Object-Oriented Project Part 2 - GROUP

Team: Nolawee Mengist
Shawn Polson
Theerarun Tubnonghee (Steve)
Jose Escobar

Title: Blackjack!

Project Summary: A replication of the Blackjack card game that a user will play against a computer dealer and (optionally) computer players. Our goal is to create a program that looks and feels like an exact digitization of the real game; our code will faithfully represent Blackjack and our computer dealers/players will play autonomously.

Project Requirements:

Business Requirements: *We have omitted the "Business Requirements" table here because we do not have any.*

User Requirements				
ID	**Requirement**	**Topic Area**	**Actor**	**Priority**
UR-001	As a player, I can view the rules of Blackjack.	(optional)	(optional)	(optional)
UR-002	As a player, I can pick how many computer players to play with.			
UR-003	As a player, I can be dealt cards			
UR-004	As a player, I can place bets using chips			
UR-005	As a player, I can know the total amount of "money" I own.			
UR-006	As a player, I play against a CPU dealer			
UR-007	As a player, I can "hit" during my turn.			
UR-008	As a player, I can "stand" during my turn.			

UR-009	As a player, I can “double” during my turn.			
UR-010	As a player, I can “split” during my turn if my first two cards are of the same denomination.			
UR-011	As a player, I can win the bet if I get “21”			
UR-012	As a player, I lose the bet if I “bust” over 21.			
UR-013	As a player, I can play through as many rounds as I want.			
UR-014	As a player, I can still lose to the dealer even if I get 21			
UR-015	As a player, I can drop out and end the game with my winnings any time			
UR-016	As a player, I can save the current state of the game and continue to play.			
UR-017	As a player, I can continue my saved state of the game			
UR-018	As a player, I can start a new game and play with some amount of chips.			

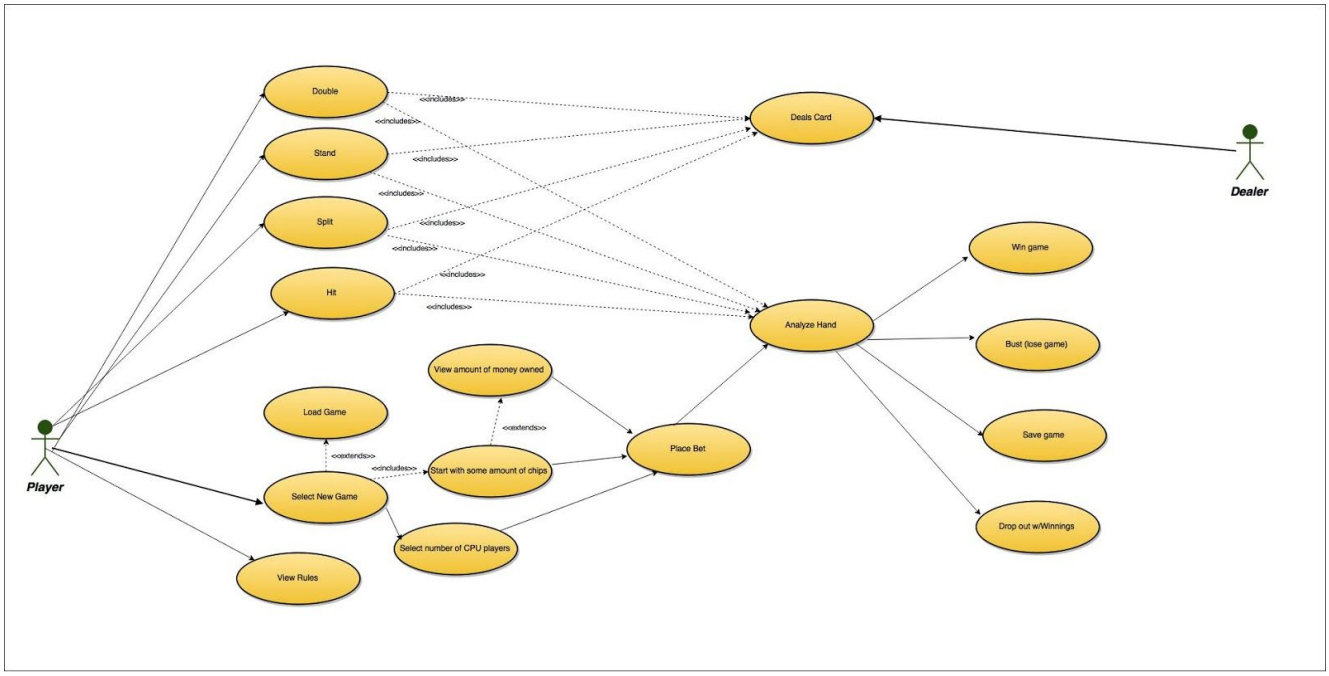
Functional Requirements				
ID	**Requirement**	**Topic Area**	**Actor**	**Priority**
FR-001	As a player, I lose if I run out of chips.	(optional)	(optional)	(optional)
FR-002	Card deck should be shuffled before play or when deck runs out			
FR-003	Dealer and computer players must			

	run through their turns when needed			
FR-004	Computer players must track their own total money and not bet past 0			
FR-005	Dealer and computer players must not hit when their currentTotal>=16			
FR-006	When player saves the game and exits, all totals and current cards in deck must be saved for reloading			
FR-007	User can navigate through the game.			

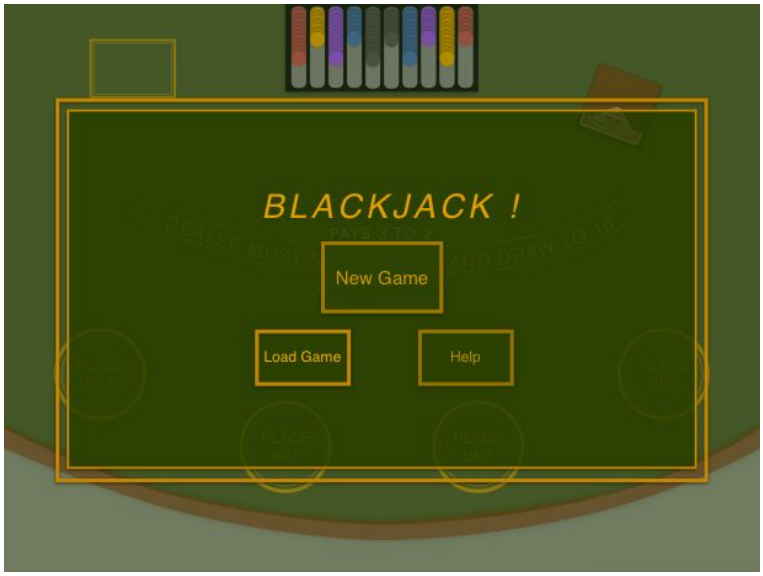
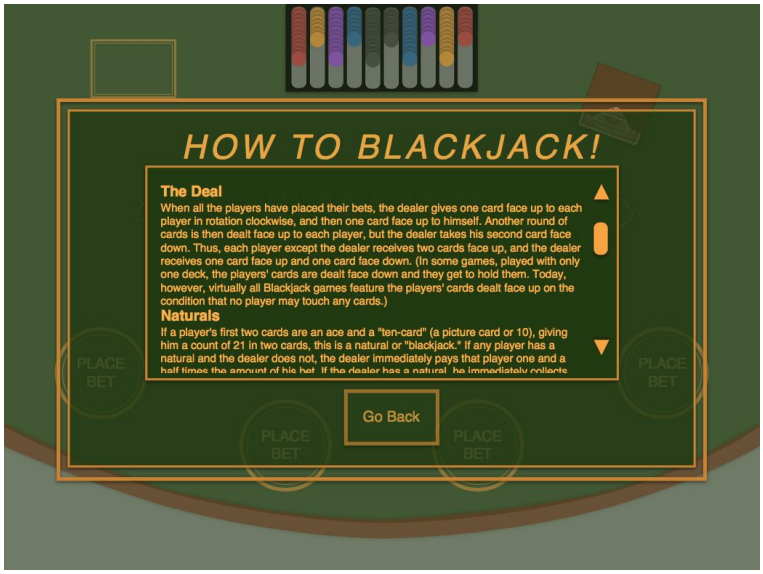
Non-Functional Requirements				
ID	**Requirement**	**Topic Area**	**Actor**	**Priority**
NFR-001	The game can be played on the command line before our GUI is implemented.	(optional)	(optional)	(optional)
NFR-002	The game should work properly each time it's opened			
NFR-003				

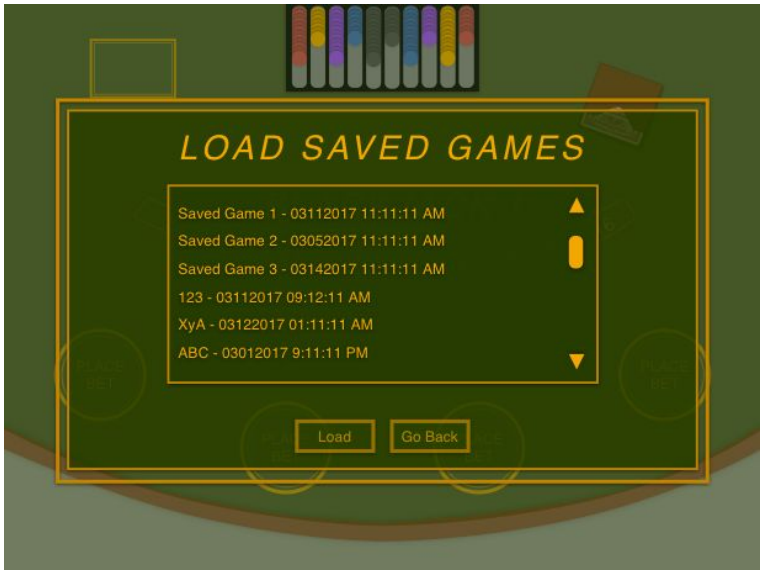
Use Cases:

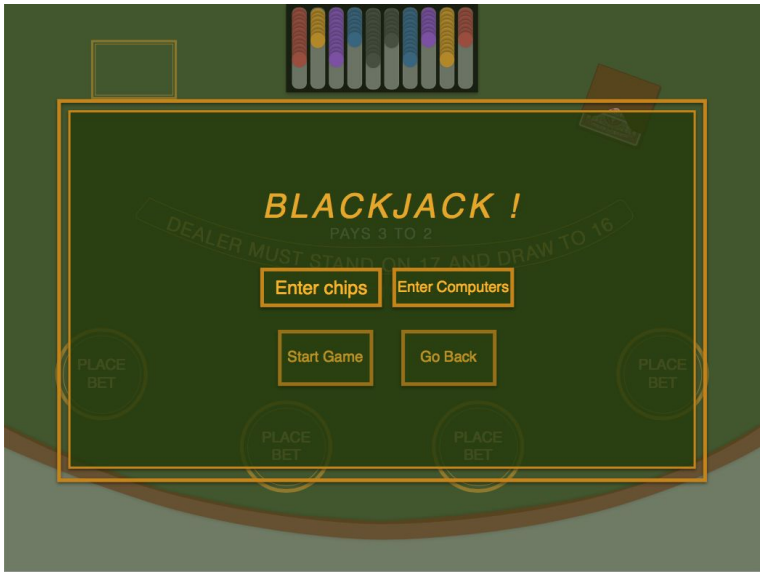
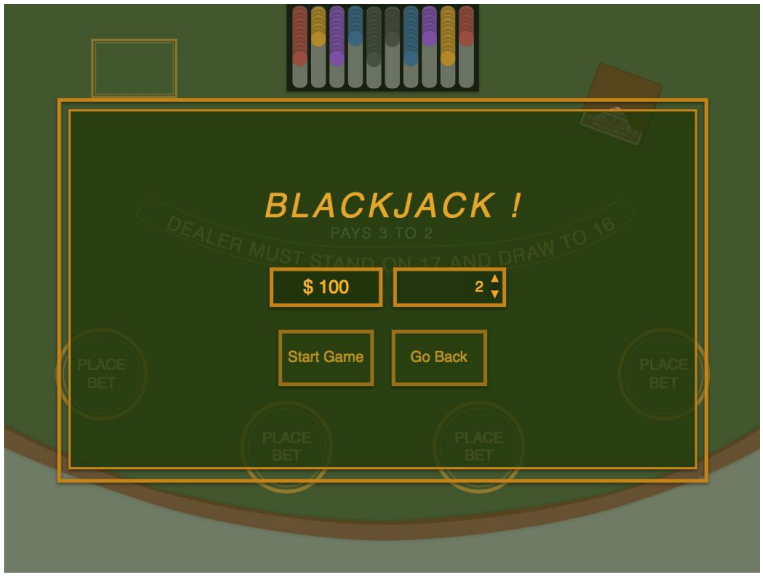
- **Use Case Overview:**

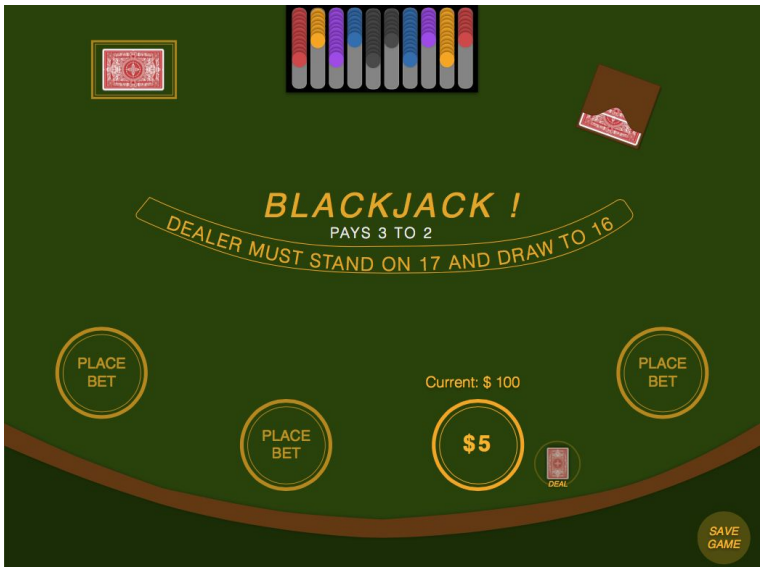


UI Mockups:

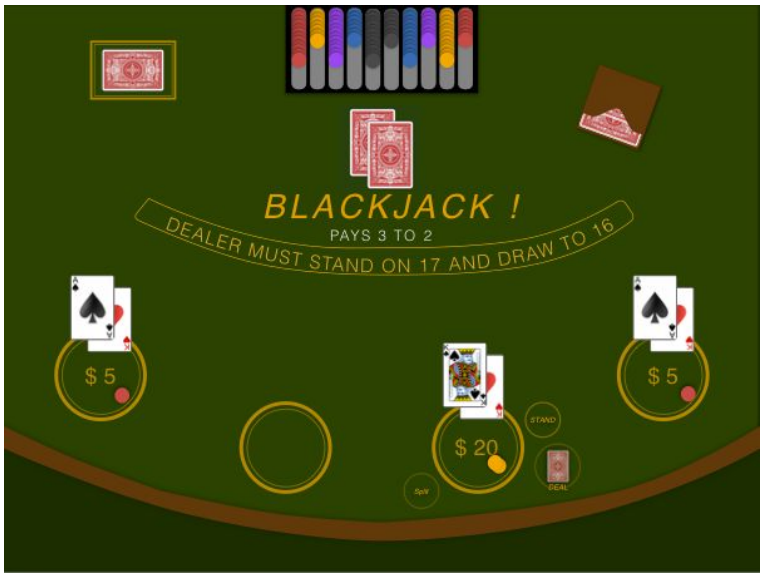
	Scenario	User Interface Response
1.1	**Player opens Blackjack! Program. The UI displays Main Menu.**	
1.2	**Player clicks Help button. The UI displays rules of Blackjack!**	

1.3	**Player clicks Go Back to Main Menu. Then, Player decided to load saved games by clicking Load Game. The UI displays Load Saved Game Menu**	
1.3.1	**Player clicks on one saved game from the list, and clicks Load. The system loads the saved one and UI display the current state of saved game.**	

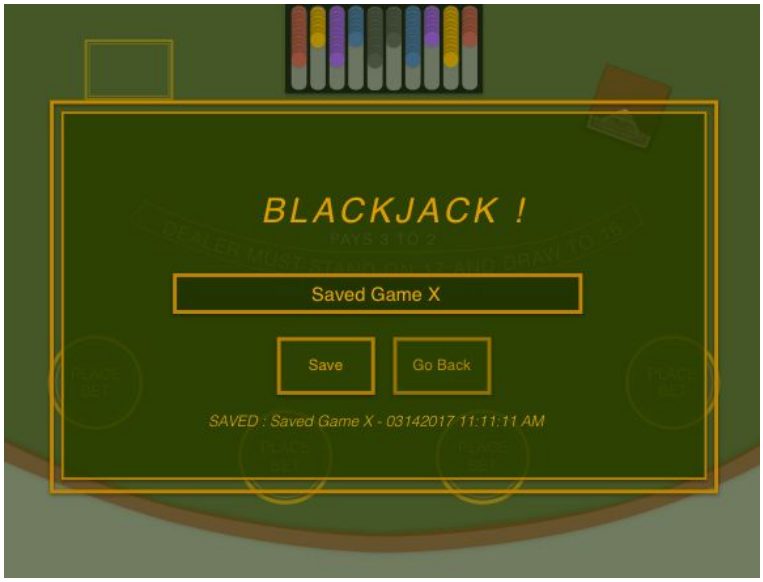
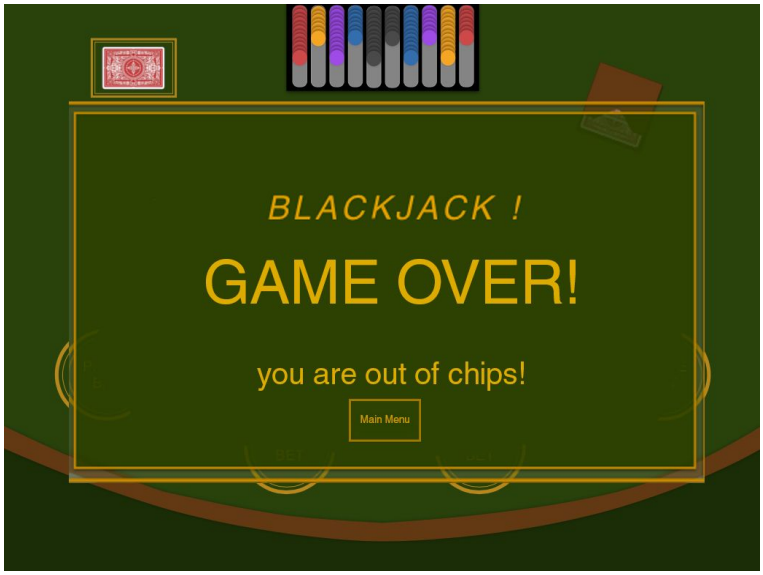
2.1.1	**In Scenario 1.1, Player clicks New Game to start a new game. The UI displays Options' requirement to start a game. Prompt Player to enter amount of starting chips and a number of computer players to play with.**	
2.1.2	**Following, Player enters desired options. The UI real time responses back the entered options. If Player clicks Start Game, the UI will retrieve the information entered, and displays the empty table in 2.2.1. If Player clicks Go Back, the UI displays Main Menu.**	

2.2.1	**Player is in the first game. The Table will be empty. The UI displays current amount of chips/money Player owned.**	
2.2.2	**After Player clicks PLACE BET, the UI prompts Player to enter the bet using keyboard. The UI displays the amount of bet, and prompts Player to start a round by clicking DEAL.**	

2.3	**Promptly, the dealer distributed two cards to each player according to the rules. The UI will displays the Table with cards in hands of players but not displays dealer's hand if the dealer is not blackjack!**	
2.3.1	**If the dealer has blackjack!, the game is end. The UI displays dealer's hand. Then the dealer claims and pays players. Promptly, the next round start (bring back 2.2.2).**	

2.3.2	**If the dealer does not have blackjack!, each player can deal more or stand or double. The UI displays options of the Player.**	
2.3.3	**If there is an empty seat, HumanPlayer can split the card if the dominating cards are the same number. The UI displays options of the Player.**	

2.3.4	**After splitted, Player can stand/deal, and is need to confirm options on both hands by clicking on stands. The UI displays options of the Player. Then the dealer reveals his hand and claim/pay players. Promptly, the next round start (bring back 2.2.2).**	
2.4.1	**During each placing bet state (2.2.2), HumanPlayer will be allowed to save the game. The UI displays SAVE GAME button to be clicked as an option.**	

2.4.2	**Promptly, after Player clicked SAVE GAME (in 2.4.1), the UI display Saving Game menu. Player can enter the desired title of saving game, and click Save to complete the process. Player can click Go Back, Player will be brought back to the current game (in 2.4.1). Caution: The Table will not save the game unless Player click Save.**	
2.5	**If unfortunately you run out of chips, game is over! The UI responses GAME OVER! Player can click Main Menu.**	

Data Storage: *We do not have plans for using a database. We plan on implementing "save game" by saving/loading our model (table) object; to have that be the object that persists).*

Class Diagram: