor	Opponent Player
Flow of events	1. If player has an offer, the system show the offer to player 2. Player clicks “accept” or “decline” 3. If player clicked accept, the system transfers money from one account to another and rearranges the owner of properties 4. Player rolls dice
Entry condition	Player’s turn starts
Exit conditions	Player rolled dice

Use case name	Borrow credit from bank
Primary actor	Player
Flow of events	1. Player clicks borrow credit from bank option 2. The system shows the player the maximum amount he/she can borrow after the calculation of a certain percentage of his/her total money

3. Player selects the amount of money he/she wants to take
4. The system shows the player the how much to pay with interest after player completed 1 turn
5. Player clicks “accept” or “decline”
6. If it is accept, The bank increases money in the player’s account
7. Player rolls dice

Entry condition	Player clicks borrow credit from bank option
Exit conditions	Player rolled dice

Use case name	Pay tax
Primary actor	Player
Flow of events	1. Player comes to cell which is “Pay Tax” 2. The bank reduces money in the player’s account
Entry condition	Player rolled dice
Exit conditions	The bank reduced money in the player’s account

---

Use case name	Pass
Primary actor	Player
Flow of events	1. Player comes to property which is not belong any opponents 2. If Player does not want to buy the property, can select pass
Entry condition	Player rolled dice
Exit conditions	Player selected pass option

Use case name	Declare Bankruptcy
Primary actor	Player
Flow of events	1. Player clicks declare bankruptcy option 2. The system show a message for sure “Do you accept your Bankruptcy” 3. If Player clicks accept, all of the properties of the player returned to the bank and become unowned and player’s account is deleted 4. Player’s name is added the ranking table of the game
Entry condition	Player clicks declare bankruptcy option
Exit conditions	Player’s name is added the ranking table

Use case name	Back to Menu
Primary actor	Player
Flow of events	1. Player who lost the game clicks the back to menu option 2. Player go back to menu
Entry condition	Player clicked the back to menu option
Exit conditions	Back to Menu

Use case name	Exit Game
Primary actor	Player
Flow of events	1. Player clicks exit game option 2. The game is closed
Entry condition	Player clicked exit game option
Exit conditions	The game is closed

3.5.2. Object and Class Model

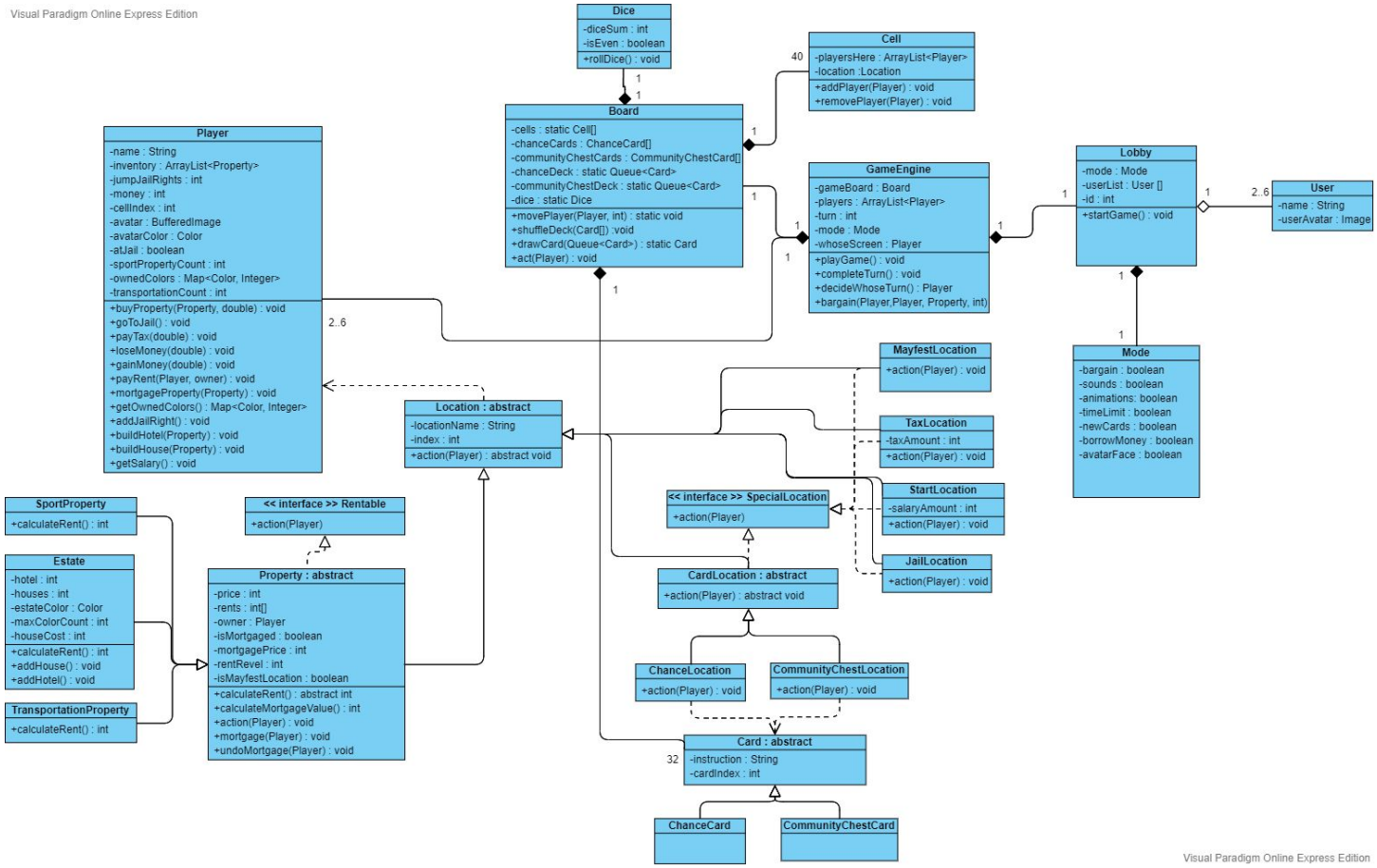


Figure 2: Class diagram for Monopoly Bilkent

Player: Represents a player of classical Monopoly.
-money savings of the Player
-name name of the Player
-jumpJailRights free pass credits for jail
-cellIndex current location of the Player
-avatar an avatar of a player
-avatarColor color of the avatar
-atJail flag for the player if he is at jail
-ownedColors (general explanation) counters of the colors that Player has
-inventory a collection of what Player has as properties
-sportPropertyCount number of sport properties for the athletic win mode
-transportationCount number of transportation properties
+payTax(int) Player pays the tax if he visits a tax location
+getSalary() Player gets the salary if he visits the start location
+buyProperty(Property) Player buys a property calls addLocationToOwned

+buildHouse(), +buildHotel() Player builds houses or a hotel to an estate
+ownedColor() Returns the colors of the completed color collections e.g. when three yellow estates is owned yellow will be added to return list
+loseMoney(double) is called when Player loses money
+gainMoney(double) is called when Player gains money
+borrowMoney(double) Player take loans
+addPropertyToOwned(Property) is called in the buy property method

Cell: A data structure class that holds locations and etc.
-location Cell holds a location
-playersHere current players who are actually here
+addPlayer(Player) add a player here
+removePlayer() removes player from the cell

Board: A game board that hold cells
-cells 40 cells
-chanceCards Array of chance cards
-communityChestCards Array of comm. Chest cards
-chanceDeck Array of shuffled chance cards
-communityChestDeck Array of shuffled comm. Chest cards
-dice Dice of the game
+movePlayer(Player, int) moves the player in board
+shuffleDeck() shuffles the card deck for new draws
+drawCard() draws a card for player
+act(Player) action method of current location (like go to jail)

Location: A class which is the parent class of the other location types in the game.
-locationName Holds the name of the location
-index Starting from the “start” location this holds the value of the location relative to the board.
+action(Player) It is an abstract method for all inherited classes

<< interface >>
Special Location: These interface is implemented by some special locations
+action(Player) It is an abstract method for all inherited classes

Mayfest Location: This is a special location which is free parking location in original board game monopoly
+action(Player) It is an method for all inherited classes

Tax Location: This location make player some pay taxes
-tax amount Tax amount of that special location
+action(Player) It is an method for all inherited classes

Start Location: Player gets Salary after passing this location
+action(Player) It is an method for all inherited classes

Jail Location: Player goes to jail if he come to “Go to Jail” location on the board
+action(Player) It is an method for all inherited classes

<<interface>>
Rentable: This interface is implemented by properties
+action(Player) It is an method for all inherited classes

Property: A class that holds buyable locations.
-price Price of the property
-rents To access different levels of rents
-owner Which player owns the property.
-isMortgaged Flag of whether it is mortgaged or not
-mortgagePrice Simply half of the price
-rentLevel Level according to building count or property count
-isMayfestLocation Flag of whether it is mayfest location
+calculateRent() this will calculate the rent for the Player to pay.
+calculateMortgagedValue() Halves the price
+mortgage(Player) Mortgages a property
+action(Player) It is an method for all inherited classes
+undoMortgage(Player) Undo mortgage in a property

Sport Property:
+calculateRent() this will calculate the rent for the Player to pay.

Transportation Property:
+calculateRent() this will calculate the rent for the Player to pay.

Estate: A class that holds buyable locations.
-hotel Shows if there is a hotel or not.
-houses House count of that estate
-estateColor Color of that estate
-maxColorCount MAX color count of that estate e.g. Blue’s is 2 or Red’s is 3
-houseCost Cost of one house when it is about to be built.
+calculateRent() Calc
+addHouse() Add a house if conditions are satisfied
+addHotel() Add a hotel if conditions are satisfied

Card: A class that holds cards.
-instruction A long index of the card instruction
-cardIndex Card index of that card

Chance Card: A class that holds cards.

Community Chest Card: A class that holds cards.

Card Location: A class that picks cards and do action
+action(Player) It is an method for all inherited classes

Chance Location: A class that picks chance cards and do action
+action(Player) It is an method for all inherited classes

Community Chest Location: A class that picks community chest cards and do action
+action(Player) It is an method for all inherited classes

Game Engine: A class that has the loop that makes the game continue.
-gameBoard: Game Board (Board Class.)
-players [] Holds the players
-turn Number of the turns played
-mode Holds a Mode object
-whoseScreen Indicates whose screen
+decideWhoseTurn() returns the Player who has the turn.
+completeTurn() this method will complete the actions after the turn has been played.
+bargain(Player, Player, Property) this method will do the actions of a bargain and change the attributes accordingly.
+playGame() this method is the main loop.

Dice: This class is the dice class.
-diceSum Total sum of 2 die
-isEven Returns true if 2 die is even
+rollDice() this method will roll dice that will return the summation of two dices that have been thrown randomly

Lobby: Concerned with the actions with the creation of online lobbies.
-mode Holds a mode object.
-userLİst [] This holds the Users in a game lobby
+startGame() This function will start the game.

Mode: A class that keeps track of the game rules that have been chosen
-bargain Rule of the game
-sounds Change if sounds are present
-animations Change if animations are present
-timeLimit Change if there is a time limit for the game
-newCards Change if new cards are present
-borrowMoney Change if players can borrow money from the bank
-avatarFace Change if avatars with faces can be selected

User: This is a player, but before starting a game.
-name The name of the User
-userAvatar Photo chosen by a User to be displayed as an avatar

3.5.3. Dynamic Models

3.5.3.1. Sequence Diagrams

3.5.3.1.1. Getting Out of Jail

Player's Actions: This diagram indicates the player's actions when it is in jail. We assume that the game has been initialized and a Player object named 'p1', a Dice object named 'd' and a Board object named 'b' has been initialized. Firstly the Player gets its get out of jail number. Then we throw the Dice. Then, if the Player has rights left it will move using the 'move' method in Board, Player will have its rights reduced. If the Player has no rights left, then we look if the number returned from the Dice is even. If even, we move the Player. If not even the turn for this Player ends and we use Game Engine class's 'next player' method. This is the only instance on this diagram we call this method because it is clear that the Player's turn has ended here. With other alternative scenarios the Player can make other choices like buying properties or bargaining with other Players.

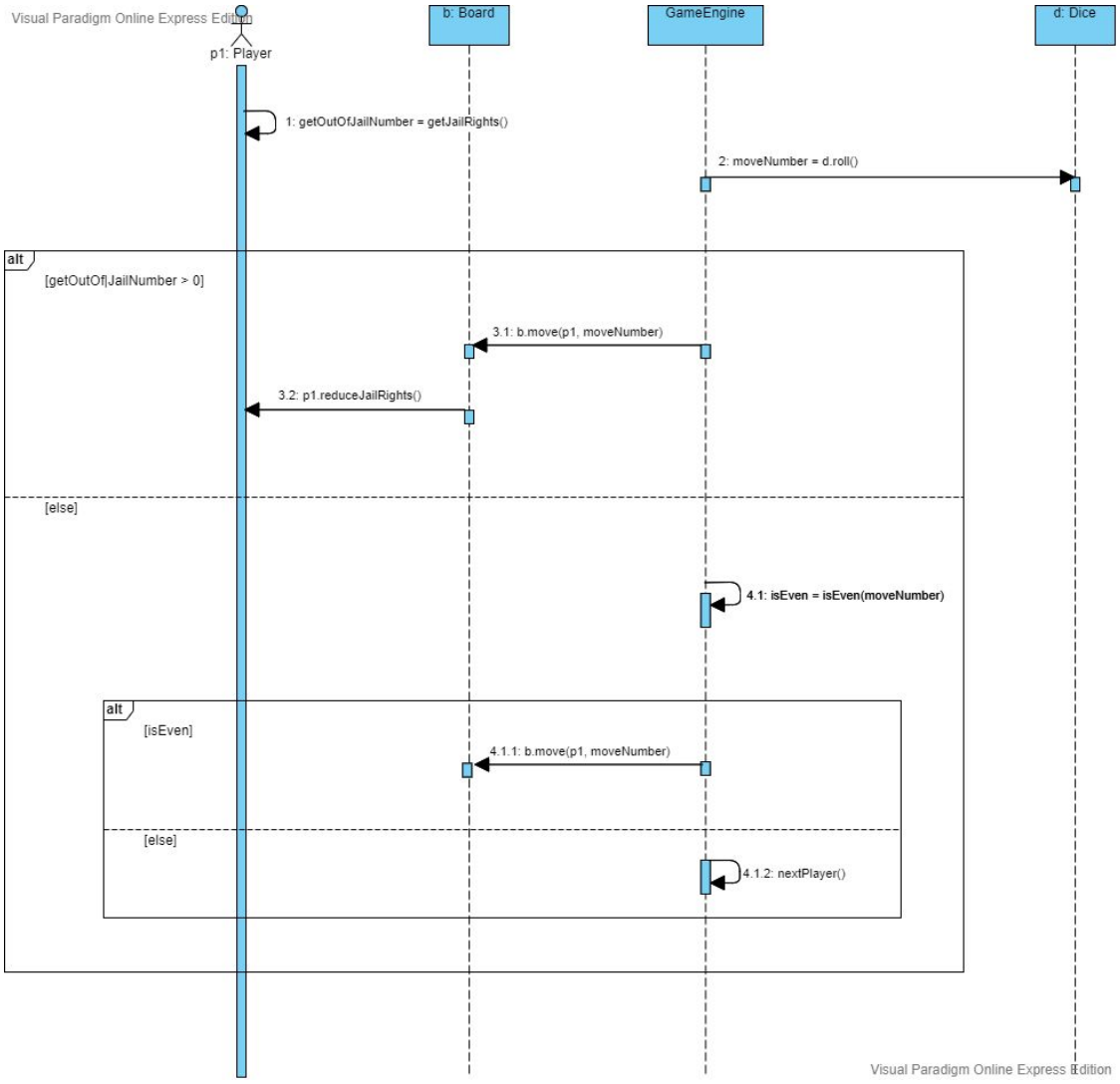


Figure 3: Sequence diagram about jail for Monopoly Bilkent

3.5.3.1.2. Buying Property and Building House

Player's Actions: This diagram indicates the player's actions with the example of buying a Property it has moved and adding a house (or building) on another Property. This diagram assumes that two Property objects named prop1 and prop2 have been initialized with a Player object named p1 and a Cell object named cell. Firstly the Player has chosen to buy the Property on the cell it has landed on. Then, Game Engine class uses cell's 'find property' method to find the Property it is on. Game Engine class calls its 'buy specified' method to turn to the Player p1. Then, Player uses the 'get price' method of the prop1 to get the price of the Property. Then, Player gets its money with the 'get money' method. If the money is bigger than the price we add the Property to the Player's list of Properties. Then we reduce the Player's money according to price. Then, the Player has chosen to add a house to its owned Property named prop2. Then the Player gets the current level of the buildings of prop2 using Property's method. Then the Player also gets the price of adding a building using Property's method. Then we will get the money of the Player again because the Player's money has changed earlier. If the price is lower than the money then the Players raises the building level of the Property by one using Property's 'set current level' method. Then we decrease the money of the Player according to the price. Here the turn of the Player has not ended so Game Engine class will not use its 'next player' method yet.

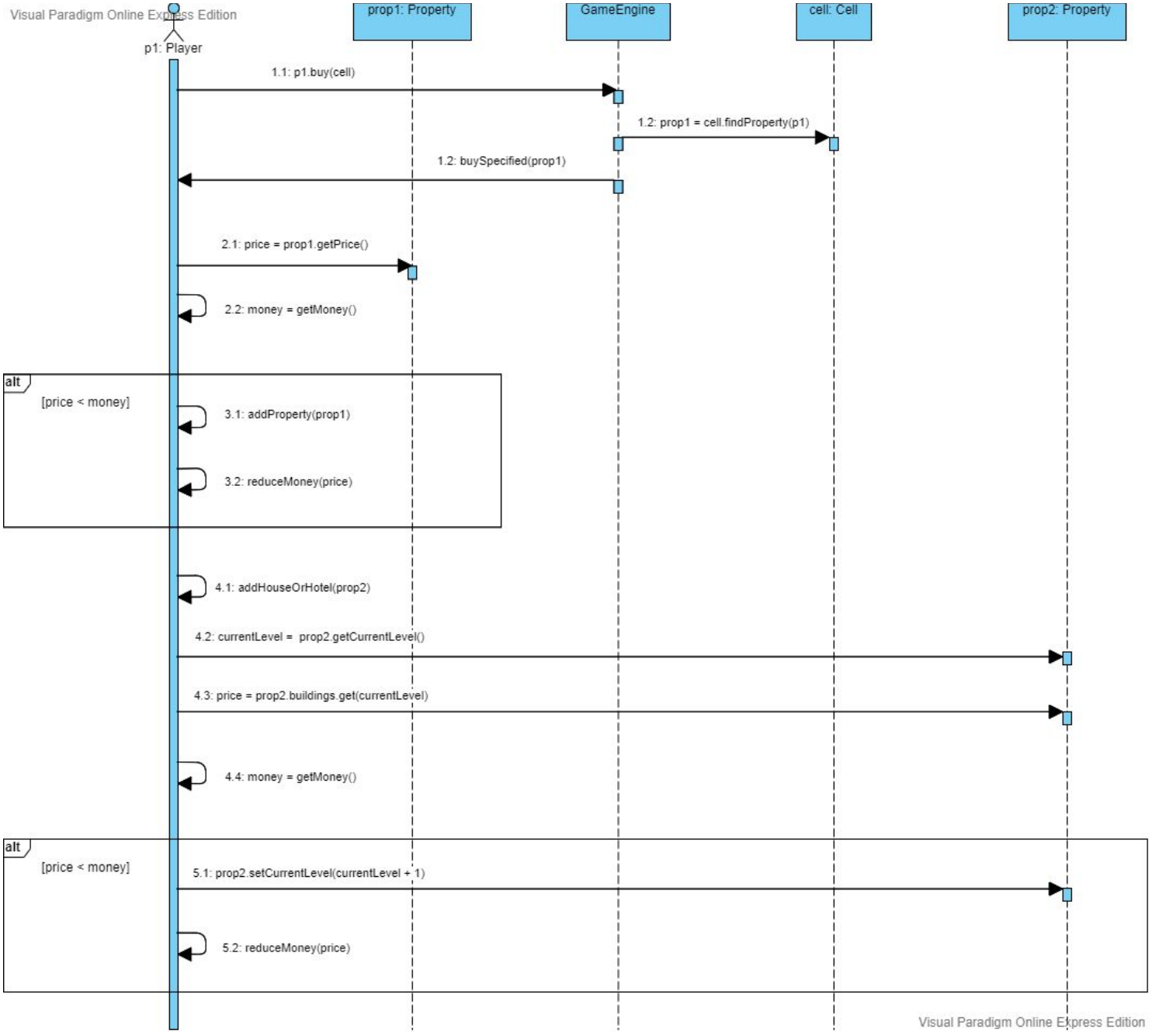


Figure 4: Sequence diagram about property actions for Monopoly Bilkent

3.5.3.2. State Diagrams

Buying a hotel state diagram: This diagram shows the path that a player has to follow to buy a new house or a hotel. You cannot erect more than one house on any one property of any color-group until you have built one house on wery property of that group. You may then begin on the second row of houses, and so on, up to a limit of four houses to a property. For example, you cannot build three houses on one property if you have only one house on another property of that group. (monopoly manual)

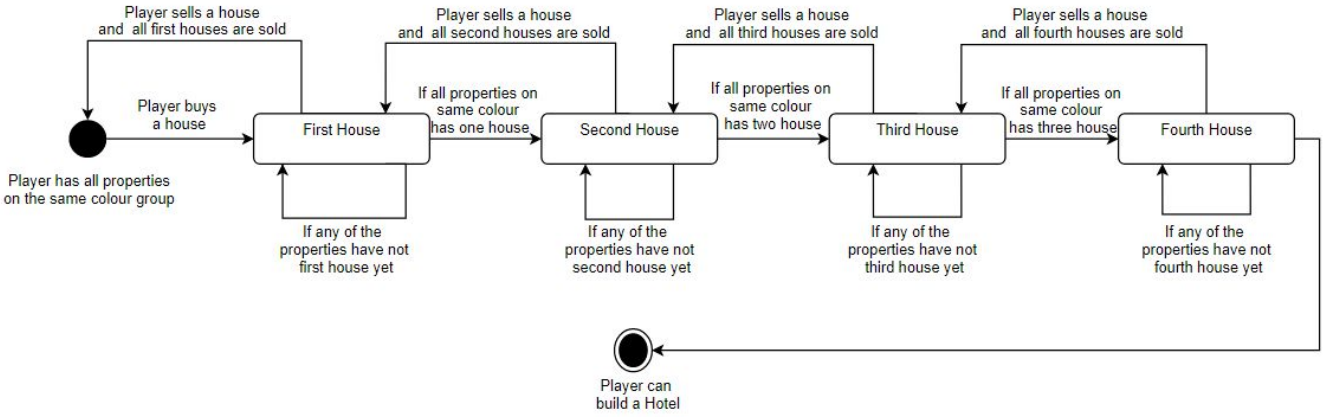


Figure 5: State diagram for Monopoly Bilkent

Going in and out of jail: You get out of Jail by.. .(I) throwing doubles on any of your next three turns; if you succeed in doing this you immediately move forward the number of spaces shown by your doubles throw; even though you had thrown doubles, you do not take another turn; (2) using the "Get Out of Jail Free" card if you have it; (3) purchasing the "Get Out of Jail Free" card from another player and playing it; (4) paying a fine of $50 before you roll the dice on either of your next two turns.(monopoly manual).

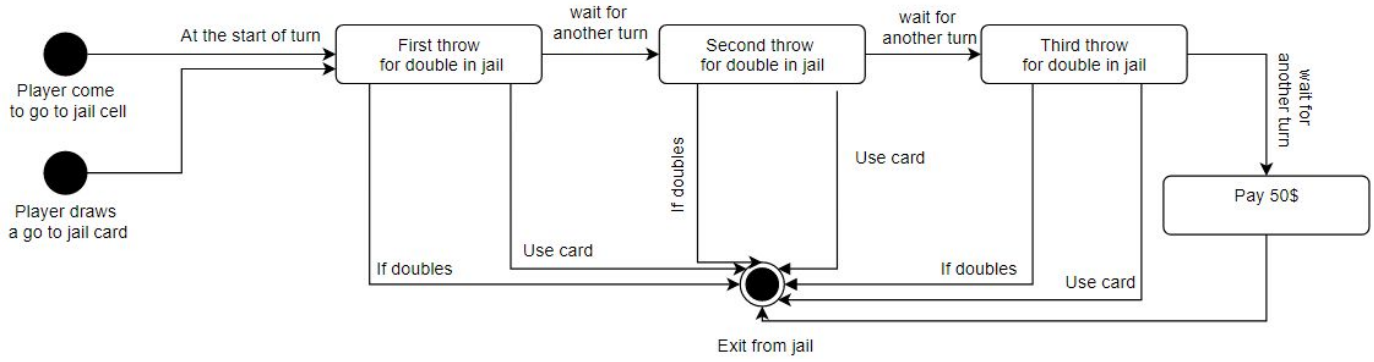


Figure 6: State diagram for Monopoly Bilkent

3.5.3.3. Activity Diagrams

Roll Dice and Play diagram: This diagram models the system behaviors after a player's turn starts. It also shows the system decisions and reactions to the actions of the player.

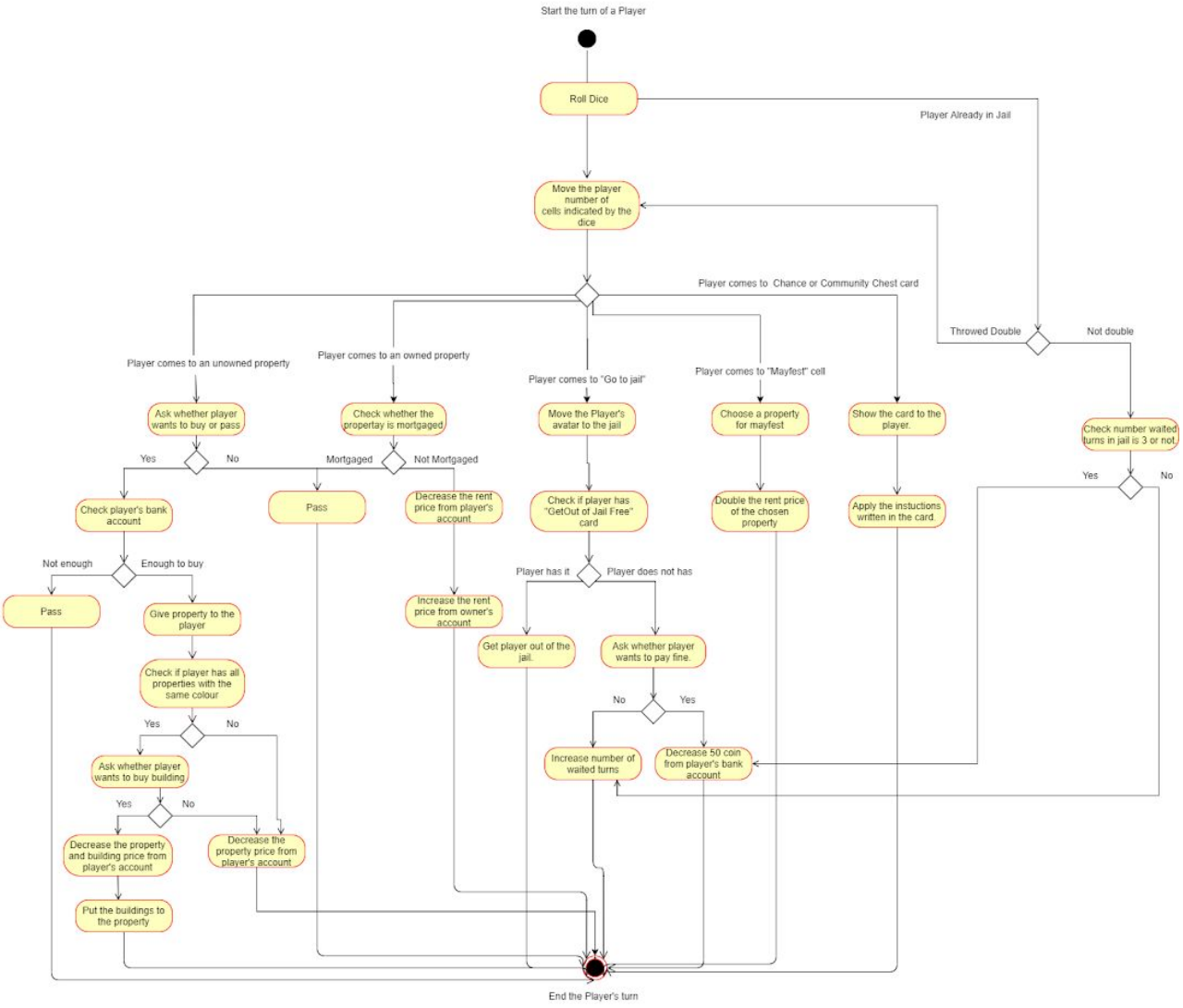
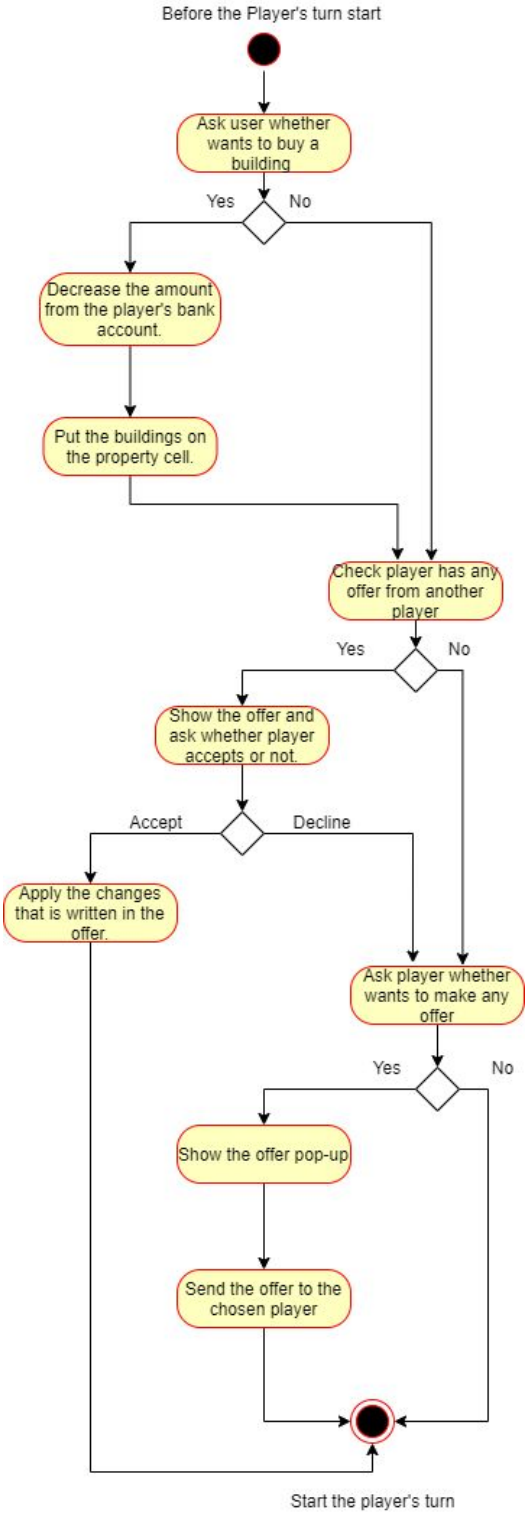


Figure 7: Activity diagram for Monopoly Bilkent



Before the player's turn starts, make an offer - buy building diagram: This diagram shows how players make offers and buy houses and the decisions system takes care in these processes.

Figure 8: Activity diagram for Monopoly Bilkent

3.5.4. User Interface & Screen Models

Figure 9: Mock-up for Sign-In screen

Figure 10: Mock-up for Sign-Up screen

Figure 11: Mock-up for Main Menu screen

Figure 12: Mock-up for Join-Lobby screen

Figure 13: Mock-up for Create-Lobby screen

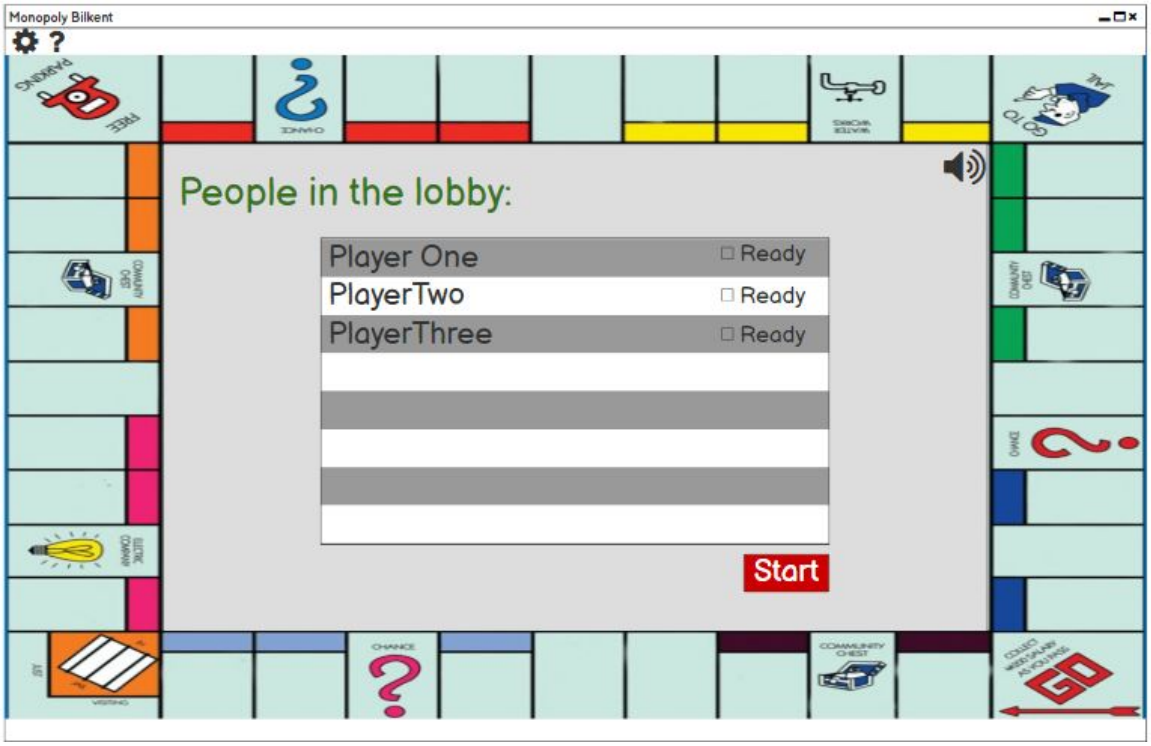


Figure 14: Mock-up for Lobby screen

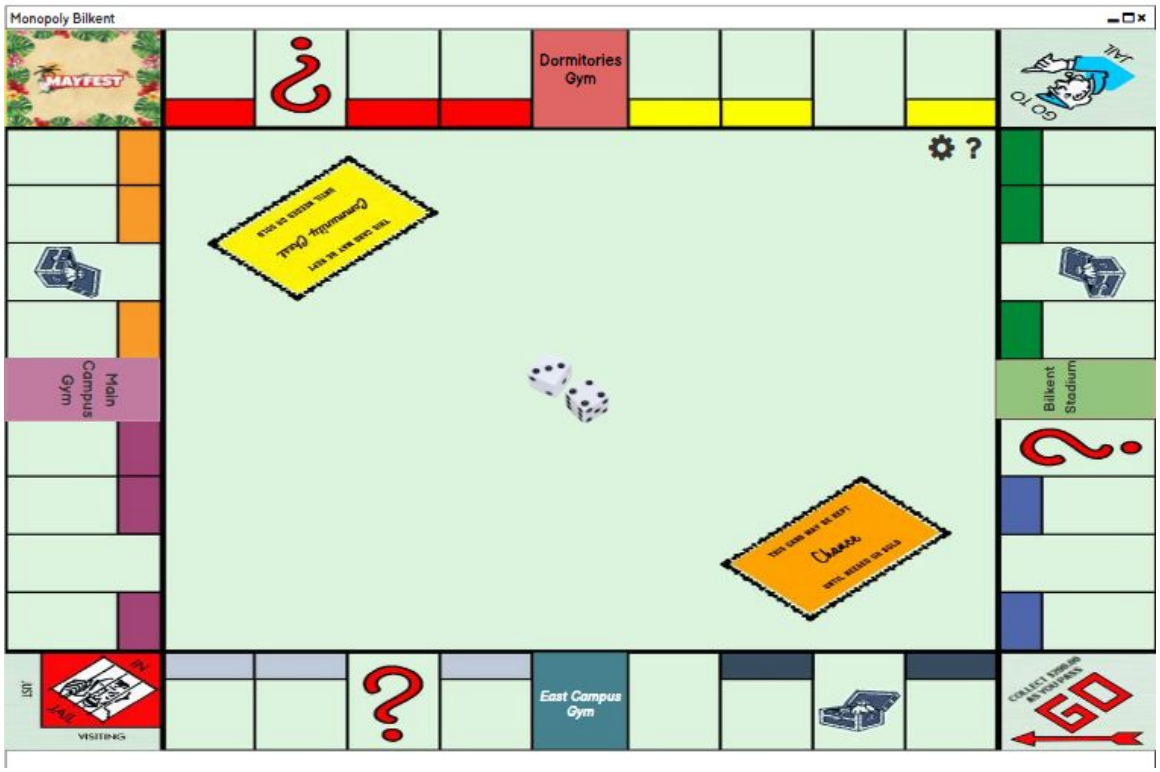


Figure 15: Mock-up for In-Game screen

4. Improvement Summary

In our first report, since we are not fully familiar with the defining features of our project and try to chance thinking in more detail during the project progress, in addition to the functional requirements we already have determined, we have added 6 different functional requirements with their more inclusive explanations to the second iteration. Likewise, in addition to the non-functional requirements, we have added 2 different non-functional requirements with their measurable claims. Also, We have written the implementation as a non-functional requirement, and based on the feedback given, we decided that it was not a requirement and took it out of the non-functional requirement part of the analysis report.

In exit from the jail state diagram, we changed the diagram name to getting in and out of jail to make it more inclusive. We tried to add the scenarios that a player can get into jail. In addition, we tried to change state names to make the states of the player more clear.

For the sequence diagrams we have redesigned the diagrams completely according to the feedback given. We have firstly decided to focus on two specific events. The first was concerning the actions that happen when a player is in jail. The second one was concerning the property buying and building adding to the properties in the

game. We have used non-conventional and wrong ways of writing the diagram like writing ‘return money’ to message parts instead of writing real method names like ‘money = returnMoney()’ in the first iteration. But, in this iteration we have made sure that the sequence diagram follows the conventions that are used in the class. Also, at the earlier iteration we have chosen the scenario to be broad, namely we have tried to make a sequence diagram for a whole turn of the player. This was too broad and following the feedback given we have decided to focus on more narrow and detailed scenarios. We have also divided this section to two diagrams, so that we could be more precise with the scenarios considered.

Since we did not start coding the project during the first iteration, we noticed our class diagram has missing lots of classes that need to exist in this project and the features and methods included in these classes too by examining the feedback given to us for the iteration 1 report and the feedback we received during the presentation. We have added the newer classes such as special location, chance location and community chest location classes. We updated the class diagram with multiplicity relations.

In addition to the explanations above, we have added the numbers and captions of all the figures we have uploaded, appropriately and reuploaded the figures with higher resolution so that they can be read more clearly. Also, we have adjusted the font and sizes of the titles accordingly, and be careful about page margins as well.

5. References

[1] Object-Oriented Software Engineering, Using UML, Patterns, and Java, 2nd Edition, by Bernd Bruegge and Allen H. Dutoit, Prentice-Hall, 2004, ISBN: 0-13-047110-0.

[2] Monopoly Manual, https://www.hasbro.com/common/instruct/00009.pdf