Multiplayer Yahtzee With a Graphical User Interface

Final Report

That One Team

Adam Kowalchyk, Daniel Hoberman, Maya Fleming, Trevor Diuco

Course: CPSC 224 - Software Development

I. Introduction and Project Description

This project is a simulation of the game Yahtzee. This is a game that allows multiple players to compete with one another through a game of dice rolling. Players begin their turn by rolling 5 dice, each with 6 sides. Once the dice are finished rolling, the player can choose which dice they would like to keep, and which dice they would like to roll again. Each player is allowed 3 attempts to roll their dice. After the player either chooses to keep all their dice or their three rolls have been used, they must decide which category they would like to apply their dice roll to. The scorecard has two sections, and upper and lower. The upper section is where you can record the scores of the number of dice that show the numbers one, two, three, four, five, or six. The lower scorecard consists of several scores that resemble different things: three of a kind, four of a kind, full house, small straight, large straight, Yahtzee, and chance. The game ends after 13 turns. The goal is to have all the slots filled on the scorecard, but if you are unable to fill a slot during a turn, you must cross it out. At the end of all thirteen turns, the players add up their scores of the upper and lower scorecard. The upper scorecard includes a bonus of 35 points, which can be applied if the upper scorecard is 63 or larger. The person with the highest combine score of the upper and lower scorecard wins.

Concerning the Java code, there are several classes implemented to create the game. The main classes used in our project are Dice, Hand, Rules, Scorecard, Player, and Yahtzee game. The Dice class creates an object for the dice which are being rolled. The Hand class represents an object for the hand which rolls the dice. The Rules class lays out all the rules for the game, so the player knows how to play before starting. The Scorecard class illustrates the scorecard, which keeps the score for the players after each turn. The Player class controls the different players in the game and allows the Yahtzee project to be multiplayer. The Yahtzee class combines all the classes to play the game, and the GUI commands to make the game visual. The final product will resemble a real-life digital Yahtzee game where you can visualize the dice, pick which ones to keep and roll again, and choose where you want to apply your dice roll.

The purpose of this document is to showcase the progress made on our Yahtzee GUI design. Throughout developing our game, we encountered obstacles with GitHub, Java Swing, and multiple design interface issues. To best correct these obstacles, we utilized tools like GitHub Branching and better-utilized team communication skills. Overall, our team feels proud about our final game result, as it accomplishes all of the functional and non-functional requirements we had as well as showcases a simple and well-made multiplayer Yahtzee game.

II. Team Members - Bios and Project Roles

Trevor Diuco:

Trevor Diuco is an accounting student with a computer science minor. His interests include mobile app development and automation. This summer he will be interning with Deloitte, working with a group that specializes in tax management system development and integration. His roles in this project included GitHub administration and implementation of the

player naming functionality. Trevor's skills include C++, Python, Java, HTML, CSS and JavaScript, and SQL.

Daniel Hoberman:

Daniel Hoberman is a computer science major interested in web development, and video game development. His prior projects have included http://net-rocket.herokuapp.com, https://grocery-calc.herokuapp.com (this ones kinda old and when I wasn't very good), and a Visualgo copy using c# and Unity. His roles in the project included development of the final scorecard, design of the sorting algorithm, and some of the development of the multi-player game functionality. Daniel's skills include C++, C#, JavaScript, ReactJS, MongoDB, NodeJS, Python, Unity, Java, and Jupyter Notebook.

Adam Kowalchyk

Adam Kowalchyk is a computer science major interested in data science and web development. His prior projects have included data science projects as well other school coding assignments. Adam's skills include C++, Python, Java, and Jupyter Notebook. For this project his responsibilities included the developing the original design, working on changing players, and implementing the new GUI Scorecard.

Maya Fleming:

Maya Fleming is a computer science major interested in cybersecurity and web development. Some of her skills include C++ and Java. In this project, her roles were developing the end panel of the game, which included a sorting algorithm of the player's scores from 1st place to last.

III. Project Requirements

Functional Requirements:

0. User should be able to start the program

Part	Start of the Program
Priority	High
Purpose	The Application shall allow a player to begin the program by clicking on the play button

Inputs/needs	NA
Operators/ Actors	Actors include the player that presses the game to play. Once the play button is clicked, a new Rules Class is instantiated.
Outputs	The Rules Class should showcase a GUI dropdown box allowing the player to select the number of players

1. Program should prompt the user to choose the number of players that will be playing

Part	Selecting number of players
Priority	High
Purpose	The Application shall allow the user to select the number of players they wish to play the game
Inputs/needs	A GUI Drop down box showcased to the user generated by the Rules Class
Operators/ Actors	Actors include the user that selects the number of players as well as the submit button which holds an action listener. Classes include the Rules class which generates the drop-down box as well as the YahtzeeGame class which is generated when the number of players is selected.

Outputs	The Number of players is saved to the Rules class, and a new Yahtzee Game is started, showcasing the game interface to the players.

2. Start of program ask for rules (Wish list)

Part	Ask user for Lizard-Spock Yahtzee Rules
Priority	Low
Purpose	The Application shall allow the user to select the number of sides on a die, number of dice in a hand, and number of roles per turn.
Inputs/needs	A GUI Drop down box showcased to the user generated by the Rules Class
Operators/ Actors	Actors include the user that selects the rules as well as the submit button which holds an action listener. Classes include the Rules class which generates the drop-down box to select the rules as well as the YahtzeeGame class which is generated when the submit rules button is clicked.

Outputs	The number of sides on a die, number of dice in a hand, and number of rolls per turn selected by the user is saved to the Rules class. A new Yahtzee Game is started, showcasing the game interface to the players.

3. Display the current 5 6-sided dice hand as images in a frame

Part	Display hand in a frame
Priority	medium
Purpose	The Application shall display the current 5 6-sided dice hand as images in a frame for each player when it's their turn.
Inputs/needs	The current hand of the player stored in the HandOfDice class is needed to translate into a GUI representation. This translation to the GUI representation is stored in the HandView Class.
Operators/ Actors	Actors include the player's hand of dice that is displayed. Classes include: the HandView Class which displays the current player's hand to the Screen, the HandOfDice class which stored the current hand, The YahtzeeGUI class which displays the current hand to the window.
Outputs	The current players hand should be displayed to the frame as images in a panel.

4. Click a button to display the current state of a scorecard including bonus and totals.

Part	Show Scorecard on Demand
Priority	HIgh
Purpose	The application shall be able to display their current scorecard when the player wishes.
Inputs/needs	The PlayerGUI will need inputs from the player's Scorecard in addition to the attributes from Player.
Operators/ Actors	PlayerGUI will drive this task in addition to the attributes from the Player class.
Outputs	The Scorecard GUI should update its values with the most current values from the Scorecard class. Those values are then displayed in a new window of the GUI.

5. Click button to select dice

Part	Select Dice
Priority	High
Purpose	Player needs to be able to select the die they wish to hold in their hand in order to properly score the hand.

Inputs/needs	Select Dice will need to have an action event listener which tells the Button when it has been clicked or not.
Operators/ Actors	When the button has been clicked and the input has been confirmed by the player, handOfDie will be updated with the appropriate value.
Outputs	Ideally, we would like the die to change color so that the player knows which dice they had just selected. The kept dice are passed to the hand of Die and non-selected dice are rolled again.

6. Click button to roll new hand

Part	Roll new hand
Priority	High
Purpose	Both at the change of turn and the change of round, a new hand is rolled for the player to start new. Or if the player does not want any of the die, a new hand is rolled.
Inputs/needs	Roll new hand will not need any inputs besides an action listener to tell when to roll a new hand,

Operators/ Actors	Roll new hand method will iterate through the ArrayList of die and call a random function in order to get a new value.
Outputs	The output is a new hand with brand new die values than prior to the event.

7. Display possible scores

Part	Display possible scores
Priority	High
Purpose	Shows the player what their current hand has the possibility of scoring.
Inputs/needs	This method needs HandOfDie as well as the Scorecard class.
Operators/ Actors	This method will call the individual lines of the scorecard methods in order to calculate potential scores. It will then update the ScorecardGUI with those potential values so that when queried to show the scorecard, it will have the updated potential values.
Outputs	Those values are passed to ScorecardGUI and the mainGUI to display to the user when Player is asked to score.

8. Properly keep score

Part	Properly keep score
Priority	high
Purpose	The application shall properly keep track of score for each player in the game. After each turn will update the player score.
Inputs/needs	A GUI list of options for scoring each round, for each player.
Operators/ Actors	Actors include the users that selects the scoring action (ie. full house, yahtzee, etc.). Classes involved are Player, Scorecard, and YahtzeeGUI to help keep track of each player.

9. Compare player scores to determine winner

Part	Compare player scores to determine winner
Priority	medium
Purpose	The application shall properly keep track of score for each player in the game. At the end of the game application must compare each player's total score (with bonus included). The highest score must be declared as the winner of the game.
Inputs/needs	A GUI button for the players to end the game.

Operators/ Actors	Actors include the users that selects the scoring action (ie. full house, yahtzee, etc.). Classes involved are Player, Scorecard, and YahtzeeGUI to help keep track of each player.

10. Show all scores with total score at the end
11.

Part	Compare player scores to determine winner
Priority	medium
Purpose	When the game is over a GUI displays all players' total scores, and the winner should have an individual spot, showing who the winner is.
Inputs/needs	NA. Should be a clean screen, just showing scores.
Operators/ Actors	Actors include the users that selects the scoring action (ie. full house, yahtzee, etc.). Classes involved are Player, Scorecard, and YahtzeeGUI to help keep track of each player. When the game is over it should automatically display the final score.

Non-Functional Requirements:

- User is able to name each of the players in the game
- The final screen displays the final scores of each player as well as the order of standing
- Red highlight on dice selected
- Displays scorecard on main screen
- Shows players names during gameplay.

IV. Solution Approach

Creation of the YahtzeeGUI class

- For each player in the game, YahtzeeGUI is instantiated which gives the player their own screen when it is their turn.
- This was crucial for the implementation of multiplayer functionality

Addition of setPlayerNames method in Rules class

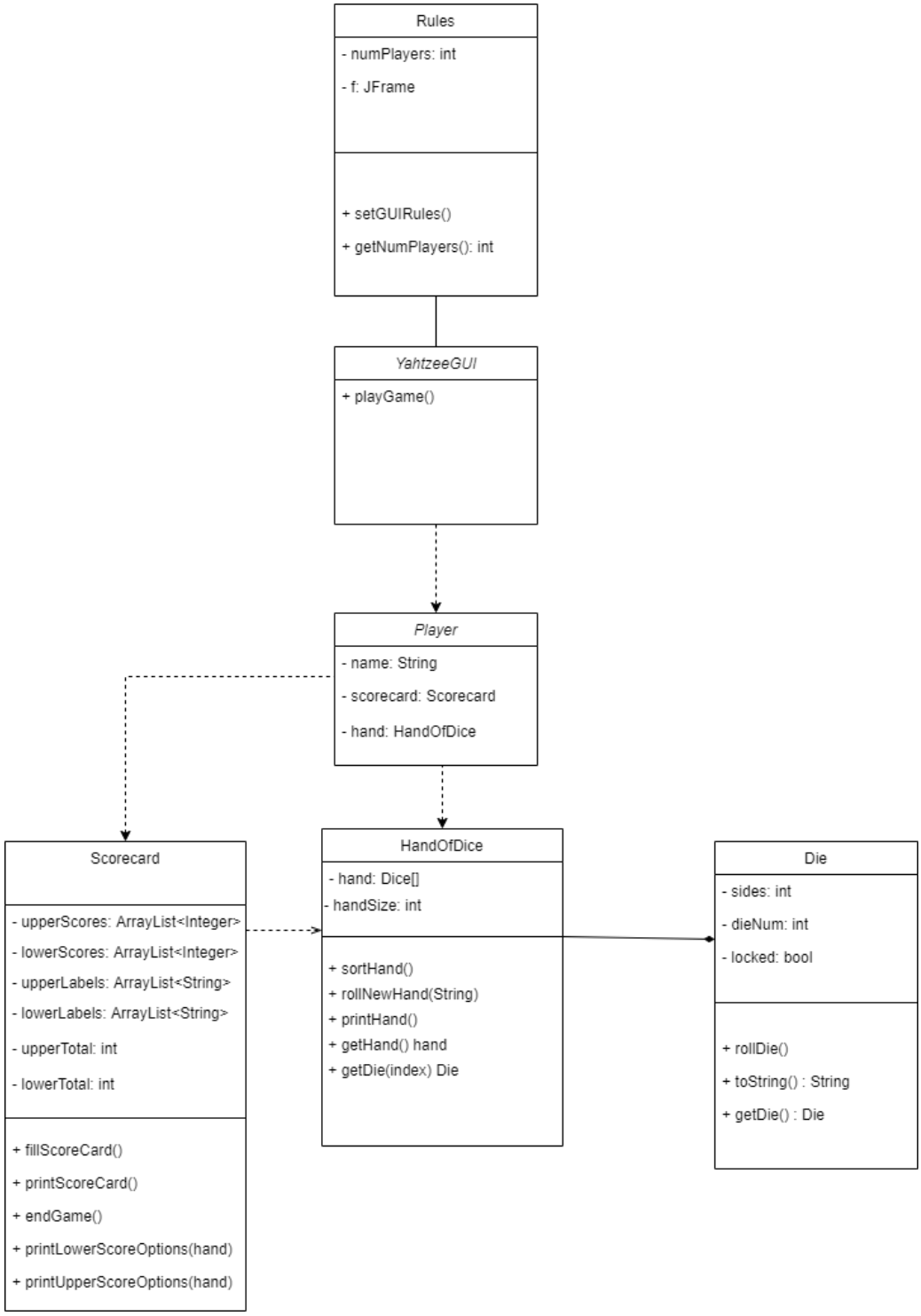
- Runs the code to prompt the user/s to name their player at the beginning of the game
- Result is passed to YahtzeeGUI to be used further on in the program

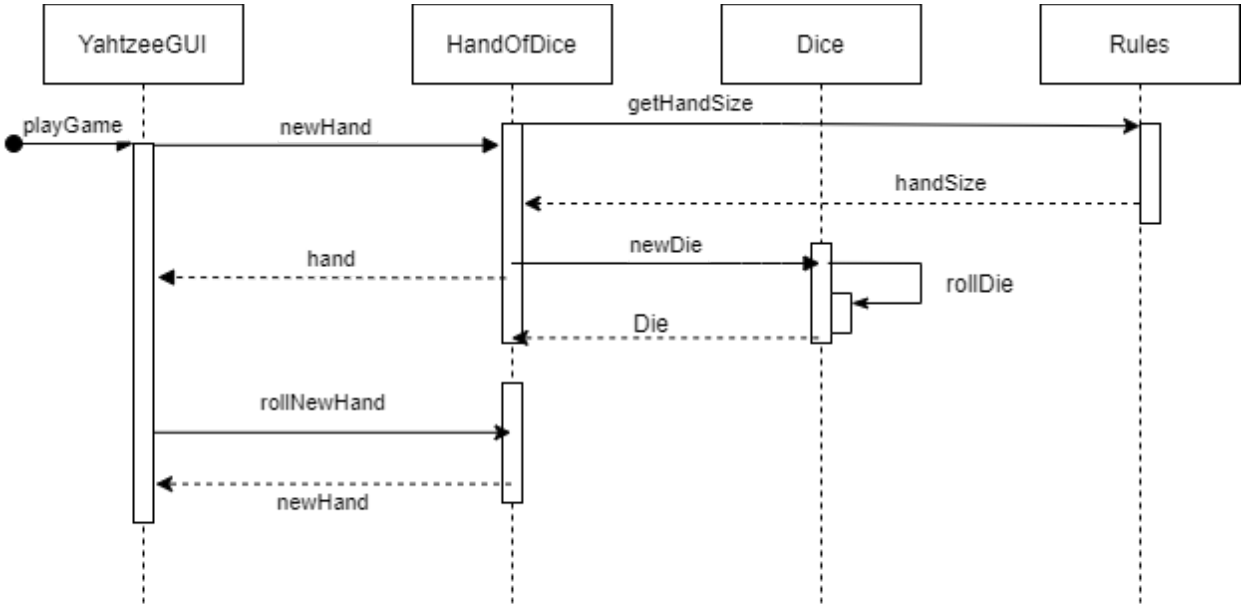
Make visible player's scorecard on their YahtzeeGUI

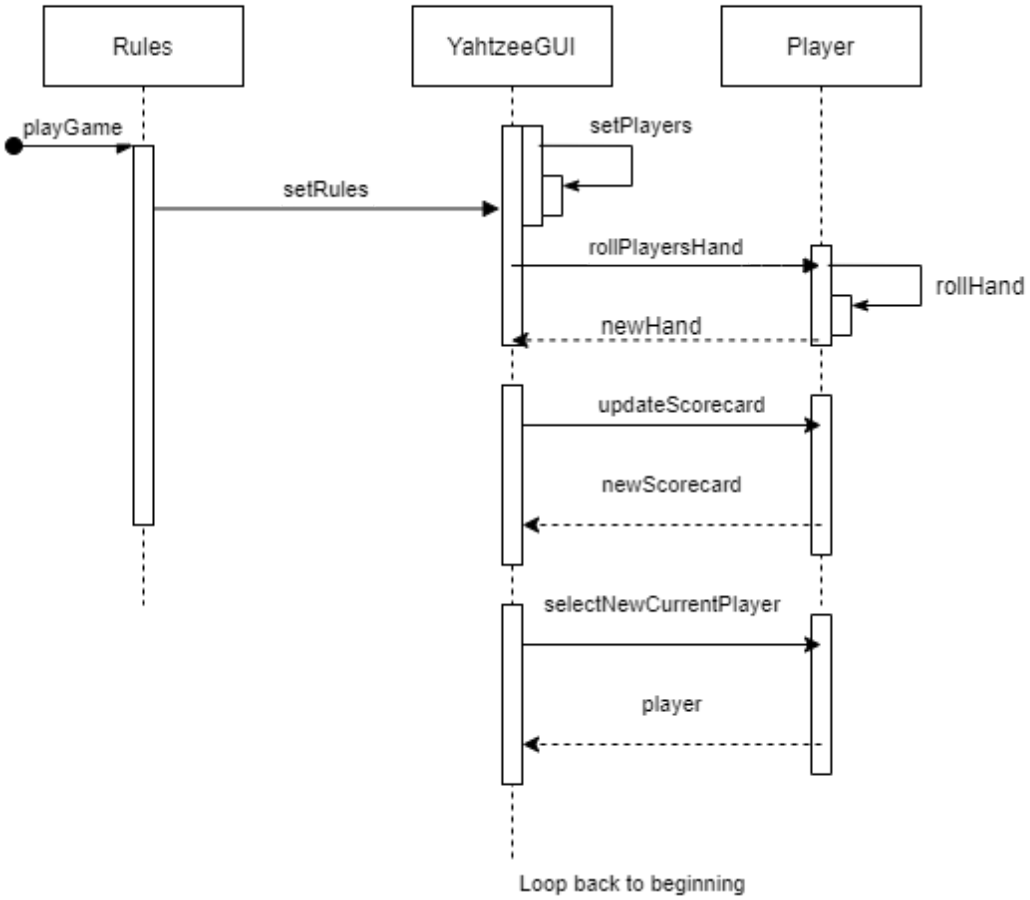
- We felt that by adding this feature, it greatly improves the UI design of the program

Display of final game results

- Program announces the winner of the game and their respective score


Rules
- numPlayers: int
- f: JFrame
+ setGUIRules()
+ getNumPlayers(): int
YahtzeeGUI
+ playGame()
Player
- name: String
- scorecard: Scorecard
- hand: HandOfDice
Scorecard
- upperScores: ArrayList<Integer>
- lowerScores: ArrayList<Integer>
- upperLabels: ArrayList<String>
- lowerLabels: ArrayList<String>
- upperTotal: int
- lowerTotal: int
+ fillScoreCard()
+ printScoreCard()
+ endGame()
+ printLowerScoreOptions(hand)
+ printUpperScoreOptions(hand)
HandOfDice
- hand: Dice[]
- handSize: int
+ sortHand()
+ rollNewHand(String)
+ printHand()
+ getHand() hand
+ getDie(index) Die
Die
- sides: int
- dieNum: int
- locked: bool
+ rollDie()
+ toString() : String
+ getDie() : Die


YahtzeeGUI
HandOfDice
Dice
Rules
playGame
newHand
getHandSize
handSize
hand
newDie
rollDie
Die
rollNewHand
newHand

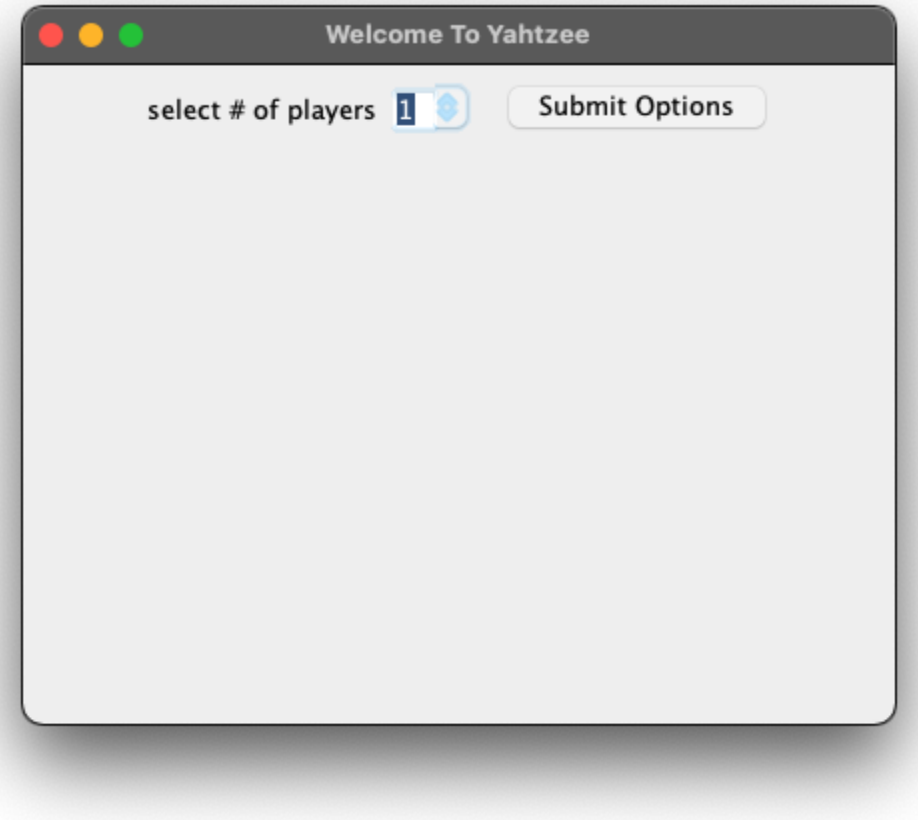

Rules
YahtzeeGUI
Player
playGame
setPlayers
setRules
rollPlayersHand
rollHand
newHand
updateScorecard
newScorecard
selectNewCurrentPlayer
player
Loop back to beginning

V. Test Plan

The test plan mainly consisted of functionality tests and user tests. Throughout the designing process, several run tests were performed when implementing new functions in order to test whether the functionality of the new additions improved the code or not. User tests were also a main source of testing. The developers allowed other people outside of the project to run the code in order to receive some feedback on either what needed to be changed/added or what the users liked in the program.

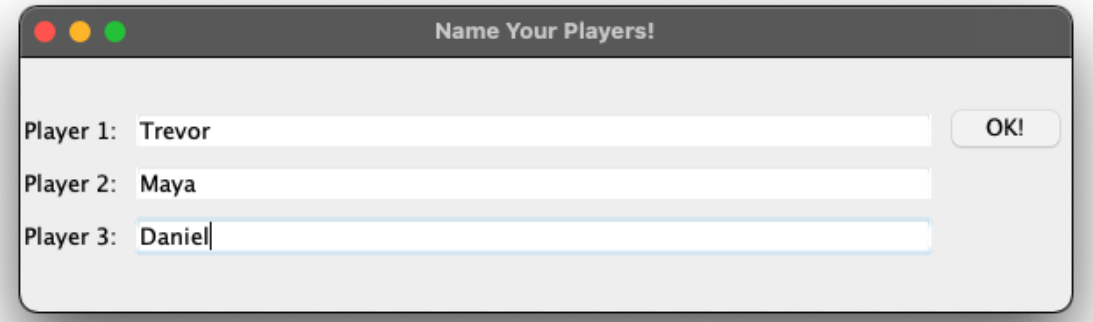
VI. Project Implementation Description

1. Welcome / Start screen



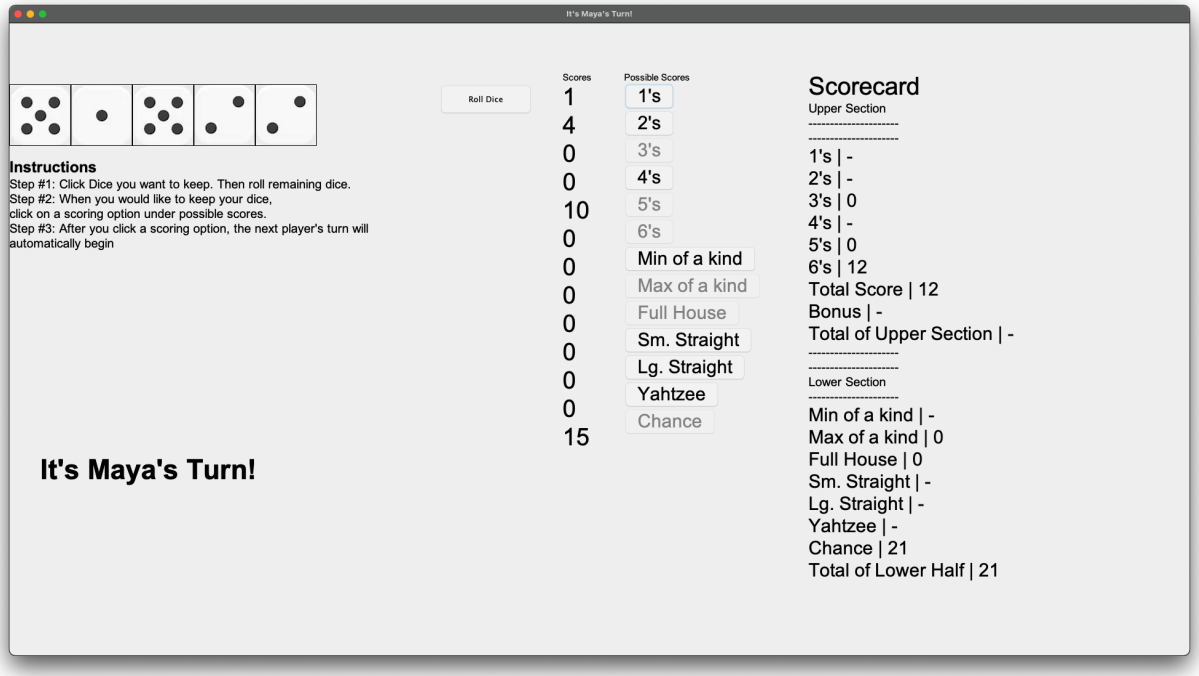
Users can choose the number of players for the game.

2. Player name choice



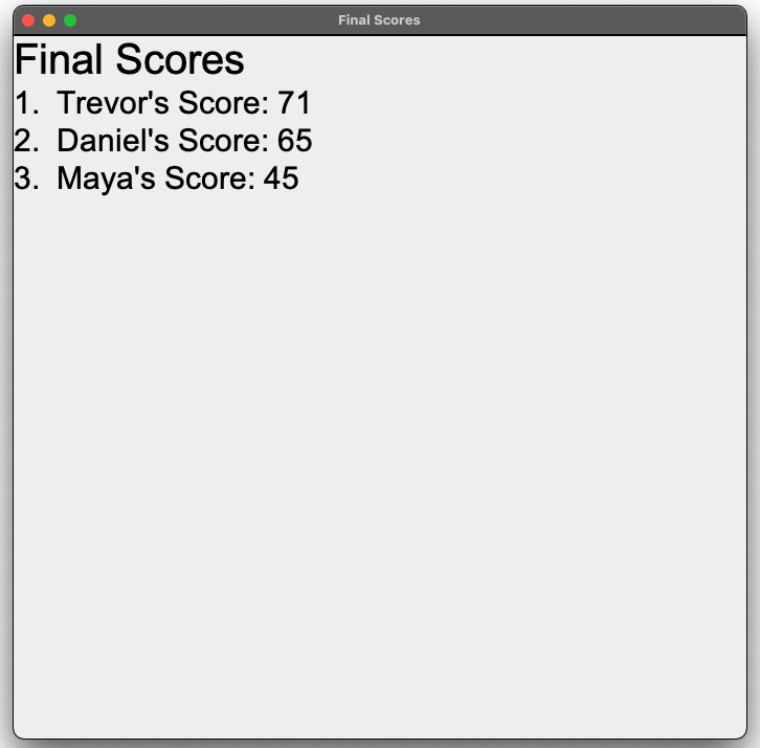
Players can give themselves a username. This username will display later in the game to show whose turn it is.

3. Main gameplay screen



This controls the main gameplay. Users can select the dice to keep, roll the dice, select specific scorecard item, view instructions, and view current game scorecard.

4. Final Score Page



Shows final scores from the games in sorted order. The username will be displayed next to the final score.

Github Repo:
https://github.com/Gonzaga-University/yahtzee-final-game-that-one-team

VII. Future Work

- Online multiplayer functionality
  - Play with people on different computers
- Give the game a theme and style
  - Change the background colors, design user icons, etc.
- Database to store previous game results, or save a current game.

VIII. References

"Yahtzee." Wikipedia. Wikimedia Foundation, April 12, 2021. https://en.wikipedia.org/wiki/Yahtzee#:~:text=Yahtzee%20is%20a%20dice%20game,Edwin%20S.%20Lowe%20in%201956.

"The History of Yahtzee." UltraBoardGames. Accessed May 6, 2021. https://www.ultraboardgames.com/yahtzee/history.php.