Group 2 (Allen, Exley, Farah, Goff, Kershaw, Lysaght)
April 6th, 2020
Requirements, Part III

Class Specification Document

Introduction:
This document sets to expand on the classes in the class diagram for Group 2's project for SWE 3313. Each class has its own purpose in the project and that will be specified in the sections below. Important things, like what the class is supposed to do and how it will perform that, are in this document. How each class interacts with each other through their relationships will also be expanded upon. The chart below with list everything that's required in the program's code like methods, constructors, and values.

Class Specification:

Class Name:	Constructor:	Values and Properties:	Methods:
Card	+Card (Suit: String, Value: String) ~creates a card with a suit and value.	-Suit: String -Value: String ~Suit and value are stored as strings	-GetSuit() -GetValue() ~Methods that return the suit and value respectively. -IsDraggable() ~Boolean to determine if the card can be moved.
Board	+Board() ~Creates a board	-Cards[] Card ~Array of Card objects. -Tableaus[] Tableau ~Array of Tableau objects -Foundations[] Foundation ~ Array of Foundation objects.	ShuffleCards() ~Shuffles all the cards. ~Places them all in every tableau with 4 in each deck. ~Moves kings to the back of every tableau deck. Restart() ~Starts the game over.
BoardSlot **PARENT**	+BoardSlot() ~Creates a BoardSlot	-Deck[] Card ~A deck of cards.	-SetCard(Card) ~Sets the card in the deck array.
Tableau **CHILD**	+Tableau() ~Creates a Tableau.	-Deck[] Card = 4 ~A deck of cards	-SetCard(index, Card)

			~Sets the card to a specific index in the deck array.
Foundation **CHILD**	+Foundation(Suit) ~Creates foundation with a specific suit.	-Deck[] Card ~A deck of cards.	-SetCard(Card) ~Sets the card at the end of the deck array. -GetSuit() ~Returns this foundation's suit.
GameMaster	GameMaster() ~Creates a GameMaster object.	-PlayerWin bool ~A bool that will determine if the player has won. -MoveCount int ~An int that counts moves.	-IsMoveLegal() ~A method that returns a bool. -PlayerDrag() ~Allows the player to move the cards. -PlayerWin() ~If the conditions are met the player wins the game.

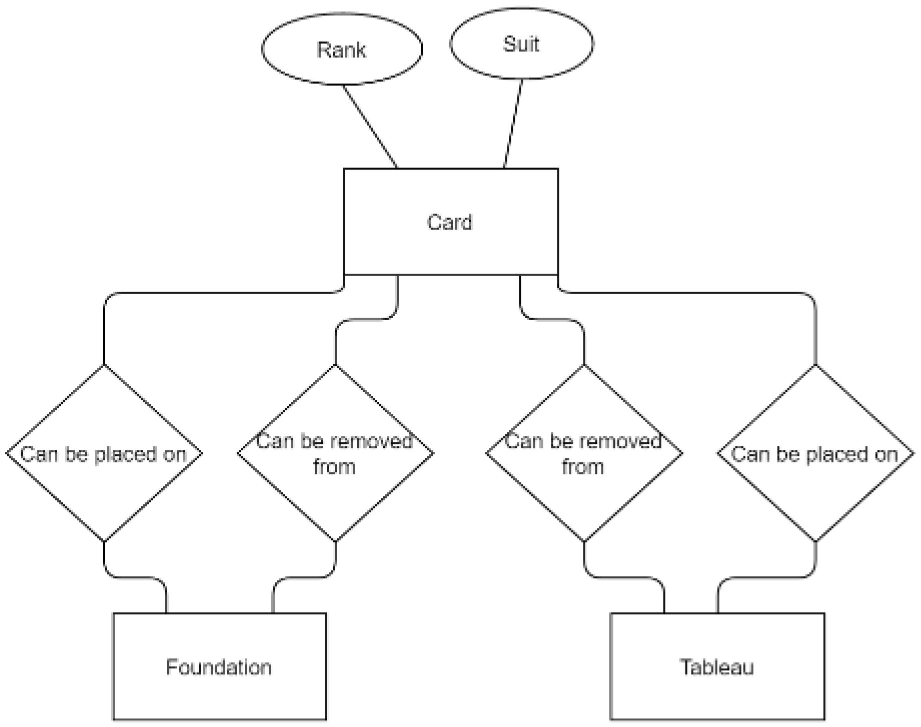
Class Descriptions:

- ### Card:
  - This class represents an individual card.
  - Each card is represented with a value like ace, queen, 10, 2, etc… and a suit like, Spade, Heart, Clubs, and Diamonds.
  - The class has to use a get method that will return the value and suit.
  - The constructor has to require both a suit and a value, so they can be set with those values when created in the board class.
- ### Board:
  - Creates array of cards with their own suit and value.
  - The board class will create and use objects Tableau, foundation, and card.

  - The board will represent the gameboard that has 13 Tableaus, 4 foundations, and 52 cards. That is how many objects will be created onto of the board object.
  - The Board class will have a relationship with the GameMaster class to allow the player to make moves and determine if the move is legal.
  - The Card Shuffler class will shuffle and distribute all cards on the board.
- BoardSlot:
  - The parent class to the tableau and foundation classes, with the basic function of being able to take in cards and store them on a stack.
- Tableau:
  - Has a max limit of 4 cards and is filled at the beginning of the game.
  - 13 of them will be created on the board.
- Foundation:
  - Can hold all cards of a specific suit.
  - Starts the game empty.
  - Game is won when all ace’s and kings are moved into foundation decks.
- GameMaster:
  - Handles player input.
  - Handles the legality of moves.
  - Determines if the player has won or lost the game.
  - To do this the GameMaster class must be able to check on the conditions on the board.
  - Handles GUI usability.

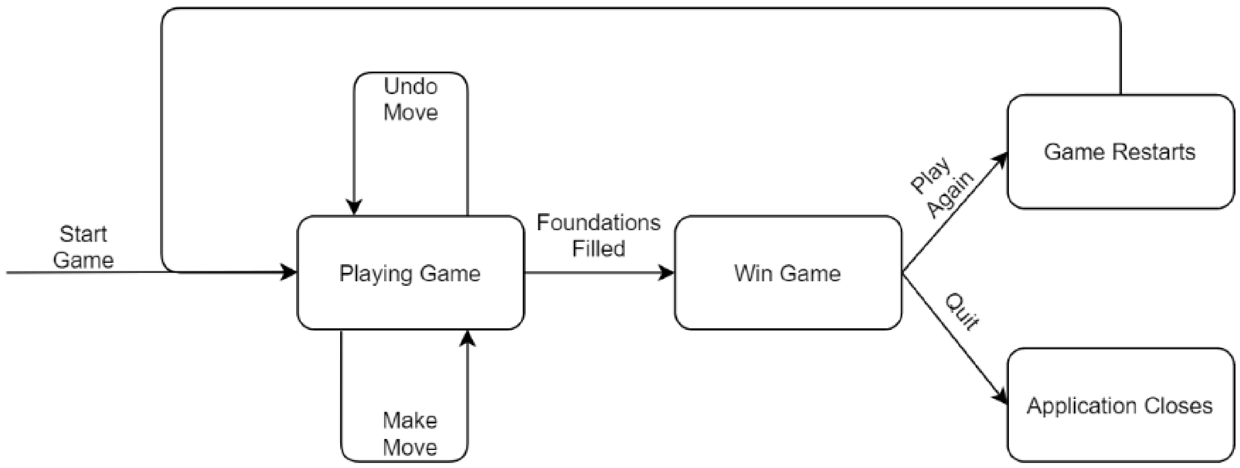
Group 2 (Allen, Exley, Farah, Goff, Kershaw, Lysaght)
April 6th, 2020
Requirements, Part III

Game Element Entity-Relationship Diagram



Group 2 (Allen, Exley, Farah, Goff, Kershaw, Lysaght)
April 6th, 2020
Requirements, Part III

Game State Transition Diagram


Undo Move
Start Game
Playing Game
Foundations Filled
Win Game
Play Again
Game Restarts
Quit
Application Closes
Make Move