**Software Implementation and Testing Document**

**For**

**Group**

**Casino and More**

Version 2.0

**Authors**:

Brandon Mohammed

Alex Tedeschi

David Risi

# Programming Languages (5 points)

*List the programming languages use in your project, where you use them (what components of your project) and your reason for choosing them (whatever that may be).*

We use Java solely for all components of the project. Specifically, the Java Swing and Awt libraries. We choose to use Java due to its many easy to use graphical components and easy implementable options such as ActionListener’s and Paintcomponents. As our game has multiple different screens, we implemented new games with JFrames so they open in a separate window instead of the same JPanel.

# Platforms, APIs, Databases, and other technologies used (5 points)

*List all the platforms, APIs, Databases, and any other technologies you use in your project and where you use them (in what components of your project).*

No APIs or databases were used in the project. We used VsCode and IntelliJ to code the software and test on downloaded jdks.

# Execution-based Functional Testing (10 points)

*Describe how/if you performed functional testing for your project (i.e., tested for the* ***functional requirements*** *listed in your RD).*

Tests have been performed on all aspects of the final product. All the features for the main menu, settings, how to play, Blackjack, and Roulette are functional. Most of Solitaire is functional, however there was not enough time to complete all of the functional features. Currently there are problems with which cards are on top in the main board and moving cards to the suit pils. We tested main menu, settings, and how to play by compiling on jdks and seeing the output and then using the buttons. We tested Roulette, Blackjack, and Solitaire by compiling the software and tested the gameplay features to make sure they work properly by waiting for cases that cause specific features. For example, repetition of playing Blackjack until the case when the dealer busts to check that feature.

# Execution-based Non-Functional Testing (10 points)

*Describe how/if you performed non-functional testing for your project (i.e., tested for the* ***non-functional requirements*** *listed in your RD).*

We have tested the animation to spin the roulette wheel when the “spin” button is pressed. We tested this by compiling the roulette program and clicking on the spin button. Money system in roulette has also been tested this allows a user to keep track of their money throughout the program. Changing card backs has been tested. In Blackjack red x’s are placed on players that have busted, this was tested by purposely causing players to bust.

# Non-Execution-based Testing (10 points)

*Describe how/if you performed non-execution-based testing (such as code reviews/inspections/walkthroughs).*

We have all reviewed and walked through all the code implemented. Code was made in a step by step process, making sure that each part worked properly before moving on.