Software Requirements Specification

for

Rabbit Checkers

Version 1.0 approved

Prepared by Theodore Zuniga, Danielle Fernandez, Amy Ripley, and Chris Yanez

Team Rabbit

September 21st, 2018

Table of Contents

Table of Contents ii

Revision History ii

1. Introduction 1

1.1 Purpose 1

1.2 Document Conventions 1

1.3 Intended Audience and Reading Suggestions 1

1.4 Product Scope 1

1.5 References 1

2. Overall Description 2

2.1 Product Perspective 2

2.2 Product Functions 2

2.3 User Classes and Characteristics 2

2.4 Operating Environment 2

2.5 Design and Implementation Constraints 2

2.6 User Documentation 2

2.7 Assumptions and Dependencies 3

3. External Interface Requirements 3

3.1 User Interfaces 3

3.2 Hardware Interfaces 3

3.3 Software Interfaces 3

3.4 Communications Interfaces 3

4. System Features 4

4.1 System Feature 1 4

4.2 System Feature 2 (and so on) 4

5. Other Nonfunctional Requirements 4

5.1 Performance Requirements 4

5.2 Safety Requirements 5

5.3 Security Requirements 5

5.4 Software Quality Attributes 5

5.5 Business Rules 5

6. Other Requirements 5

Appendix A: Glossary 5

Appendix B: Analysis Models 5

Appendix C: To Be Determined List 6

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Date** | **Reason For Changes** | **Version** |
|  |  |  |  |
|  |  |  |  |

# Introduction

## Purpose

The purpose of this document is to provide an overview of the comprehensive requirements for the Rabbit Checkers application. This document will also describe the purpose and features of the application. Rabbit Checkers shall provide the user with a stable, solid checkers game. This SRS shall cover the full scope of Rabbit Checkers. The version of this product is 1.0.

## Document Conventions

WILL BE DEFINED AS NEEDED

**| Term | Definition |**

| RC | Rabbit Checkers |

## Intended Audience and Reading Suggestions

DON’T REALLY KNOW WHAT TO PUT EXACTLY

<Describe the different types of reader that the document is intended for, such as developers, project managers, marketing staff, users, testers, and documentation writers. Describe what the rest of this SRS contains and how it is organized. Suggest a sequence for reading the document, beginning with the overview sections and proceeding through the sections that are most pertinent to each reader type.>

## Product Scope

Rabbit Checkers will be developed entirely in Java and can be run on any system capable of running Java code. The entire project will be available on GitHub.

## References

"Checkers." Wikipedia, The Free Encyclopedia. 25 Aug 2018, 15:17 UTC. 20 Sep 2018, 23:30 <https://simple.wikipedia.org/w/index.php?title=Checkers&oldid=6232760>.

# Overall Description

## Product Perspective

This product will be the first of its kind from Team Rabbit. Rabbit Checkers will be a standalone desktop app, and will provide a game of checkers to the user(s). It will be developed using Java.

## Product Functions

**2.2.1)** Rabbit Checkers shall provide a full screen, color UI, with a simulated Checkers board and pieces.

**2.2.2)** Rabbit Checkers shall provide a “Player vs. AI” game mode.

**2.2.3)** Rabbit Checkers shall provide a “Player vs. Player” game mode.

**2.2.4)** Rabbit Checkers shall provide a “Timed Game” option, for both Player vs. AI and Player vs. Player game modes.

**2.2.5)** Rabbit Checkers shall provide a “Leader Board” where players will be able to see the number of games they have won compared to other players.

**2.2.6)** Rabbit Checkers shall provide “Color Themes” for players to choose between that will affect the UI color scheme and in-game colors.

2.2.7) Rabbit Checkers shall provide a “Get Hint” option that will provide the user with a hint on possible moves in-game.

**Note:** The above functions will be available for selection in either the main menu screen of the application, or as in-game buttons.

## User Classes and Characteristics

The main users of Rabbit Checkers will be individuals with access to a computer capable of running the JRE. There are no specific limitations or constraints on any classes of users.

## Operating Environment

Rabbit Checkers shall perform on any desktop or laptop platform that is capable of running java code.

## Design and Implementation Constraints (I DON’T THINK WE NEED THIS)

## User Documentation(I DON’T THINK WE NEED THIS)

## Assumptions and Dependencies(I DON’T THINK WE NEED THIS)

# External Interface Requirements

## User Interfaces

<Describe the logical characteristics of each interface between the software product and the users. This may include sample screen images, any GUI standards or product family style guides that are to be followed, screen layout constraints, standard buttons and functions (e.g., help) that will appear on every screen, keyboard shortcuts, error message display standards, and so on. Define the software components for which a user interface is needed. Details of the user interface design should be documented in a separate user interface specification.>

## Hardware Interfaces

“Rabbit Checkers” is a standalone desktop application. RC shall require a mouse and keyboard for input (and to interact with the user interface), along with a monitor for the display.

\*maybe a computer that can support the system?

\*Do we need to describe specific keys for the system

## Software Interfaces

Rabbit Checkers is a standalone application that will use Java.

## Communications Interfaces (NOT SURE IF WE NEED THIS)

\* The application will have a player vs AI and player vs player mode, but will all be on the same system.

# System Features

<This template illustrates organizing the functional requirements for the product by system features, the major services provided by the product. You may prefer to organize this section by use case, mode of operation, user class, object class, functional hierarchy, or combinations of these, whatever makes the most logical sense for your product.>

## User Interface

4.1.1 Description and Priority

This high priority feature shall welcome the user to a game of chess using a screen. This screen will allow the user to start a new game, explore other options, or exit the game.

4.1.2 Stimulus/Response Sequences

The user interface will open on start of the program and exit once the user presses the exit button. The options button will open a new window with the extra features. If anything else is clicked, the program will do nothing.

## Win/ Loss Determination

4.2.1 Description and Priority

This high priority feature shall end the game when a player wins and let the user know which player won.

4.2.2 Stimulus/Response Sequences

After all the pieces of one player have been captured, the program will disallow the opposing player from making a move, and will end the game with a window letting the player know if they won or lost.

## Position and Movement Enforcement

4.3.1 Description and Priority

This high priority feature shall put restrictions on where a player can move so they are only moving one piece one space diagonal, or two adjacent diagonal spaces if capturing an opposing piece.

4.3.2 Stimulus/Response Sequences

The game will allow the user to click one of their own pieces and click an appropriate space to move. If the space is illegal, nothing will happen. If the space is legal, the piece will move and the user’s turn will end.

## Transition to Kings

4.4.1 Description and Priority

This high functioning feature shall allow pieces to be “kinged” allowing that piece to move forward and backwards.

4.4.2 Stimulus/Response Sequences

Once a user moves one of its pieces to the farthest row from the user, the piece changes to an identifying “king” pieces, and the user’s turn end.

## AI Implementation

4.5.1 Description and Priority

This high priority feature shall allow the user to play the game against a simulated AI.

4.5.2 Stimulus/Response Sequences

After the user finishes a turn, the AI will move one of its pieces to an allowed space. It will the end its turn and allow the player to move one of their pieces.

## Player versus Player

4.6.1 Description and Priority

This medium priority feature shall allow the user to play the game with another controlling user.

4.6.2 Stimulus/Response Sequences

After the user finishes a turn, the game will allow a second user to move one of its pieces to an allowed space. The second user’s turn will end and allow the first user to move one of their pieces.

## Timer Mode

4.7.1 Description and Priority

This medium priority feature shall end the user’s turn after an allotted time.

4.7.2 Stimulus/Response Sequences

The game will keep a set amount of time (determined in the options menu) which will start at the beginning of a player’s turn. If the user takes a turn in the given time, the timer will stop and start over for the next turn. If the user doesn’t take a turn in the allotted time, the timer will end, and the player’s turn will end without a making a move.

## Color Theme

4.8.1 Description and Priority

This medium priority feature shall allow the user to select a different color theme for the board and pieces which will only change appearance.

4.8.2 Stimulus/Response Sequences

If the user selects the options menu, and the themes menu, the user will be allowed to select one of the given themes. The board and pieces will change colors, and possibly shapes.

## Leaderboard

4.9.1 Description and Priority

This medium priority feature shall keep a record of the players with the most amount of wins.

4.9.2 Stimulus/Response Sequences

<<<<<<<????????????>>>>>>>>

# Other Nonfunctional Requirements

## Performance Requirements

<If there are performance requirements for the product under various circumstances, state them here and explain their rationale, to help the developers understand the intent and make suitable design choices. Specify the timing relationships for real time systems. Make such requirements as specific as possible. You may need to state performance requirements for individual functional requirements or features.>

## Safety Requirements

<Specify those requirements that are concerned with possible loss, damage, or harm that could result from the use of the product. Define any safeguards or actions that must be taken, as well as actions that must be prevented. Refer to any external policies or regulations that state safety issues that affect the product’s design or use. Define any safety certifications that must be satisfied.>

## Security Requirements

<Specify any requirements regarding security or privacy issues surrounding use of the product or protection of the data used or created by the product. Define any user identity authentication requirements. Refer to any external policies or regulations containing security issues that affect the product. Define any security or privacy certifications that must be satisfied.>

## Software Quality Attributes

<Specify any additional quality characteristics for the product that will be important to either the customers or the developers. Some to consider are: adaptability, availability, correctness, flexibility, interoperability, maintainability, portability, reliability, reusability, robustness, testability, and usability. Write these to be specific, quantitative, and verifiable when possible. At the least, clarify the relative preferences for various attributes, such as ease of use over ease of learning.>

## Business Rules

<List any operating principles about the product, such as which individuals or roles can perform which functions under specific circumstances. These are not functional requirements in themselves, but they may imply certain functional requirements to enforce the rules.>

# Other Requirements

<Define any other requirements not covered elsewhere in the SRS. This might include database requirements, internationalization requirements, legal requirements, reuse objectives for the project, and so on. Add any new sections that are pertinent to the project.>

Appendix A: Glossary

<Define all the terms necessary to properly interpret the SRS, including acronyms and abbreviations. You may wish to build a separate glossary that spans multiple projects or the entire organization, and just include terms specific to a single project in each SRS.>

Appendix B: Analysis Models

<Optionally, include any pertinent analysis models, such as data flow diagrams, class diagrams, state-transition diagrams, or entity-relationship diagrams.>

Appendix C: To Be Determined List

<Collect a numbered list of the TBD (to be determined) references that remain in the SRS so they can be tracked to closure.>