Team: Tommy Trieu
William Diment
Title: Definitely Not Monopoly

Summary: Our goal is to present an interactive client version of monopoly, where users will be able to play the game of monopoly against each other.
User Requirements:

ID	Description	Priority
US-001	As a player, I want to be able to play with up to four of my friends	Critical
US-002	As a player, I want to be able to see the entirety of the gameboard	Critical
US-003	As a player, I want to be able to roll the proper amount of dice so the game is fair	Critical
US-004	As a player, I want to be able to move to a new spot on the board according to the dice roll I received	Critical
US-005	As a player, I want to be able to see what other properties users own	Critical
US-006	As a player, if I land on a spot with a property, I want to be able to purchase the property	Critical
US-007	As a player, I want to be able to trade my property to other players	Critical
US-008	As a player, if I can not pay rent or have enough property to sell, I want to be able to declare bankruptcy	Critical
US-009	As a player, I want to be able to be able to mortgage properties to the bank	High

US-010	As a player, I want to be able to land on the chance/community chest spot and receive a card from their respective piles	High
US-011	As a player, I want to be able to vote on a time limit with the other players	Medium
US-012	As a player, I want to be able to vote on adjusting the starting cash with other players	Medium
US-013	As a player, I want to be able to vote on whether or not properties can be auctioned off after a player declines to buy the property	Medium
US-014	As a player, I want to be able to play against a computer opponent	Low

Functional Requirements:

ID	Description	Priority
FU-001	Banking must be handled by program	Critical
FU-002	Banker will automatically distribution and transfer of cash	Critical
FU-003	Players should not be able to influence Banker - no cheating is allowed	Critical
FU-004	Every time a player passes go, the bank must distribute $200 to the player	Critical

FU-005	All transfers of property between players must go through the Bank first to prevent cheating	Critical
FU-006	A player must roll their dice and move to their new position on the board before making any action	Critical
FU-007	A player will automatically win if they are not the last player to declare bankruptcy	Critical
FU-008	If a player lands on a Chance/Community chest spot, a randomly generated number should determine which card they get	High
FU-009	If a player selects a Computer AI Opponent, the actions of the AI opponent must be random enough to not be guessable	Low

Business Requirements: No Business Requirements

Database: We plan on using a standard MySQL RDBMS for management of our data – we will be using it to store the relevant information for the property cards and tile information. In addition, we may look at later expanding the requirements to store a history of moves made by the player into the database.

Use Cases: *Tommy Trieu*

Use Case ID:	UC-04
Use Case Name:	View properties
Description	Player can view their own and other players' owned properties

Actors:	Players
Pre-Conditions:	The game has begun. Property information has been loaded. It is the viewing player's turn.
Post-Conditions:	The player has seen what properties each player owns.
Frequency of Use	Often

Flow of events:	Actor Action	System Response
1)	Click on 'View All Properties' from the main view	Chart of information on all properties is displayed.
Variations:	None	
Exceptions:	None	

Use Case ID:	UC-05
Use Case Name:	Mortgage properties
Description:	Player can mortgage any of their properties for an instant sum of cash from the bank, but forgo the ability to collect rent on mortgaged properties

Actors:	Player
Pre-Conditions	A player owns a piece of unmortgaged property that they wish to mortgage. It is the player's turn.
Post-Conditions	The player has received cash from the bank

	equal to the sum of the mortgage values of their selected properties. The selected properties are mortgaged and unable to collect rent.
Frequency of Use	Almost always at least once per match, often more, by each player.

Flow of Events:	Actor Action	System Action
1)	Player selects ‘View my properties’ from main view	Window containing list of player-owned properties is displayed
2)	Player clicks on property that they wish to mortgage	Window containing detailed information of selected property is displayed
3)	Player clicks on ‘Mortgage’ from property information view	Confirmation to mortgage property for the appropriate mortgage value is displayed
4)	Player accepts mortgage confirmation prompt	Property becomes mortgaged, bank disburses cash to player, window containing list of player-owned properties is displayed
Variations	None	
Exceptions	Player has no properties to mortgage	

Use Case ID	UC-06
Use Case Name:	Declare bankruptcy
Description:	Player must declare bankruptcy if they owe rent or taxes that they cannot pay.

Actors	Players
Pre-Conditions	The player has just moved to a space such that

	they owe money either to another player or the bank. It is still the player's turn.
Post-Conditions	The player is declared bankrupt and removed from play. The bankrupt player's assets are transferred to the collecting player or bank accordingly.
Frequency of Use	Up to the number of players per game

Flow Of Events	Actor Action	System Action
1)	Player clicks 'Pay' button	The player's asset value is calculated and a prompt warns the player that there are insufficient funds. The 'Pay' but is changed to read 'Declare Bankruptcy'.
2)	Player clicks on confirmation of the 'Insufficient funds' warning	Main view is displayed
3)	Player clicks 'Declare Bankruptcy' button	Prompt asking player to confirm selection appears
4)	Player clicks 'Yes'	The bankrupt player's properties and cash are transferred to the payment collector. The bankrupt player is removed from play and their turn is ended.
Variations:	None	
Exceptions:	Player has enough funds to pay rent/tax	

Use Cases: *William Diment*

Use Case ID:	UC-01
Use Case Name:	Buying Property
Description	After moving to a new spot on the game board, a player has the option of buying property

Actors:	Players
Pre-Conditions:	A player has yet to roll dice and move from their original position to their new position
Post-Conditions:	The player has moved to a new position and bought the piece of property associated with that position
Frequency of Use	Often

Flow of events:	Actor Action	System Response
1)	Actor rolls the dice	Two Random numbers between 1-6 generated, added together
2)	Actor moves from original spot to new spot	
3)	Actor sees that the property is available to buy	
4)	Actor gives money to bank	Bank takes money from Actor
5)	Actor receives property	Bank gives property to actor
Variations	4) Actor does not have enough money to buy property	Bank does not take money from actor
Exceptions:	None	

Use Case ID:	UC-02
Use Case Name:	Trade Property to Player
Description:	After moving to a new spot, a player wishes to trade a piece of property with another player

Actors:	Player
Pre-Conditions	A player owns a piece of property that he wants to trade
Post-Conditions	The player has traded his piece of property, losing the property he had but gaining the property of the other player
Frequency of Use	Sometimes

Flow of Events:	Actor Action	System Action
1)	Player selects property he wants to trade	
2)	Player selects other player to trade with	
3)	Player requests property from the other player	
4)	If other player accepts, they swap properties	System takes the properties from the players to give to the other players
Variations	4. If other player does not accept, they do not swap properties	System does not swap properties
Exceptions	None	

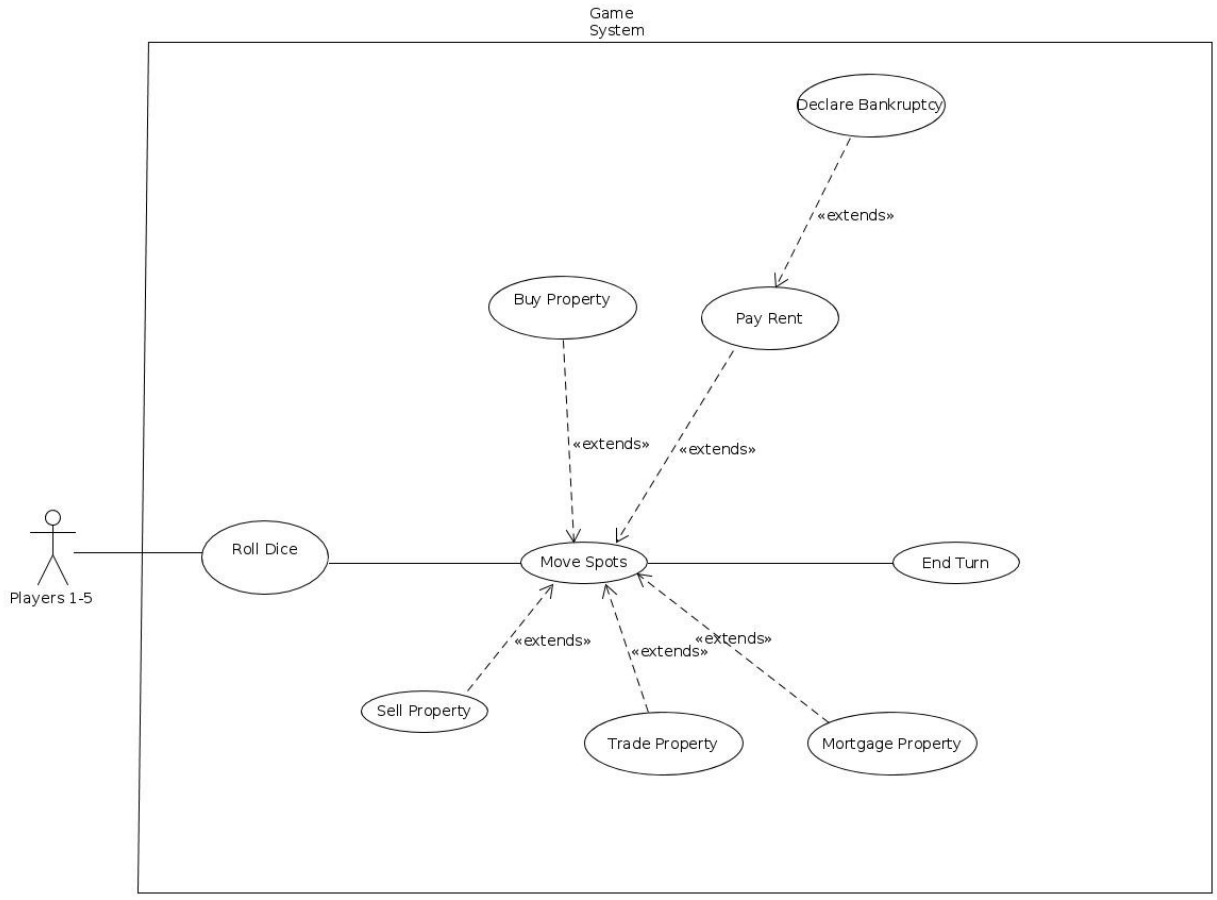
Use Case ID	UC-03

Use Case Name:	Set Time
Description:	Before the game begins, all players will have the option of voting to set the time of the game

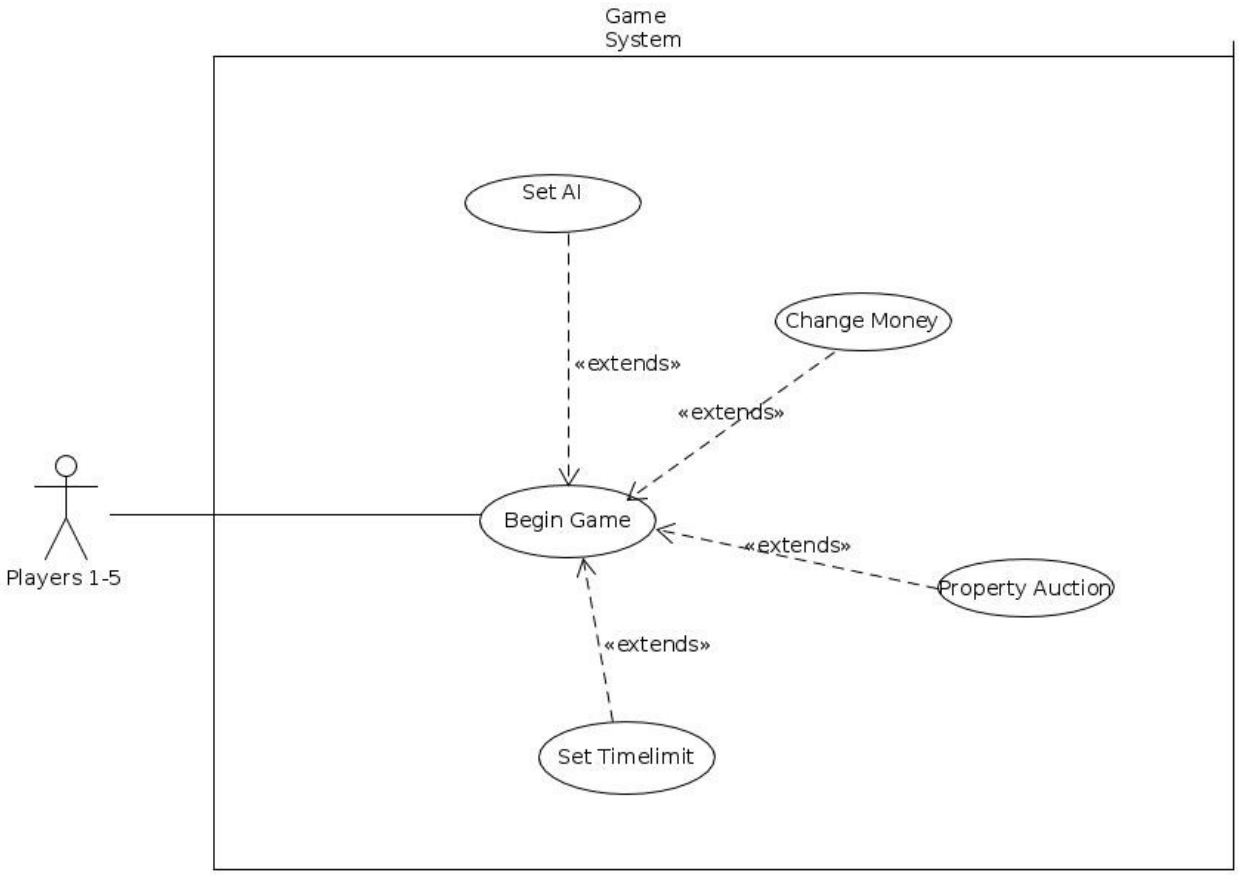
Actors	Players
Pre-Conditions	Game has not begin and a time has not been set for the game
Post-Conditions	A time has been set for the game
Frequency of Use	Sometimes

Flow Of Events	Actor Action	System Action
1)		Display option “Would you like to set a time?”
2)	Actor picks yes	Record vote
3)		Display time options
4)	Actor picks time option	Record time option vote
5)		Tally votes, set time
Variations:	2) Actor picks no	Record vote
Exceptions:	None	

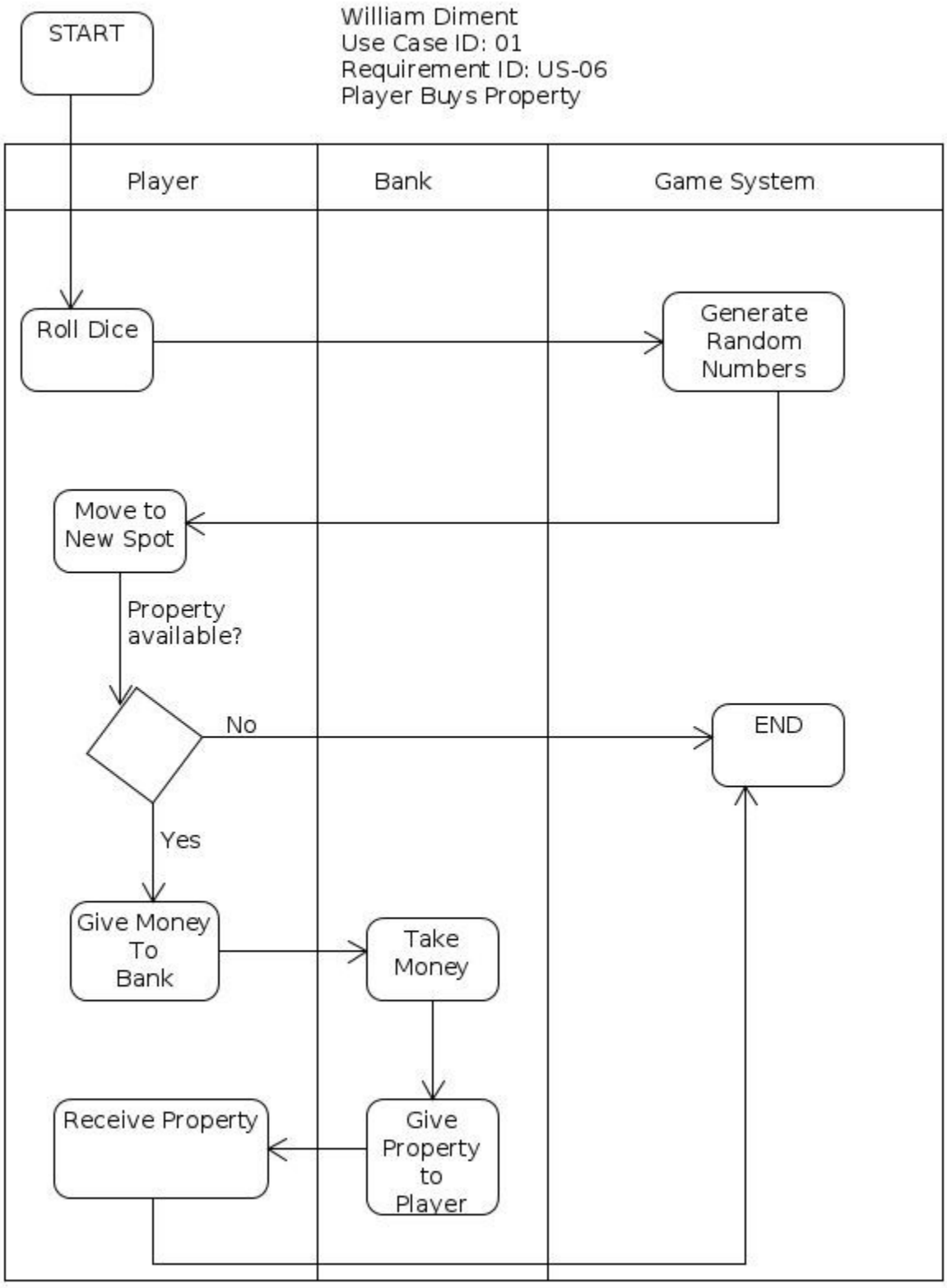
Use Case Overview



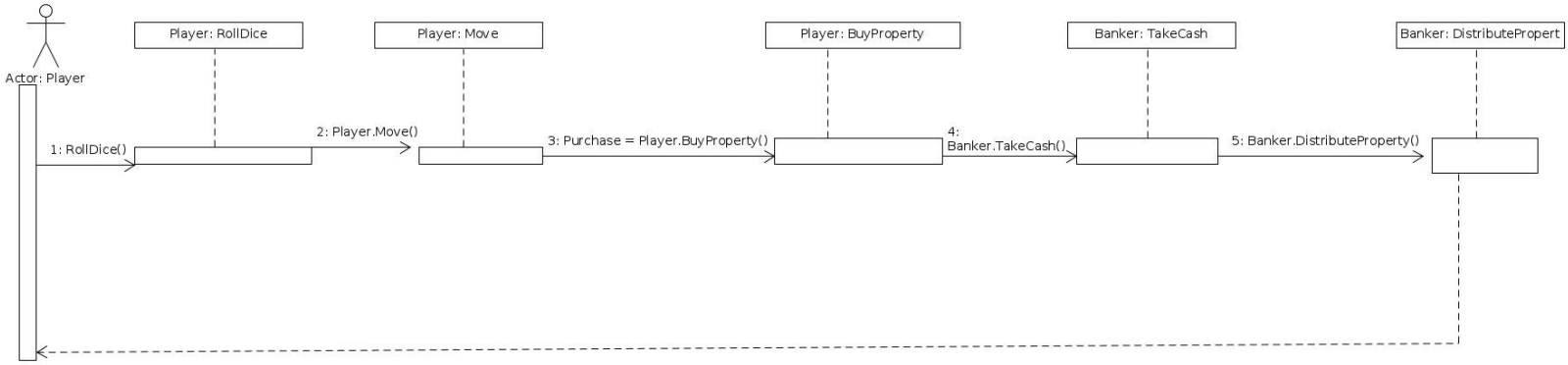
Use Case for the beginning of the game



Activity Diagram: *William Diment*



Sequence Diagram: *William Diment*



Class Diagram