**Sudoku Puzzle Solver Application**

**Test Design Document**

*Carl Koeppel, Mathew Grolemund, Garrett Luskey, Jarrett Schnaare*

1. **Introduction**

Sudoku puzzles are a type of puzzle in which players insert the numbers one to nine into a grid consisting of nine squares subdivided into a further nine smaller squares in such a way that every number appears once in each horizontal line, vertical line, and square. The players of the puzzle are usually given some sort of starting state of the puzzle, and they must work to fill in the remaining squares of the puzzle, while still following the previous rules. These puzzles are commonly found in newspapers across the world, and are usually fairly simple to solve. However, what happens when they cannot solve these puzzles? This was the central idea behind the Sudoku Puzzle Solver.

The Sudoku Puzzle Solver is a software application built in Java. It was designed so that a user, who is having trouble solving a Sudoku puzzle, may insert their current puzzle into the program, and the Sudoku Puzzle Solver will help them solve it. The application is set up with a nine by nine grid, where the users may input their current state of the puzzle with a mouse and keyboard, or they may choose to insert the information from a file. Either way, once the information is inserted into the grid, the user may choose to have the application solve it entirely, outputting the finished puzzle, or to solve a single square of the puzzle, by clicking on the appropriate buttons. If the current state of the puzzle is unsolvable, the Sudoku Puzzle Solver will tell them so, allowing the user to correct their mistakes. In the end, the Sudoku Puzzle Solver will help the user in their quest to solve their Sudoku puzzle.

This document contains the various tests that would have been implemented to test the nonfunctional and functional requirements of the application.

1. **Requirement**

* 1. **Functional Requirements**

IN-FN-01 The system shall allow the user to enter the original unsolved puzzle through a GUI.

IN-FN-02 The system shall be able to receive input from mouse and keyboard.

IN-FN-03 The system shall be able to read in an unsolved puzzle from a file.

IN-FN-04 The system shall allow the user to select a solve action through a clickable button once the unsolved puzzle is entered.

OUT-FN-01 The system shall utilize a GUI to accept user input and display the completed grid after the solve process is run.

OUT-FN-02 The system shall be able to write the initial, unsolved sudoku puzzle to a file.

OUT-FN-03 The system shall display a blank 9x9 grid to accept input into the system.

OUT-FN-04 The user shall have the option to display the board as it's being solved.

OUT-FN-05 The system shall be able to write the final, solved sudoku puzzle, to a file.

OUT-FN-06 The system shall display a completed 9x9 grid after the solve button is pressed and the solve process is completed.

OUT-FN-07 The system shall output music during runtime.

PT-FN-01 The system shall check each row in accordance to the game rules during the solve process.

PT-FN-02 The system shall check each column in accordance to the game rules during the solve process.

PT-FN-03 The system shall check each 3x3 square in accordance to the game rules during the solve process.

PT-FN-04 The system shall store counts of each number in each row and each column.

PT-FN-06 The system shall use the count of numbers to determine probability.

PT-FN-07 The system shall remove numbers from the list of possible correct integers in each box after checking each row, column, and 3x3 box for the same number.

PT-FN-11 The system shall be able to solve sudoku puzzle for user.

PT-FN-12 The system shall be able to run the solve process step-by-step for user.

* 1. **Non-Functional Requirements**

OUT-NF-08 The system shall output the solved sudoku puzzle, or an error message, within five seconds of clicking the solve button.

PT-NF-08 The system shall run using Java 1.8 and higher.

PT-NF-09 The system shall work on Windows Vista, 7, 8, 10.

PT-NF-10 The system shall be unavailable no longer than 5 minutes everyday.

PT-NF-13 The system shall be able to run the program remotely through a connected server.

PT-NF-14 The system shall be compatible with android and iOS systems.

GR-NF-01 The system shall only allow unique integers in every row between 1 and 9.

GR-NF-02 The system shall only allow unique integers in every column between 1 and 9.

GR-NF-03 The system shall require each column and row to contain numbers 1 through 9 without repetition or omission.

GR-NF-04 The system shall only permit unique integers in every one of the 9 3x3 boxes.

GR-NF-05 The system shall only permit one unique solution to each puzzle.

1. **Test Design**

|  |  |
| --- | --- |
| Test Case ID | T01 |
| Purpose | To test IN-FN-01 / IN-FN-02 / OUT-FN-03, the ability of the program to accept input from keyboard and mouse. |
| Pre-conditions | The application is running. |
| Inputs | Integers representing the numbers of an unsolved sudoku puzzle into the GUI's table using mouse and keyboard input. |
| Expected Outputs | Unsolved sudoku puzzle displayed in the GUI. |
| Post-conditions | Application waiting for user action, with appropriate integers in the GUI. |
| Design Technique | Reviewing Requirement |

|  |  |
| --- | --- |
| Test Case ID | T02 |
| Purpose | To test IN-FN-03, if a an unsolved sudoku puzzle file can be read in via a .txt file. |
| Pre-conditions | The application is running, and the load from file button is pressed. |
| Inputs | Unsolved sudoku puzzle file .txt file is chosen after the choose file button pressed. |
| Expected Outputs | The unsolved sudoku puzzle is displayed in the GUI. |
| Post-conditions | Application waiting for user action, with appropriate integers displayed in the GUI. |
| Design Technique | Reviewing Requirement |

|  |  |
| --- | --- |
| Test Case ID | T03 |
| Purpose | To test OUT-FN-02, the ability to write the original unsolved puzzle to a file. |
| Pre-conditions | The application is running, and an unsolved puzzle is input into the GUI board. |
| Inputs | Save to file box is ticked. |
| Expected Outputs | Unsolved sudoku puzzle written to a .txt file. |
| Post-conditions | New .txt file exists in the file directory |
| Design Technique | Reviewing Requirement |

|  |  |
| --- | --- |
| Test Case ID | T04 |
| Purpose | To test OUT-FN-04 / PT-FN-12, the ability to solve the unsolved puzzle step by step by pressing the solve step button, and display the puzzle being solved step by step. |
| Pre-conditions | The application is running, and an unsolved or partially solved puzzle is loaded into the GUI board. |
| Inputs | Unsolved sudoku puzzle, solve step button press. |
| Expected Outputs | Next step of the sudoku puzzle is solved, so one cell is now populated with the correct integer. |
| Post-conditions | Application is ready for user action. |
| Design Technique | Reviewing Requirement |

|  |  |
| --- | --- |
| Test Case ID | T05 |
| Purpose | To test IN-FN-04 / OUT-FN-01 / OUT-FN-06 / PT-FN-11, the ability to solve the unsolved puzzle by pressing the solve puzzle button, and the ability to display a completed puzzle in the GUI. |
| Pre-conditions | The application is running, and an unsolved puzzle is input into the GUI board. |
| Inputs | Unsolved sudoku puzzle, solve button press |
| Expected Outputs | Solved sudoku puzzle in the board |
| Post-conditions | Application is ready for user action |
| Design Technique | Reviewing Requirement |

|  |  |
| --- | --- |
| Test Case ID | T06 |
| Purpose | To test OUT-FN-05, the ability to write the solved puzzle to a .txt file. |
| Pre-conditions | The application is running, and a solved sudoku puzzle is in the GUI board. |
| Inputs | Write to file button is checked |
| Expected Outputs | .txt file of solved puzzle is produced |
| Post-conditions | .txt file exists in directory |
| Design Technique | Reviewing requirement |

|  |  |
| --- | --- |
| Test Case ID | T07 |
| Purpose | To test OUT-FN-07, that music is played in the background of the software. |
| Pre-conditions | The application is running. |
| Inputs | None |
| Expected Outputs | Music |
| Post-conditions | Music is being played |
| Design Technique | Reviewing requirement |

|  |  |
| --- | --- |
| Test Case ID | T08 |
| Purpose | To test PT-FN-01, that only one integer from 1 to 9 appears in each row of a completed board. |
| Pre-conditions | The application is running, and a board with 2 numbers of the same value in the same row of the board has been input. |
| Inputs | Two identical numbers in the same row of the board, solve button pressed. |
| Expected Outputs | Error message prompting the user to check over their board. |
| Post-conditions | Prompt to reenter the numbers into the cells. |
| Design Technique | Boundary Analysis |

|  |  |
| --- | --- |
| Test Case ID | T09 |
| Purpose | To test PT-FN-01, that only one integer from 1 to 9 appears in each row of a completed board. |
| Pre-conditions | The application is running, and a board without 2 numbers of the same value in the same row of the board is input. |
| Inputs | Partial sudoku puzzle, solve button pressed. |
| Expected Outputs | No error message, solved puzzle displayed in GUI |
| Post-conditions | Application is ready to accept user input. |
| Design Technique | Boundary Analysis |

|  |  |
| --- | --- |
| Test Case ID | T10 |
| Purpose | To test PT-FN-02, that only one integer from 1 to 9 appears in each column of a completed board. |
| Pre-conditions | The application is running, and a board with 2 numbers of the same value in the same column of the board is input. |
| Inputs | 2 numbers with the same value in the same column, solve button pressed. |
| Expected Outputs | Error message prompting the user to check over their board |
| Post-conditions | Prompt for reentry of integers |
| Design Technique | Boundary Analysis |

|  |  |
| --- | --- |
| Test Case ID | T11 |
| Purpose | To test PT-FN-02, that only one integer from 1 to 9 appears in each column of a completed board. |
| Pre-conditions | The application is running, and a board without 2 numbers of the same value in the same column of the board are input. |
| Inputs | Partial sudoku puzzle, solve button pressed. |
| Expected Outputs | No error message, solved puzzle displayed |
| Post-conditions | Application is waiting for user input |
| Design Technique | Boundary Analysis |

|  |  |
| --- | --- |
| Test Case ID | T12 |
| Purpose | To test PT-FN-03, that only one integer from 1 to 9 appears in each 3x3 square |
| Pre-conditions | The application is running, and a board with 2 integers of the same value in the same 3x3 box of the board are input. |
| Inputs | 2 integers of the same value in the same 3x3 box of the board, solve button pressed. |
| Expected Outputs | Error message prompting the user to check over their board. |
| Post-conditions | Application is waiting for user input |
| Design Technique | Boundary Analysis |

|  |  |
| --- | --- |
| Test Case ID | T13 |
| Purpose | To test PT-FN-04, that the classes will keep counts of each number input. |
| Pre-conditions | Application is running in debug mode. |
| Inputs | Integers between 1-9 input into the GUI |
| Expected Outputs | Nothing to the user, but the expected count in debugging |
| Post-conditions | Application is waiting for user action. |
| Design Technique | Boundary Analysis, Reviewing Requirements |

|  |  |
| --- | --- |
| Test Case ID | T14 |
| Purpose | To test OUT-NF-08, the system's ability to solve a solvable puzzle and output it to the GUI within 5 seconds of a solve command. |
| Pre-conditions | The application is running, and a solvable sudoku puzzle loaded into the board. |
| Inputs | Solvable sudoku puzzle into the GUI |
| Expected Outputs | No error message, solved puzzle within 5 seconds |
| Post-conditions | Application is waiting for user action |
| Design Technique | Reviewing requirements |

|  |  |
| --- | --- |
| Test Case ID | T15 |
| Purpose | To test PT-FN-03, that only one integer from 1 to 9 appears in each 3x3 square |
| Pre-conditions | The application is running, and a board without 2 integers of the same value in the same 3x3 box of the board. |
| Inputs | Partial sudoku puzzle, solve button pressed. |
| Expected Outputs | No error message, solved sudoku puzzle. |
| Post-conditions | Application is waiting for user action. |
| Design Technique | Boundary Analysis |

|  |  |
| --- | --- |
| Test Case ID | T16 |
| Purpose | To test OUT-NF-08, the system's ability to output an error message for an unsolvable puzzle to the error window in the GUI within 5 seconds of a solve command. |
| Pre-conditions | The application is running, and an unsolvable sudoku puzzle is loaded into the board. |
| Inputs | Unsolvable sudoku puzzle |
| Expected Outputs | Error message printed to the error window. |
| Post-conditions | Error message appears within 5 seconds of the solve command being pressed. |
| Design Technique | Reviewing Requirements |

|  |  |
| --- | --- |
| Test Case ID | T17 |
| Purpose | To test PT-FN-07, the system's ability to remove a number from the list of correct numbers of a cell when that number already exists in the row/column/3x3box |
| Pre-conditions | The application is running, and an unsolved sudoku puzzle loaded into the board. Run in debugging mode. |
| Inputs | Unsolved sudoku puzzle, solve step button is pressed |
| Expected Outputs | The puzzle is solved one step, and, in debugging, the numbers within the list of "possible numbers" update correctly in debug window. |
| Post-conditions | The application is waiting for user action |
| Design Technique | Reviewing Requirements |

|  |  |
| --- | --- |
| Test Case ID | T18 |
| Purpose | To test GR-NF-04 / GR-NF-01 / GR-NF-02, that the program will only allow unique digits from 1 to 9. |
| Pre-conditions | Application is running |
| Inputs | A number larger than 9 is input into a cell, solve button pressed |
| Expected Outputs | An error message for an invalid input is displayed. |
| Post-conditions | Prompt for reentry of inputs by users. |
| Design Technique | Boundary Analysis |

|  |  |
| --- | --- |
| Test Case ID | T19 |
| Purpose | To test PT-NF-08 / PT-NF-09, to see if software runs on windows vista running java 1.8 |
| Pre-conditions | Must have windows vista and java 1.8 installed. |
| Inputs | Unsolved sudoku puzzle, solve button is pressed |
| Expected Outputs | Solved Sudoku Puzzle and ability to save and upload sudoku puzzle |
| Post-conditions | Program runs all functional buttons |
| Design Technique | Reviewing Requirements |

|  |  |
| --- | --- |
| Test Case ID | T20 |
| Purpose | To test PT-NF-10, the theoretical system's ability to be online for 24 hours with only 5 minutes of downtime. |
| Pre-conditions | The program is loaded and running on a computer. |
| Inputs | The system is pinged every minute |
| Expected Outputs | There aren't more than 5 minutes of downtime per 24 hours. |
| Post-conditions | The system continues to run. |
| Design Technique | Reviewing Requirements |

|  |  |
| --- | --- |
| Test Case ID | T21 |
| Purpose | To test PT-NF-14, the ability for the software to run on an Android operating system. |
| Pre-conditions | Have a running Android device with the application and its requirements installed. |
| Inputs | Application is launched |
| Expected Outputs | Application runs successfully, with the screen popping up successfully |
| Post-conditions | The software continues to run, and is ready for user action. |
| Design Technique | Reviewing Requirements |

|  |  |
| --- | --- |
| Test Case ID | T22 |
| Purpose | To test PT-NF-13, the system's ability to run the program remotely through a connected server. |
| Pre-conditions | The server is running. |
| Inputs | The software is launched from the server remotely |
| Expected Outputs | The application runs on the host computer. |
| Post-conditions | A connection to the application server, and the application is ready and waiting for user action. |
| Design Technique | Reviewing Requirements |

|  |  |
| --- | --- |
| Test Case ID | T23 |
| Purpose | To test GR-NF-04 / GR-NF-01 / GR-NF-02, that the system shall only allow unique integers between 1 and 9. |
| Pre-conditions | The application is running on the computer. |
| Inputs | A number smaller than 1 is input into a cell |
| Expected Outputs | An error message to the message window is printed for having invalid input. |
| Post-conditions | Prompt for reentry of integer |
| Design Technique | Boundary Analysis |

|  |  |
| --- | --- |
| Test Case ID | T24 |
| Purpose | To test GR-NF-04 / GR-NF-01 / GR-NF-02, that the system shall only allow unique integers between 1 and 9. |
| Pre-conditions | The application is running on the computer. |
| Inputs | A character (char) is input into a cell. |
| Expected Outputs | An error message to the message window is printed for having invalid input. |
| Post-conditions | Prompt for reentry of integer. |
| Design Technique | Boundary Analysis |

|  |  |
| --- | --- |
| Test Case ID | T25 |
| Purpose | To test GR-NF-05, To check that the system is outputting only one solution. |
| Pre-conditions | Application is running |
| Inputs | Unsolved Sudoku Puzzle |
| Expected Outputs | One solved Sudoku Puzzle. |
| Post-conditions | The application is waiting for user action |
| Design Technique | Requirement Review |

|  |  |
| --- | --- |
| Test Case ID | T26 |
| Purpose | To test GR-NF-03, that the system shall require each column and row to contain numbers 1 through 9 without repetition or omission after the solve process. |
| Pre-conditions | Application is running |
| Inputs | Unsolved Sudoku Puzzle, solve button pressed |
| Expected Outputs | A solved Sudoku puzzle is displayed in the GUI. |
| Post-conditions | Every row and every column contains integers 1 through 9 without omission or repetition. |
| Design Technique | Reviewing Requirements |

|  |  |
| --- | --- |
| Test Case ID | T27 |
| Purpose | To test PT-NF-08 / PT-NF-09, to see if software runs on windows 7 running java 1.8 |
| Pre-conditions | Must have windows 7 and java 1.8 installed. |
| Inputs | Unsolved sudoku puzzle, solve button is pressed |
| Expected Outputs | Solved Sudoku Puzzle and ability to save and upload sudoku puzzle |
| Post-conditions | Program runs all functional buttons |
| Design Technique | Reviewing Requirements |

|  |  |
| --- | --- |
| Test Case ID | T28 |
| Purpose | To test PT-NF-08 / PT-NF-09 to see if software runs on windows 8 running java 1.8 |
| Pre-conditions | Must have windows 8 and java 1.8 installed. |
| Inputs | Unsolved sudoku puzzle, solve button is pressed |
| Expected Outputs | Solved Sudoku Puzzle and ability to save and upload sudoku puzzle |
| Post-conditions | Program runs all functional buttons |
| Design Technique | Reviewing Requirements |

|  |  |
| --- | --- |
| Test Case ID | T29 |
| Purpose | To test PT-NF-08 / PT-NF-09, to see if software runs on windows 10 running java 1.8 |
| Pre-conditions | Must have windows 10 and java 1.8 installed. |
| Inputs | Unsolved sudoku puzzle, solve button is pressed |
| Expected Outputs | Solved Sudoku Puzzle and ability to save and upload sudoku puzzle |
| Post-conditions | Program runs all functional buttons |
| Design Technique | Reviewing Requirements |

|  |  |
| --- | --- |
| Test Case ID | T30 |
| Purpose | To test PT-NF-06, the system's ability to use the count of integers in other rows and columns to determine the probability that a number exists in a given cell. |
| Pre-conditions | Application is running |
| Inputs | Unsolved sudoku puzzle |
| Expected Outputs | Probability value for a given integer 1 trough 9 at a given cell. |
| Post-conditions | N/A |
| Design Technique | Reviewing Requirements |

|  |  |
| --- | --- |
| Test Case ID | T31 |
| Purpose | To test PT-NF-14, the ability for the software to run on an IOS operating system. |
| Pre-conditions | Have a running IOS device with the application and its requirements installed. |
| Inputs | Application is launched. |
| Expected Outputs | Application runs. |
| Post-conditions | The software continues to run. |
| Design Technique | Reviewing Requirements |

1. **Traceability**

|  |  |
| --- | --- |
| Test Case Number | List of the Requirements tested |
| T01 | IN-FN-01 / IN-FN-02 / OUT-FN-03 |
| T02 | IN-FN-03 |
| T03 | OUT-FN-02 |
| T04 | OUT-FN-04 / PT-FN-12 |
| T05 | IN-FN-04 / OUT-FN-01 / OUT-FN-06 / PT-FN-11 |
| T06 | OUT-FN-05 |
| T07 | OUT-FN-07 |
| T08 | PT-FN-01 |
| T09 | PT-FN-01 |
| T10 | PT-FN-02 |
| T11 | PT-FN-02 |
| T12 | PT-FN-03 |
| T13 | PT-FN-04 |
| T14 | OUT-NF-08 |
| T15 | PT-FN-03 |
| T16 | OUT-NF-08 |
| T17 | PT-FN-07 |
| T18 | GR-NF-04 / GR-NF-01 / GR-NF-02 |
| T19 | PT-NF-08 / PT-NF-09 |
| T20 | PT-NF-10 |
| T21 | PT-NF-14 |
| T22 | PT-NF-13 |
| T23 | GR-NF-04 / GR-NF-01 / GR-NF-02 |
| T24 | GR-NF-04 / GR-NF-01 / GR-NF-02 |
| T25 | GR-NF-05 |
| T26 | GR-NF-03 |
| T27 | PT-NF-08 / PT-NF-09 |
| T28 | PT-NF-08 / PT-NF-09 |
| T29 | PT-NF-08 / PT-NF-09 |
| T30 | PT-NF-06 |
| T31 | PT-NF-14 |

*Create a traceability table similar as the above example. Make sure each requirement is tested by some test case at least once.*

**Other instructions:**

* All documents shall have a footer that contains Team Name, Page Number, and Date
* All documents shall have a header that contains the name/title of the document.
* Professional appearance is very important.
* The stuff that is in italic should not be in the final version of the document.

1. **Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Action** | **Who Reviewed** | **Comments** |
| 4/26 | Created | Matt, Jarrett, Carl, Garrett | Edited the traceability table where inconsistencies occurred. |
| 4/27 | Reviewed | Carl, Mathew, Jarrett, Garrett | Read through the test designs looking for missing inputs and repetitious tests. |
| 5/4 | Revised | Carl, Mathew | Changed the introduction and made sure the document was ready to be handed i |
| 5/4 | Hand in | Carl |  |

1. **Glossary**

|  |  |
| --- | --- |
| Abbreviation | Description |
| GR-NF | Functional game rules requirement |
| GUI | Graphic User Interface |
| IN-FN | Functional input requirement |
| IN-NF | Non-functional input requirement |
| iOS | Apple iPhone Operating System |
| OUT-FN | Functional output requirement |
| OUT-NF | Non-functional output requirement |
| PT-FN | Functional process or task requirement |
| PT-NF | Non-functional process or task requirement |

1. **Reference**