|  |
| --- |
| Team JEM |
| Settlers of Catan |
| Programmers Manual |
|  |
| **Eric Maxwell; Justin Lynn; Matt Jackels** |
|  |

|  |
| --- |
|  |

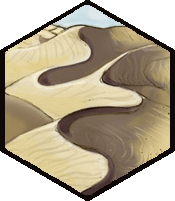


Table of Contents

[1. Revision History 2](#_Toc481758576)

[2. Purpose 2](#_Toc481758577)

[3. Definitions 3](#_Toc481758578)

[4. Software Requirements 3](#_Toc481758579)

[4.1. General Details & Description 3](#_Toc481758580)

[4.2. Software Requirements Matrix 4](#_Toc481758581)

[5. Design Documentation 11](#_Toc481758582)

[5.1 High Level Architecture 11](#_Toc481758583)

[5.2 Game Class 12](#_Toc481758584)

[5.3 GUI Class 13](#_Toc481758585)

[5.4 Board Class 16](#_Toc481758586)

[5.5 Tile & Associated Classes 17](#_Toc481758587)

[5.6 Location, LocationList, LocationButton Classes 18](#_Toc481758588)

[5.7 Player and PlayerManager Classes 19](#_Toc481758589)

[5.8 GamePiece Abstract Class with Subclasses 20](#_Toc481758590)

[5.9 DevelopmentCard Abstract Class with Subclasses 21](#_Toc481758591)

[5.10 Dice Class 22](#_Toc481758592)

[6 Source Code 23](#_Toc481758593)

[7 Testing Strategy & Documentation 23](#_Toc481758594)

[7.1 Test Plan 23](#_Toc481758595)

[7.1.1 Black Box Strategy 23](#_Toc481758596)

[7.1.2 White Box Strategy 23](#_Toc481758597)

[7.1.3 Test Cases 23](#_Toc481758598)

[7.2 Results 26](#_Toc481758599)

[8 Known Bugs 78](#_Toc481758600)

# Revision History

|  |  |  |
| --- | --- | --- |
| **Date** | **Change Description** | **Changed By** |
| 5/1/2017 | Final polishing. | M. Jackels |
| 4/30/2017 | Added Design and Testing sections. Polished all sections and final review. | M. Jackels |
| 4/5/2017 | Added source code section, testing section & known bug section to align document w/ deliverable requirements. | M. Jackels |
| 4/2/2017 | Due to the creation of the Project Plan document, the Configuration Management and Tools & Standards sections were removed. | M. Jackels |
| 3/5/2017 | Added RequirementsMatrix and ConfigurationMgmt references in ‘References’ section.  Completed ‘Purpose’ section.  Added ‘Revision History’ section.  Completed ‘Configuration Management’ section – sourced from ConfigurationMgmt\_JEM document. | M. Jackels |
| 2/13/2017 | Initial Release | M. Jackels |

# Purpose

The purpose of this document is to consolidate the software design specifications to a single-source document for reference and understanding of the source code for the *Settlers of Catan* software. This is intended to be written for, and used by, any future programmers intending to modify the code.

The scope of this document is to cover all design considerations for the most current *Settlers of Catan* software release built by Team JEM. It includes the requirements used to architect the program, the design, testing documentation, and currently known bugs.

The scope of the JEM project is to plan, design, build, and implement a software version of the game *Settlers of Catan*. This board game popularized by Mayfair offers gameplay for 2-4 players who take turns rolling dice, managing cards, and building settlements. The software will be written in Java and developed to work with the java runtime environment. It will feature a graphical user interface for each player to take their turn and perform all necessary actions. The software will initially be a stand-alone game running on a single PC. We will add a dedicated AI and network connectivity in later iterations as time permits.

# Definitions

**Term** – definition

* **Requirements** – the officially approved functionalities to be considered and programmed in the creation of the software.
* **Bugs** – issues or inconsistencies with the program. This can include both front-end (noticeable to an end user) and back-end (noticeable in the source code) issues.
* **Configuration** – The version of the software, as determined by the revision level of the source-code.
* **Software** – Used with, or interchangeably with, ‘Program’ or ‘Application’ in this document and refers to the executable source code.
* **Architecture** – The design considerations that are used to build a program. This includes diagrams of dependencies and interactions of classes/modules in the software.

# Software Requirements

# General Details & Description

Settlers of Catan is a board game consisting of:

* Nineteen (19) hexagonal land tiles,
* Six (6) sea tiles,
* Nine (9) harbor tiles,
* Resource cards consisting of Ore, Grain, Lumber, Wool, and Brick
* Development cards consisting of Soldiers, Victory Points, and Progress
* Pieces shaped as a Robber, Cities, Settlements, and Roads
* Two (2) dice

The setup of the game involves randomly placing all tiles into a large platform, where the land tiles are surrounded by the harbor and sea tiles to form a “settlement”. The land tiles are each labeled with a number and have resource designated by the background. The game is then played by switching turns between the players. During their turn, a player gets to roll the two dice and any cities that are placed at a point that touches the resource with matching number to the sum of the dice earns the player that resource card. Resource cards then get used to build more roads, cities, settlements, or development cards. The goal of the game is to earn victory points through development cards and settlement growth.

This software program is being created as a merge between existing technology, while maintaining the excitement of the original board game. There are a variety of advantages to having a software version of the game, and the biggest driving factor is mobility. The physical board game takes up a large space and is completely immobile due to the modular design of the hexagonal tiles. It must sit on a flat, stable surface. In contrast, the software version can be used in vehicles, on the go, and in smaller spaces. The intended market for this application consists of two complementary groups: the first is current Catan players who will find value in the mobility of this version; the second group are people who currently utilize mobile game stores and may venture to find this previously unknown game is enjoyable to them.

The Settlers of Catan software is intended to work on any computer that has Java Runtime Environment installed. The program is packaged in a .JAR file, and is to run locally. This means there are some limitations that currently exist, such as the inability to play online in a player network environment. Additionally, the capability of the hardware on the computer running the program can conceivably limit the effectiveness of the software.

This game was inspired by the existing board game Settlers of Catan, which can be found at: <http://www.catan.com/>. The instruction manual for the board version, which served a reference to the software user manual, can be found at <http://www.catan.com/service/game-rules>. The Settlers of Catan board game can be viewed and purchased there, along with many other retail stores. All rights to the game are owned by them, and this software application is not intended for sale or distribution unless approval is requested and granted by Catan Studio.

# Software Requirements Matrix

This section identifies all captured requirements for the Settlers of Catan application. This includes implemented requirements, as well as requirements identified for future revisions. The matrix titled “**Table 1 – Requirements**” below captures all requirements and associated information. The included fields are:

* ID: A unique identifier for each requirement for improved clarity and traceability.
* Priority: The level of importance to the application. P1 is highest priority, P2 is second highest, and P3 is lowest. The three digits succeeding the ‘R’ indicate the level of the requirement, where the leftmost digit is the broad categorization, the middle digit is a subrequirement of its corresponding ‘Level 1’ requirement, and the rightmost digit is a sub-subrequirement.
* Description: The explicit explanation of what is required.
* Comments: Notes that can help clarify the requirement
* Approved By & Date: The testing lead of the project will review and approve all satisfactory requirements.
* Color Code:
  + Orange – Top level explanation of the requirements. This is a Level 1 requirement
  + White – Detailed subrequirement and sub-subrequirements of the Level 1 requirements
  + Blue – Identifies P2 requirements

Upon completion of adding all requirements and final review, the requirements matrix was locked down so that coding and testing could be completed accurately. Additionally, all met requirements will be identified in the testing phase. Any requirements not met, or not implemented, will be called out and subsequently identified as in scope for immediate revision, or in scope for future iterations.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Table 1 - Requirements** | | | | | |
| ID | Priority  (1-3) | Requirement Description | Comments | Approved Date | Approved By |
| R1.0.0 |  | The game should set up the board correctly |  | 2/26/2017 | J. Lynn |
| R1.1.0 | P1 | There should be a minimum of two players |  | 2/26/2017 | J. Lynn |
| R1.1.1 | P1 | Three players should be allowed |  | 2/26/2017 | J. Lynn |
| R1.1.2 | P1 | There should be a maximum of four players |  | 2/26/2017 | J. Lynn |
| R1.2.0 | P1 | There should be 19 resource tiles |  | 2/26/2017 | J. Lynn |
| R1.2.1 | P1 | There should be 3 brick tiles |  | 2/26/2017 | J. Lynn |
| R1.2.2 | P1 | There should be 3 ore tiles |  | 2/26/2017 | J. Lynn |
| R1.2.3 | P1 | There should be 4 lumber tiles |  | 2/26/2017 | J. Lynn |
| R1.2.4 | P1 | There should be 4 wool tiles |  | 2/26/2017 | J. Lynn |
| R1.2.5 | P1 | There should be 4 grain tiles |  | 2/26/2017 | J. Lynn |
| R1.2.6 | P1 | There should be 1 desert tile |  | 2/26/2017 | J. Lynn |
| R1.2.7 | P1 | The resource tiles should be placed randomly on the board |  | 2/26/2017 | J. Lynn |
|  |  |  |  |  |  |
| R1.3.0 | P1 | There should be 18 number tokens |  | 2/26/2017 | J. Lynn |
| R1.3.1 | P1 | The numbers on the tile should be (2,3,3,4,4,5,5,6,6,8,8,9,9,10,10,11,11,12) |  | 2/26/2017 | J. Lynn |
| R1.3.2 | P1 | Each resource producing tile should have one visible number token |  | 2/26/2017 | J. Lynn |
| R1.3.3 | P1 | The desert tile should not have a number token |  | 2/26/2017 | J. Lynn |
| R1.3.4 | P1 | The number tokens 8, 8, 6 and 6 should not be in adjacent resource tiles. |  | 2/26/2017 | J. Lynn |
|  |  |  |  |  |  |
| R1.4.0 | P1 | There should be one robber, which starts on the desert tile |  | 2/26/2017 | J. Lynn |
|  |  |  |  |  |  |
| R1.5.0 | P1 | The game should scale window sizes to accommodate native screen resolutions |  | 2/26/2017 | J. Lynn |
|  |  |  |  |  |  |
| R1.6.0 | P2 | There should be 9 harbor pieces |  | 2/26/2017 | J. Lynn |
| R1.6.1 | P2 | There should be one harbor for each resource |  | 2/26/2017 | J. Lynn |
| R1.6.2 | P2 | There should be 4 generic harbors |  | 2/26/2017 | J. Lynn |
|  |  |  |  |  |  |
| R2.0.0 | P1 | There should be a Graphical User Interface |  | 2/26/2017 | J. Lynn |
| R2.1.0 | P1 | The correct score should display for each player. |  | 2/26/2017 | J. Lynn |
|  |  |  |  |  |  |
| R2.2.0 | P1 | The current player should be listed at the top of the GUI |  |  |  |
| R2.2.1 | P1 | The current player's score should be visible |  | 2/26/2017 | J. Lynn |
| R2.2.2 | P1 | The current player's resources should be visible |  | 2/26/2017 | J. Lynn |
|  |  |  |  |  |  |
| R2.3.0 | P1 | Building Costs should be referenced in the GUI |  | 2/26/2017 | J. Lynn |
|  |  |  |  |  |  |
| R2.4.0 | P1 | There should be a build settlement button | implemented in later requirements | 3/3/2017 | E. Maxwell |
| R2.4.1 | P1 | There should be a build city button | implemented in later requirements | 3/3/2017 | E. Maxwell |
| R2.4.2 | P1 | There should be a build road button | implemented in later requirements | 3/3/2017 | E. Maxwell |
| R2.4.3 | P1 | There should be a buy development card button | implemented in later requirements | 3/3/2017 | E. Maxwell |
| R2.4.4 | P1 | There should be a play development card button | implemented in later requirements | 3/3/2017 | E. Maxwell |
| R2.4.5 | P1 | There should be a finish turn button which changes the current player to the next player |  | 3/3/2017 | E. Maxwell |
| R2.4.6 | P1 | There should be a trade button | implemented in later requirements |  |  |
|  |  |  |  |  |  |
| R3.0.0 |  | Proper number of Settlements, Cities and Roads for each player and proper Settlement and Road Locations |  | 2/26/2017 | J. Lynn |
|  |  |  |  |  |  |
| R3.1.0 | P1 | There should be 5 settlements for every player |  | 2/26/2017 | J. Lynn |
| R3.1.1 | P1 | There should be 4 cities for every player |  | 2/26/2017 | J. Lynn |
| R3.1.2 | P1 | There should be 15 roads for every player |  | 2/26/2017 | J. Lynn |
| R3.1.3 | P1 | The players' quantity of unplaced cities, settlements, and roads should be visible to them during their turn |  | 2/26/2017 | J. Lynn |
|  |  |  |  |  |  |
| R3.2.0 | P1 | The build settlement, build road, and build city buttons should be implemented |  | 3/5/2017 | E. Maxwell |
| R3.2.1 | P1 | The placed cities, settlements, and roads should be visible on the tiles at all times |  | 2/26/2017 | J. Lynn |
| R3.2.2 | P1 | Allow placement of settlements and cities on tile corners |  | 2/26/2017 | J. Lynn |
| R3.2.3 | P1 | Allow placement of roads along tile edges |  | 2/26/2017 | J. Lynn |
|  |  |  |  |  |  |
| R4.0.0 |  | The game should include a menu and the beginning of the game should execute |  | 2/26/2017 | J. Lynn |
|  |  |  |  |  |  |
| R4.1.0 | P1 | Each player should set 2 settlements and 2 roads before play begins |  | 2/26/2017 | J. Lynn |
| R4.1.1 | P1 | Each player takes a turn setting one settlement and one road |  | 2/26/2017 | J. Lynn |
| R4.1.2 | P1 | Player 1 begins, then P2 until the last player. |  | 2/26/2017 | J. Lynn |
| R4.1.3 | P1 | Once a settlement is placed, the player places a road on a road location next to the settlement |  | 2/26/2017 | J. Lynn |
| R4.1.4 | P1 | For the second settlement, the players go in reverse order, beginning with the last player. |  | 2/26/2017 | J. Lynn |
| R4.1.5 | P1 | Each player should receive one resource for every resource tile the player's initial settlement touches. |  | 2/26/2017 | J. Lynn |
|  |  |  |  |  |  |
| R4.2.0 | P1 | There should be an instruction manual in the menu. |  | 2/26/2017 | J. Lynn |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
| R5.0.0 |  | The game should execute the dice roll phase |  | 2/26/2017 | J. Lynn |
| R5.1.0 | P1 | There should be two 6-sided dice which can be rolled and with the roll information available |  | 2/26/2017 | J. Lynn |
|  |  |  |  |  |  |
| R5.2.0 | P1 | Each players turn should begin with a dice roll, which is displayed |  | 2/26/2017 | J. Lynn |
| R5.2.1 | P1 | Every tile with a number token that matches the current roll should give every settlement touching it one resource and every city gets two resources. |  | 2/26/2017 | J. Lynn |
| R5.2.2 | P1 | If a 7 is rolled, any player with more than 7 cards must select half of their resources (rounded down) and throw them out. |  | 2/26/2017 | J. Lynn |
| R5.2.3 | P1 | If a 7 is rolled, the current player moves the robber to a new tile center location |  | 2/26/2017 | J. Lynn |
| R5.2.4 | P1 | When the robber is moved, the current player may choose a player with a settlement touching the tile with the robber. The player chosen must give the current player a resource. |  | 2/26/2017 | J. Lynn |
|  |  |  |  |  |  |
| 6.0.0 |  | Build Phase: Current player should be able to build settlements, cities and roads at appropriate locations with required resources |  | 2/26/2017 | J. Lynn |
|  |  |  |  |  |  |
| R6.1.0 | P1 | Roads should be built when the player requests a road location which touches a city or is adjacent to a road owed by the player and said player has at least one brick and one lumber resource |  | 2/26/2017 | J. Lynn |
|  |  |  |  |  |  |
| R6.2.0 | P1 | Settlements should be built when the player requests a township location which touches a road owned by the player, the location is at least two township locations from any township, and the player has at least one brick, one lumber, one wool and one grain. |  | 2/26/2017 | J. Lynn |
|  |  |  |  |  |  |
| R6.3.0 | P1 | Cities should be built when the player requests a location with a settlement owned by the player, and the player has at least 2 ore and 3 grain resources |  | 2/26/2017 | J. Lynn |
|  |  |  |  |  |  |
| R6.4.0 | P1 | Buttons should be displayed when moves are available, notifications should be displayed when selections or illegal moves are made |  | 2/26/2017 | J. Lynn |
|  |  |  |  |  |  |
| R7.0.0 |  | The game should include development card play |  | 2/26/2017 | J. Lynn |
| R7.1.0 | P1 | There should be 25 development cards |  | 2/26/2017 | J. Lynn |
| R7.1.1 | P1 | There should be 14 knight cards. |  | 2/26/2017 | J. Lynn |
| R7.1.2 | P1 | There should be 5 Victory Point cards |  | 2/26/2017 | J. Lynn |
| R7.1.3 | P1 | There should be 2 road building cards |  | 2/26/2017 | J. Lynn |
| R7.1.4 | P1 | There should be 2 monopoly cards |  | 2/26/2017 | J. Lynn |
| R7.1.5 | P1 | There should be 2 year of plenty cards |  | 2/26/2017 | J. Lynn |
|  |  |  |  |  |  |
| R.7.2.0 | P1 | The current player should have access to their development cards |  | 2/26/2017 | J. Lynn |
|  |  |  |  |  |  |
| R7.3.0 | P1 | The current player should be able to purchase a development card if they have a minimum of one grain and one ore and one wool |  | 2/26/2017 | J. Lynn |
| R7.3.1 | P2 | The current player should be able to play development cards |  | 2/26/2017 | J. Lynn |
|  |  |  |  |  |  |
| R7.4.0 | P1 | When played, the knight card should enable the current player to move the robber to a different resource tile and claim one resource from one of the players (if any) with a settlement or a city which touches the tile. The Player's army size should also increase by one. | See R3.2.1 and R3.2.2 | 2/26/2017 | J. Lynn |
|  |  |  |  |  |  |
| R7.5.0 | P1 | The monopoly card should require the current player to select a resource, then all players must give the current player every resource they have of that type |  | 2/26/2017 | J. Lynn |
|  |  |  |  |  |  |
| R7.6.0 | P1 | The year of plenty card should require the current player to select any two resources to be added to their current resources |  | 2/26/2017 | J. Lynn |
|  |  |  |  |  |  |
| R7.7.0 | P1 | The victory point card should add one point to the players score. |  | 2/26/2017 | J. Lynn |
|  |  |  |  |  |  |
| R7.8.0 | P1 | The road building card should allow the current player to build two roads. | See R4.1.2 | 2/26/2017 | J. Lynn |
|  |  |  |  |  |  |
| R8.0.0 |  | Trade: Current players should be able to trade resources with other players and perform maritime trades. |  | 2/26/2017 | J. Lynn |
|  |  |  |  |  |  |
| R8.1.0 | P1 | The current player should be able to trade resources with any other player after an agreement has been made |  | 2/26/2017 | J. Lynn |
| R8.2.0 | P2 | Players should be able to trade any 4 resources of the same type for a single resource of their choice. |  | 2/26/2017 | J. Lynn |
| R8.3.0 | P2 | Current players should be able to trade two of a resource listed on a harbor for a resource of their choosing if they have a settle located on that harbor. | R1.2.1 | 2/26/2017 | J. Lynn |
| R8.4.0 | P2 | Current players should be able to trade 3 resources of any type for a single resource of their choosing if they have a settlement or city located on a generic harbor. | R1.2.2 | 2/26/2017 | J. Lynn |
|  |  |  |  |  |  |
| R9.0.0 |  | Scoring should be accurate |  |  |  |
|  |  |  |  |  |  |
| R9.1.0 | P1 | Each player should receive one point per settlement placed |  | 2/26/2017 | J. Lynn |
|  |  |  |  |  |  |
| R9.2.0 | P1 | Each player should receive two points per city |  | 2/26/2017 | J. Lynn |
|  |  |  |  |  |  |
| R9.3.0 | P2 | The first player to build a continuous road of 5 segments will receive two points for 'longest road' | Road forks should not be counted | 2/26/2017 | J. Lynn |
| R9.3.1 | P2 | If a player creates a continuous road which is longer than the continuous road built by the current player with the 'longest road', the points should be removed from the current longest road holder and given to the player who exceeded their road length. |  | 2/26/2017 | J. Lynn |
|  |  |  |  |  |  |
| 9.4.0 | P1 | Players should have a point for each victory point development card played. |  |  |  |
|  |  |  |  |  |  |
| R9.5.0 | P1 | The first player to play 3 Knight development cards should receive 2 victory points for largest army. |  | 2/26/2017 | J. Lynn |
| R9.5.1 | P1 | If a player's number of Knight cards played exceeds the number of Knight played by the current largest army point holder, the points for largest army should be taken from the current holder and added to the player who exceeded their 'army size'. |  | 2/26/2017 | J. Lynn |
|  |  |  |  |  |  |
| R9.6.0 | P1 | The first player to reach 10 points should win the game |  | 2/26/2017 | J. Lynn |
|  |  |  |  |  |  |
| R10.0.0 | P3 | AI players should be available |  | 2/26/2017 | J. Lynn |
| R10.1.0 | P3 | 1 - 3 AI players should be available |  | 2/26/2017 | J. Lynn |
| R10.2.0 | P3 | The AI players should have correct values for points, resources and development cards. |  | 2/26/2017 | J. Lynn |
| R10.3.0 | P3 | The AI players should be able to perform all functions that human players can. |  | 2/26/2017 | J. Lynn |

# Design Documentation

# High Level Architecture



**Figure 5.2.1: High-Level UML Interaction Diagram**

This is a description of the high-level architecture and class interactions of the program. More detailed UML diagrams, class structures, interactions, method descriptions and requirement references for the methods are included in the following sections. The source code includes pseudo-code incorporated into every method. The pseudo-code is included with each line of the code using // comments.

The program begins with the Game class, which contains the main() method and one GUI object. The GUI class is how the user interacts with the game. It has buttons that may be used to perform the various events. Each button has its own GUIHandler, which are nested inner classes of the GUI and implement the GUI and implement the ActionListener interface. These handlers use attributes from both the Board class and the PlayerManager class. These buttons also purchase and play objects extended from the DevelopmentCard abstract class and roll the Dice object. Thus, the GUI has attributes which include one Board object, one PlayerManager object, the deck of DevelopmentCards, which contains 25 cards, and the Dice object.

The Board object displays the Tile objects, which are contained as a linked list in the TileList object. Each Tile has a Location object. The Board also has two LocationList objects. These are linked lists of Location objects. One list for Road Locations and one list for Settlement/City Locations.

Location objects are designed to hold GamePiece objects. The Locations in the Tiles are designed so the Robber may be moved to the different Tiles. The LocationLists are designed to hold GamePiece objects which belong to the player, which are Road objects, Settlement object and City objects.

# Game Class



**Figure 5.2.1**

* Game class requirements: R1.5.0, R2.0.0, R4.0.0, R4.1.0 – 4.1.5, R5.4.2

The Game class creates a JFrame to hold other components, sets the scale, and is responsible for the set-up phase of the game. The game was designed to have a height of 1000 pixels. The scale is based off the ratio of the screen’s native resolution to 1000. The set up-phase is when each player takes turns setting up two Settlements with adjacent Roads and is accomplished with the methods placeInitialSettlements() and setInitialSettlements(). It also includes two static overloaded methods, which are called showSelectionBox(). These methods are used at various points in the program when the player must choose an item from a list or an array.

The game class also includes a help menu. When the help menu button is clicked, six help options are displayed. The user may click on the appropriate section or phase of the game they need help with.

# GUI Class



**Figure 5.3.1 – The GUI Class Diagram with Nested Inner Classes.**

* GUI class requirements: R2.0.0 through R2.4.6, R3.1.3, R5.1.0, R5.2.4, R7.1.0 – R7.1.5, R7.8.0
* GUIHandler: R2.0.0
* RoadHandler, SettlementHandler, CityHandler: R3.2.0
* BuyDevelopmentCardHandler: R7.3.0
* PlayDevelopmentCardHandler: R7.2.0
* TradeHandler: R8.0.0
* EndTurnHandler: R2.4.5

The GUI extends JPanel and has seven buttons, each with its own nested Handler class. The GUIHandler implements the ActionListener interface and abstracts the commonalities of each button. Each Handler extends the GUIHandler and implements the differences in the buttons in a method called performNextEvent(). The use cases are diagramed in figure 6.3.2. Other than the Handler called and LocationList used, the buttons which build GamePieces have the same sequence, which is diagramed in figure 6.3.3. The Build Development Card button simply adds a DevelopmentCard from the deck to the Player’s DevelopmentCard list. The Play Development Card button displays a selection box if the Player has any cards. If the player selects one, the playCard() method is called and the card is activated. The sequence diagram is shown in figure 6.3.4. The Trade button displays a selection box, which allows the user to select any number of their resources to trade, then a selection box is displayed, which allows the user to request any number of resources. For every Player in the game, starting with the first, if a Player has the requested resources, a trade is offered. The player may accept or reject the trade offer. The End Turn button checks with the PlayerManager to see if the player has earned the largest army, or if the Player has won the game. If the game is not over, the next Player becomes the current Player and the Dice are rolled for the new player.



**Figure 5.3.2 – Use Case Diagram for GUI Buttons**



**Figure 5.3.3 - Sequence Diagram for Building Game Pieces (Road, Settlement, City)**



**Figure 5.3.4 – Sequence Diagram for Playing Development Card**

# Board Class



**Figure 5.4.1 – The Board class Diagram**

* Board requirements: R1.0.0, R1.4.0, R3.0.0, R4.1.1, R4.1.3, R4.1.5, R5.2.1, R5.2.3, R5.2.4, R6.0.0 – R6.4.0, R7.4.0

The Board class is responsible for displaying the Tiles and GamePieces. It also pays resources to Players from the Tiles during the setup phase and after Dice rolls. It displays LocationButtons when the user requests to build a Road, Settlement, or City and they have the proper resources available. It also displays LocationButtons when the Robber is to be moved. GamePieces are placed by clicking the Location Buttons.

# Tile & Associated Classes



**Figure 5.5.1 – TileList Build with Associations**

* TileList class requirements: R1.0.0, R1.2.0 – 1.2.7
* Tile class requirements: R1.0.0, R1.2.0 – R1.2.7, R1.3.2, R1.3.3, R5.2.1
* Resource enumeration: R1.2.1 – 1.25
* NumberTokens: R1.3.0, R1.3.4
* NumberToken: R1.3.1, R1.3.2, R5.2.1
* Value enumeration: R1.3.1

The TileList object is a linked list of Tile objects. Each Tile has a Resource enumeration associated with it. Each Tile also has a NumberToken. Each NumberToken has a Value enumeration associated with it. Each NumberToken is built into the NumberTokens List so they may be shuffled and easily distributed to the tiles. The Tiles are stored in the TileList for the same reasons. The Tiles are built, then added to the The TileList is added to the Board to create the tiles.

# Location, LocationList, LocationButton Classes



**Figure 5.6.1 – LocationList class with Location class**

* Location class requirements: R3.2.2, R3.2.3, R5.2.1, R5.2.3, R6.0.0
* LocationList class requirements: R3.2.2, R3.2.3

The Location class is designed to hold GamePiece objects. They are in Tile centers so the

Robber can be placed in any Tile. For Roads, Settlements and Cities the Location objects are placed in LocationLists for iterative purposes. Two LocationLists are created. One to hold Roads and one to hold Settlements and Cities. The Road LocationList places Location objects along the edges of the tiles. The Settlement LocationList, which is used for Settlements and Cities, places Location objects at the corners of the Tiles.



**Figure 5.6.2 – LocationButton class**

* LocationButton class requirements: R6.4.0

LocationButton objects are designed to be placed at a Location objects. When a user requests to build a GamePiece (Road, Settlement or City), LocationButtons are placed on the Board where the GamePiece may be placed. When an event occurs that requires the Robber is to be moved, LocationButtons are placed at the Tile centers. When the user clicks the LocationButton, the Board attribute gamePieceReadyToPlace is placed at the Location of the LocationButton.

# Player and PlayerManager Classes



**Figure 5.7.1 – Player class, PlayerManager class and PlayerColor Enumeration**

* Player class requirements: R1.1.0 – R1.1.2, R2.2.0 – R2.2.2, R3.1.0 – R3.1.3, R4.1.5, R5.2.1, R5.2.2, R5.2.4, R6.1.0 – R6.1.2, R7.2.0 – R7.6.0, R8.1.0, R9.0.0, R9.5.0, R9.5.1
* PlayerManager class requirements: R1.1.0 – R1.1.2, R2.1.0 – R2.2.1, R2.4.5, R4.1.0, R5.2.2, R7.5.0, R8.1.0, R9.5.0, R9.5.1, R9.6.0
* PlayerColor enumeration requirements: R1.1.0 – R1.1.2

The Player class is responsible for creating GamePieces for the Player and keeping track of the Player’s GamePieces, Resources, DevelopmentCards, and score. It has a string representation of the Player’s information to display in the GUI.

PlayerManager class is responsible for creating and keeping track of the Player’s in the game. It contains the PlayerColor enumeration, which is used to differentiate the Players. The color of the Player’s GamePieces match the name enumeration name. The PlayerColor enumeration also has a Color attribute, which is used to set the color of the GUI for each Player during their turn.

# GamePiece Abstract Class with Subclasses



**Figure 5.8.1 – GamPiece abstract class with Robber, Road, Settlement, and City subclasses**

* GamePiece requirements: R3.0.0, R5.2.1, R6.0.0
* Robber requirements: R1.4.0, R5.2.3, R7.4.0
* Road requirements: R3.1.2, R3.2.1, R3.2.3
* Settlement requirements: R3.1.0, R3.2.1, R5.2.1, R9.1.0
* City requirements: R3.1.1, R3.2.1, R5.2.1, R9.2.0

The GamePiece subclasses create objects which are game pieces that may be placed on the Board. They are placed in Location objects using LocationButtons. The Robber can be placed at Tile centers. Roads can be placed in one LocationList, and Settlements and Cities may be placed in the other LocationList. Each GamePiece must override the place(Location) method, which adds them to the Board at the appropriate Location.

# DevelopmentCard Abstract Class with Subclasses



**Figure 5.9.1 – DevelopmentCard abstract class with subclasses.**

* DevelopmentCard requirements: R7.0.0, R7.2.0, 7.3.1
* KnightCard requirements: R7.1.1, R7.2.0, R7.4.0
* MonopolyCard requirements: R7.1.4, R7.2.0, R7.5.0
* RoadBuildingCard requirements: R7.1.3, R7.2.0, R7.8.0
* VictoryPointCard requirements: R7.1.2, R7.2.0, R7.7.0
* YearOfPlentyCard requirements: R7.1.5, R7.2.0, R7.6.0

The DevelopmentCard subclasses create the development cards. Each subclass must override the playCard() method, which allows the Player to receive the benefits. A toSring() method, must also be overridden.

# Dice Class

**Figure 5.10.1 – Dice class Figure 5.10.2 – Value Enumeration in NumberTokens class**

* Dice requirements: R5.0.0, R5.1.0
* Value enumeration requirements: R5.2.1

The Dice class returns two randomly generated numbers between 1 and 6 to simulate rolling two dice. Two dice must be rolled to correctly represent the odds for each roll. A NumberTokens.Value Enumeration is returned from the getValue() method, which depends on the roll. This Enumeration is checked against each Tile’s NumberToken value. When they match, the Tile pays appropriate resources to each Settlement or City at each Location on the corners of the Tile.

# Source Code

*See attached .zip file for source code.*

# Testing Strategy & Documentation

# Test Plan

Testing for Project JEM: “Settlers of Catan” requires a combination of black box and white box testing. Our black box test cases were developed from the requirements matrix, which ensures a user has all the necessary actions for the game to behave correctly. White box testing was developed directly from the source code by taking an iterative approach to each Java class. The result is that every function will have undergone rigorous testing to ensure accuracy and reliability. This approach will also pinpoint any bugs within the coded segments.

# Black Box Strategy

Black box testing will be completed by running the final version of the .jar file. The tester will evaluate each scenario in a fashion that simulates actual game play, as well as any other potential scenarios that could plausibly cause the software to crash. If the requirement is fulfilled, it will be marked as passed in the matrix. If it fails, it will be entered on the list of bugs matrix. Below is a broad overview of the black box matrix, a complete list off all specific testing cases will be included in following sections.

# White Box Strategy

White box testing will be completed in an IDE on the final source code files, and covers the unit testing phase of the plan. A debugger will be used to track the inputs/outputs of the code lines and functions. If the test case is met satisfactorily, the matrix will be marked as passed. If it fails, it will be entered on the list of bugs matrix. Below is a broad overview of the white box matrix, a complete list off all specific testing cases will be included in following sections.

# Test Cases

Below are the test cases for black box and white box, formatted in a general way. This is not the complete and extensive list of all requirement and source code test cases. It instead offers a general insight into the subjects of the specific cases. For a complete list of test cases and results, see the ‘Results’ section.

General Black Box Overview:

|  |  |  |
| --- | --- | --- |
| **Requirement** | **Rationale** | **Passed?** |
| R1.0.0 | The game should set up the board correctly |  |
| R2.0.0 | There is a Graphical User Interface |  |
| R3.0.0 | The proper number of settlements, cities, and roads for each player and proper settlement and road locations |  |
| R4.0.0 | There is a menu at the beginning of the game that executes |  |
| R5.0.0 | The game can execute the dice roll phase. |  |
| R6.0.0 | Build Phase: Current player can build settlements, cities, and roads at appropriate locations with required resources. |  |
| R7.0.0 | Game includes development card play |  |
| R8.0.0 | Trade: Current players can trade cards with other players and perform maritime trades. |  |
| R9.0.0 | Scoring is accurate |  |
| R10.0.0 | AI players are available |  |

General White Box Overview:

|  |  |  |
| --- | --- | --- |
| **Class** | **Purpose** | **Passed?** |
| Board Class | Displays Resource Tiles and Game Pieces placed on the board. Contains list of possible locations for GamePieces, and displays as buttons when player attempts to place a piece. |  |
| City Class | Creates a City |  |
| Dice Class | Simulates Two Dice Being Rolled |  |
| Game Class | Creates Frame for the Board and GUI that scales to the screen’s native image. Sets the initial Settlements and Roads for each player. |  |
| GamePiece Class | Abstracts the commonalities of each game piece and encapsulates the functionality |  |
| GUI Class | Displays the current Player's information and provides buttons with which the user may interact |  |
| +GUI Buy Development  Card Handler | Buy Development Card Button |  |
| +GUI City Handler | City Button |  |
| +GUI End Turn Handler | End Turn Button |  |
| +GUI GUI Handler | Setting Game piece |  |
| +GUI Play Development Card Handler | Play Development Card Button |  |
| +GUI Road Handler | Road Button |  |
| +GUI Settlement Handler | Settlement Button |  |
| +GUI Trade Handler | Trade Button |  |
| KnightCard Class | Creates a Knight Card |  |
| Location Class | Creates Locations, Which Allow the Placement of Game Pieces |  |
| LocationButton Class | Creates a button with a Location where a Game Piece can be placed. |  |
| LocationList Class | Creates a Settlement Location List, and Road Location List |  |
| MonopolyCard Class | Creates a Monopoly Card |  |
| NumberTokens Class | Creates 18 number tokens in random order, with Eight and Six as non-adjacent tiles |  |
| Player Class | Creates Players for the game |  |
| PlayerManager Class | Manages Player turns and resources in the game, sets player colors, checks for largest army, |  |
| Road Class | Creates a Road to be placed on the board |  |
| RoadBuildingCard  Class | Creates a Road Building Card |  |
| Robber Class | Creates a Robber for the board. |  |
| Settlement Class | Creates a Settlement to be placed on the board |  |
| Tile Class | Creates resource tiles, provides resource type |  |
| TileList Class | Creates a list for tiles and adds locations for all tile centers |  |
| VictoryPointCard Class | Creates a Victory Point Card. |  |
| YearOfPlentyCard  Class | Creates a Year of Plenty Card. |  |

# Results

Requirements & Black Box Testing Results:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Requirement Tested** | **Rationale** | **Input(s)** | **Expected Output** | **Passed?** | **Comments** |
| R1.0.0 | The game should set up the board correctly |  |  |  |  |
| R1.1.1 | User should be allowed to select 2 players | User selects "2 Players" when prompted | Game Cycles Through 2 Players/Colors | Pass |  |
| R1.1.2 | User should be allowed to select 3 players | User selects "3 Players" when prompted | Game Cycles Through 3 Players/Colors | Pass |  |
| R1.1.3 | User should be allowed to select 4 players | User selects "4 Players" when prompted | Game Cycles Through 4 Players/Colors | Pass |  |
| R1.2.0 | There are 19 resource tiles displayed | After selecting users, gameboard is setup | Show 19 resource tiles | Pass |  |
| R1.2.1 | There are 3 brick tiles displayed | After selecting users, gameboard is setup | Show 3 brick tiles | Pass |  |
| R1.2.2 | There are 3 ore tiles displayed | After selecting users, gameboard is setup | Show 3 ore tiles | Pass |  |
| R1.2.3 | There are 4 lumber tiles displayed | After selecting users, gameboard is setup | Show 4 lumber tiles | Pass |  |
| R1.2.4 | There are 4 wool tiles displayed | After selecting users, gameboard is setup | Show 4 wool tiles | Pass |  |
| R1.2.5 | There are 4 grain tiles displayed | After selecting users, gameboard is setup | Show 4 grain tiles | Pass |  |
| R1.2.6 | There is 1 desert tile displayed | After selecting users, gameboard is setup | Show 1 desert tile | Pass |  |
| R1.2.7 | Resource tiles are placed randomly on board | After selecting users, gameboard is setup | Random placement of tiles | Pass | Placement1: O,G,B,G,W,B,B,O,W,G,L,O,L,L,W,L,G,D,W |
| R1.2.7 | Resource tiles are placed randomly on board | After selecting users, gameboard is setup | Placement of tiles ≠ Case 11 | Pass | Placement2: W,L,L,G,W,D,O,B,G,G,B,W,B,G,L,O,W,O,L |
| R1.2.7 | Resource tiles are placed randomly on board | After selecting users, gameboard is setup | Placement of tiles ≠ Case 11, 12 | Pass | Placement3: O,G,L,L,G,L,B,L,W,W,W,O,O,B,G,B,G,W,D |
| R1.2.7 | Resource tiles are placed randomly on board | After selecting users, gameboard is setup | Placement of tiles ≠ Case 11, 12, 13 | Pass | Placement4: O,B,D,O,W,G,G,L,G,O,L,L,W,W,B,W,L,G,B |
| R1.3.0 | There are 18 number tokens | After gameboard is setup | Count 18 tokens | Pass | 18 Tokens over 19 Tiles |
| R1.3.1 | Tokens should be (2,3,3,4,4,5,5,6,6,8,8,9,9,10,10,11,11,12) | After gameboard is setup | Count 18 specific tokens | Pass | Placement1: 8,5,D,4,5,10,3,2,6,3,9,8,11,10,4,9,12,11,6 |
| R1.3.1 | Tokens should be (2,3,3,4,4,5,5,6,6,8,8,9,9,10,10,11,11,12) | After gameboard is setup | Count 18 specific tokens ≠ Case 18 | Pass | Placement2: 8,4,6,9,5,3,3,6,9,2,D,8,10,5,11,4,11,12,10 |
| R1.3.1 | Tokens should be (2,3,3,4,4,5,5,6,6,8,8,9,9,10,10,11,11,12) | After gameboard is setup | Count 18 specific tokens ≠ Case 18, 19 | Pass | Placement3: 8,10,6,2,11,4,5,D,6,11,5,8,4,3,3,12,9,9,10 |
| R1.3.1 | Tokens should be (2,3,3,4,4,5,5,6,6,8,8,9,9,10,10,11,11,12) | After gameboard is setup | Count 18 specific tokens ≠ Case 18, 19, 20 | Pass | Placement4: 8,9,6,3,5,11,9,6,10,4,D,8,5,4,12,11,2,3,10 |
| R1.3.2 | Each resource producing tile needs a token | After gameboard is setup | 1 token per resource tile | Pass |  |
| R1.3.3 | Desert tile does not have a token | After gameboard is setup | No tokens on the desert tile | Pass | Desert tile does not have a NUMBER token |
| R1.3.4 | Number tokens 8, 8, 6, 6, are not in adjacent resource tiles | After gameboard is setup | 8 and 6 are not adjacent; TestA | Pass |  |
| R1.3.4 | Number tokens 8, 8, 6, 6, are not in adjacent resource tiles | After gameboard is setup | 8 and 6 are not adjacent; TestB | Pass |  |
| R1.3.4 | Number tokens 8, 8, 6, 6, are not in adjacent resource tiles | After gameboard is setup | 8 and 6 are not adjacent; TestC | Pass |  |
| R1.3.4 | Number tokens 8, 8, 6, 6, are not in adjacent resource tiles | After gameboard is setup | 8 and 6 are not adjacent; TestD | Pass |  |
| R1.4.0 | There is 1 robber that starts on the desert tile | After gameboard is setup | 1 robber on desert | Pass |  |
| R1.5.0 | There are multiple window sizes to accomodate different resolutions | Resolution X selected | Game resizes to resolution X | Fail | Not Yet Implemented, In scope for future releases |
| R1.5.0 | There are multiple window sizes to accomodate different resolutions | Resolution Y selected | Game resizes to resolution Y | Fail | Not Yet Implemented, In scope for future releases |
| R1.5.0 | There are multiple window sizes to accomodate different resolutions | Resolution Z selected | Game resizes to resolution Z | Fail | Not Yet Implemented, In scope for future releases |
| R1.6.0 | There are 9 Harbor Pieces | After gameboard is setup | Count 9 harbors | Fail | Not Yet Implemented, In scope for future releases |
| R1.6.1 | There is 1 harbor for each resource | After harbors are placed | 1x wool, brick, ore, lumber, and grain harbor. | Fail | Not Yet Implemented, In scope for future releases |
| R1.6.2 | There are 4 generic harbors | After harbors are placed | 4x generic harbors | Fail | Not Yet Implemented, In scope for future releases |
| R2.0.0 | There is a Graphical User Interface |  |  |  |  |
| R2.1.0 | The correct score is displayed for each player | After gameboard is setup | Player Score is 0 | Pass |  |
|  |  |  |  |  |  |
| R2.2.0 | Current player is listed at the top of the GUI | After gameboard setup for 2 players | Purple Player is displayed at the top | Pass |  |
| R2.2.0 | Current player is listed at the top of the GUI | End Purple Player Turn | Orange Player is displayed at the top | Pass |  |
| R2.2.0 | Current player is listed at the top of the GUI | End Orange Player Turn | Player 1 is displayed at the top | Pass |  |
|  |  |  |  |  |  |
| R2.2.0 | Current player is listed at the top of the GUI | After gameboard is setup for 3 players | Purple Player is displayed at the top | Pass |  |
| R2.2.0 | Current player is listed at the top of the GUI | End Purple Player Turn | Orange Player is displayed at the top | Pass |  |
| R2.2.0 | Current player is listed at the top of the GUI | End Orange Player Turn | Green Player is displayed at the top | Pass |  |
| R2.2.0 | Current player is listed at the top of the GUI | End Green Player Turn | Purple Player is displayed at the top | Pass |  |
|  |  |  |  |  |  |
| R2.2.0 | Current player is listed at the top of the GUI | After gameboard is setup for 4 players | Purple Player is displayed at the top | Pass |  |
| R2.2.0 | Current player is listed at the top of the GUI | End Purple Player Turn | Orange Player is displayed at the top | Pass |  |
| R2.2.0 | Current player is listed at the top of the GUI | End Orange Player Turn | Green Player is displayed at the top | Pass |  |
| R2.2.0 | Current player is listed at the top of the GUI | End Green Player Turn | Blue Player is displayed at the top | Pass |  |
| R2.2.0 | Current player is listed at the top of the GUI | End Blue Player Turn | Purple Player is displayed at the top | Pass |  |
|  |  |  |  |  |  |
| R2.2.1 | Current Player's score is visible | Purple Player's Turn | Purple Player's Score is visible | Pass |  |
| R2.2.1 | Current Player's score is visible | Orange Player's Turn | Orange Player's Score is visible | Pass |  |
| R2.2.1 | Current Player's score is visible | Green Player's Turn | Green Player's Score is visible | Pass |  |
| R2.2.1 | Current Player's score is visible | Blue Player's Turn | Blue Player's Score is visible | Pass |  |
| R2.2.2 | Current Player's resources are visible | Purple Player's Turn | Purple Player's Resources are visible | Pass |  |
| R2.2.2 | Current Player's resources are visible | Orange Player's Turn | Orange Player's Resources are visible | Pass |  |
| R2.2.2 | Current Player's resources are visible | Green Player's Turn | Green Player's Resources are visible | Pass |  |
| R2.2.2 | Current Player's resources are visible | Blue Player's Turn | Blue Player's Resources are visible | Pass |  |
| R2.3.0 | Building costs are referenced in the GUI | After gameboard is setup | Show build costs in GUI | Pass | Shown at the bottom of Player box |
| R2.4.0 | There are buttons for the user to enter one of the game phases or end a turn | After initial settlements are placed | Buttons are listed, along with End Turn | Pass | Build Road, Build Settlement, Build City, Buy Dev Card, Use Dev Card, Trade, End Turn |
| R2.4.1 | There is a build settlement button | On Player Turn | Show build settlement button | Pass |  |
| R2.4.2 | There is a build city button | On Player Turn | Show build city button | Pass |  |
| R2.4.3 | There is a build road button | On Player Turn | Show build road button | Pass |  |
| R2.4.4 | There is a buy development card button | On Player Turn | Show buy development card button | Pass |  |
| R2.4.4 | There is a play development card button | On Player Turn | Show play development card button | Pass |  |
| R2.4.5 | There is a finish turn button which changes the current player to the next player | On Player Turn | Show finish turn button | Pass |  |
| R2.4.5 | There is a finish turn button which changes the current player to the next player | Click Finish Turn | Ends current player's turn, switches to next player | Pass |  |
| R2.4.6 | There is a trade button | On Player Turn | Show trade button | Pass |  |
| R3.0.0 | The proper number of settlements, cities, and roads for each player and proper settlement and road locations |  |  |  |  |
| R3.1.0 | There are 5 settlements for every player | Place settlement x5 | 5 Settlements are placed | Pass | Shown under player score, Settlement's Remaining |
| R3.1.0 | There are 5 settlements for every player | Place settlement x6 | Failed: Maximum Settlements Exceeded | Pass | "You are out of Settlements or Resources" |
| R3.1.1 | There are 4 cities for every player | Place city x 4 | 4 Cities are placed | Pass |  |
| R3.1.1 | There are 4 cities for every player | Place city x 5 | Failed: Maximum Cities Exceeded | Pass |  |
| R3.1.2 | There are 15 roads for every player | Place road x 15 | 15 roads are placed | Pass |  |
| R3.1.2 | There are 15 roads for every player | Place Road x 16 | Failed: Maximum Roads Exceeded | Pass |  |
| R3.1.3 | The players' quantity of unplaced cities, settlements, and roads are visible to them during their turn | Purple Player's Turn | Purple's quantity of cities, settlements, and roads visible | Pass |  |
| R3.1.3 | The players' quantity of unplaced cities, settlements, and roads are visible to them during their turn | Orange Player's Turn | Orange's quantity of cities, settlements, and roads visible | Pass |  |
| R3.1.3 | The players' quantity of unplaced cities, settlements, and roads are visible to them during their turn | Green Player's Turn | Green's quantity of cities, settlements, and roads visible | Pass |  |
| R3.1.3 | The players' quantity of unplaced cities, settlements, and roads are visible to them during their turn | Blue Player's Turn | Blue's quantity of cities, settlements, and roads visible | Pass |  |
| R3.2.0 | The build settlement, build road, and build city buttons are implemented | On Player Turn | Show build city, build settlement, and build road button | Pass |  |
| R3.2.1 | The placed cities, settlements, and roads are visible on the tiles at all times. | Place a city, settlement, or road | Remains visible on other player's turns. | Pass |  |
| R3.2.2 | Settlements and cities are allowed to be placed on tile corners | Build settlement on tile corner | Settlement is placed | Pass |  |
| R3.2.2 | Settlements and cities are allowed to be placed on tile corners | Build settlement on other tile locations | Settlement is not placed | Pass | Settlement Buttons only come up for legal locations |
| R3.2.3 | Roads are allowed to be placed along tile edges | Build road on tile edge | Road is placed | Pass |  |
| R3.2.3 | Roads are allowed to be placed along tile edges | Build road on other tile location | Road is not placed | Pass | Road Location Buttons only show for legal places |
| R4.0.0 | There is a menu at the beginning of the game that executes |  |  |  |  |
| R4.1.0 | Each player sets 2 settlements and 2 roads before play begins | After gameboard is setup | Quantity of roads and settlements = 2 per player | Pass |  |
| R4.1.1 | Each player takes a turn setting one settlement and one road | Purple Player places a settlement | Settlement placed, allows road to be built | Pass |  |
| R4.1.1 | Each player takes a turn setting one settlement and one road | Purple Player places a road | Road placed, switches to the next player | Pass |  |
| R4.1.1 | Each player takes a turn setting one settlement and one road | Orange Player places a settlement | Settlement placed, allows road to be built | Pass |  |
| R4.1.1 | Each player takes a turn setting one settlement and one road | Orange Player places a road | Road placed, switches to the next player | Pass |  |
| R4.1.1 | Each player takes a turn setting one settlement and one road | Green Player places a settlement | Settlement placed, allows road to be built | Pass |  |
| R4.1.1 | Each player takes a turn setting one settlement and one road | Green Player places a road | Road placed, switches to the next player | Pass |  |
| R4.1.1 | Each player takes a turn setting one settlement and one road | Blue Player places a settlement | Settlement placed, allows road to be built | Pass |  |
| R4.1.1 | Each player takes a turn setting one settlement and one road | Blue Player places a road | Road placed | Pass |  |
| R4.1.2 | Player 1 begins, then player 2 until the last player. | Purple, Orange, Green, and Blue End Turn | End of initial settlement placement | Pass |  |
| R4.1.3 | Once a settlement is placed, the player can place a road location next to the settlement. | Place settlement | Allow road to be built | Pass |  |
| R4.1.4 | For the second settlement, the players go in reverse order, beginning with the last player | Blue Player places a settlement | Settlement placed, allows road to be built | Pass |  |
| R4.1.4 | For the second settlement, the players go in reverse order, beginning with the last player | Blue Player places a road | Road placed, switches to previous player | Pass |  |
| R4.1.4 | For the second settlement, the players go in reverse order, beginning with the last player | Green Player places a settlement | Settlement placed, allows road to be built | Pass |  |
| R4.1.4 | For the second settlement, the players go in reverse order, beginning with the last player | Green Player places a road | Road placed, switches to previous player | Pass |  |
| R4.1.4 | For the second settlement, the players go in reverse order, beginning with the last player | Orange Player places a settlement | Settlement placed, allows road to be built | Pass |  |
| R4.1.4 | For the second settlement, the players go in reverse order, beginning with the last player | Orange Player places a road | Road placed, switches to previous player | Pass |  |
| R4.1.4 | For the second settlement, the players go in reverse order, beginning with the last player | Purple Player places a settlement | Settlement placed, allows road to be built | Pass |  |
| R4.1.4 | For the second settlement, the players go in reverse order, beginning with the last player | Purple Player places a road | Road placed | Pass |  |
| R4.1.5 | Each player receives one resource for every resource tile the player's initial settlement touches | Initial settlement touching wool | add 1 wool | Pass |  |
| R4.1.5 | Each player receives one resource for every resource tile the player's initial settlement touches | Initial settlement touching brick | add 1 brick | Pass |  |
| R4.1.5 | Each player receives one resource for every resource tile the player's initial settlement touches | Initial settlement touching lumber | add 1 lumber | Pass |  |
| R4.1.5 | Each player receives one resource for every resource tile the player's initial settlement touches | Initial settlement touching ore | add 1 ore | Pass |  |
| R4.1.5 | Each player receives one resource for every resource tile the player's initial settlement touches | Initial settlement touching grain | add 1 grain | Pass |  |
| R4.2.0 | There is an instruction manual in the menu. | Menu -> Instructions | Show instruction Manual | Fail | Menu Implemented, But Not Filled Out |
| R4.2.2 | There is a section on Rules. | Menu -> Rules | Show Rules | Fail | Menu Implemented, But Not Filled Out |
| R5.0.0 | The game can execute the dice roll phase. |  |  |  |  |
| R5.1.0 | There are two 6-sided dice | Roll dice | 2 dice roll | Pass | "You Rolled 11 5 And 6" |
| R5.2.0 | Each players turn begins with a dice roll | Beginning of turn | Dice roll | Pass | End Turn - Next Player Rolls Dice |
| R5.2.1 | Every tile with a number token that matches the current roll gives every settlement touching it one resource and every city gets two resources. | Roll 2, 3, 4, 5, 6, 8, 9, 10, 11, 12 | Settlements on this tile get 1 resource, cities on this tile get 2 | Pass |  |
| R5.2.2 | If a 7 is rolled, any player with more than 7 cards must select half of their resources (rounded down) and throw them out. | Roll 7, more than 7 cards | Discard half of resources rounded down | Pass | 13 Cards - Removed 6. 9 Cards - Removed 4 |
| R5.2.2 | If a 7 is rolled, any player with more than 7 cards must select half of their resources (rounded down) and throw them out. | Roll 7, less than 7 cards | Don't discard resources | Pass |  |
| R5.2.3 | If a 7 is rolled, the current player moves the robber. | Roll 7 | Move Robber | Pass | Unable to continue unless robber was moved |
| R5.2.4 | When a robber is moved, the current player may choose a player with a settlement touching the tile with the robber. The player chosen must give the current player a resource. | Move Robber | Select player touching the tile to give a resource | Pass | "Choose Your Target" - "You Stole Grain!" |
| R6.0.0 | Build Phase: Current player can build settlements, cities, and roads at appropriate locations with required resources. |  |  |  |  |
| R6.1.0 | Roads can be built when the player requests a road location which touches a city or is adjacent to a road owned by the player and said player has at least one brick and one lumber resource. | Build road to connecting city owned by same player | Road placed | Pass |  |
| R6.1.0 | Roads can be built when the player requests a road location which touches a city or is adjacent to a road owned by the player and said player has at least one brick and one lumber resource. | Build road to adjacent road owned by same player | Road placed | Pass |  |
| R6.1.0 | Roads can be built when the player requests a road location which touches a city or is adjacent to a road owned by the player and said player has at least one brick and one lumber resource. | Build road to connecting city owned by another player | Road not placed | Pass | Must own adjacent city/settlement |
| R6.1.0 | Roads can be built when the player requests a road location which touches a city or is adjacent to a road owned by the player and said player has at least one brick and one lumber resource. | Build road to adjacent road owned by another player | Road not placed | Pass | Must own adjacent road |
| R6.2.0 | Settlements can be built when the player requests a township location which touches a road owned by the player, the location is at least two township locations from any township, and the player has at least one brick, one lumber, one wool and one grain. | Build settlement touching road owned by player, 2 township locations away, and has 1 brick, 1 lumber, 1 wool, 1 grain | Settlement placed | Pass |  |
| R6.2.0 | Settlements can be built when the player requests a township location which touches a road owned by the player, the location is at least two township locations from any township, and the player has at least one brick, one lumber, one wool and one grain. | Build settlement touching road not owned by player, 2 township locations away, and has 1 brick, 1 lumber, 1 wool, 1 grain | Settlement not placed | Pass | Must own adjacent road |
| R6.2.0 | Settlements can be built when the player requests a township location which touches a road owned by the player, the location is at least two township locations from any township, and the player has at least one brick, one lumber, one wool and one grain. | Build settlement touching road owned by player, <2 township locations away, and has 1 brick, 1 lumber, 1 wool, 1 grain | Settlement not placed | Pass | Must be > 2 township locations |
| R6.2.0 | Settlements can be built when the player requests a township location which touches a road owned by the player, the location is at least two township locations from any township, and the player has at least one brick, one lumber, one wool and one grain. | Build settlement touching road owned by player, 2 township locations away, and has 0 brick, 1 lumber, 1 wool, 1 grain | Settlement not placed | Pass | Must have 1 brick |
| R6.2.0 | Settlements can be built when the player requests a township location which touches a road owned by the player, the location is at least two township locations from any township, and the player has at least one brick, one lumber, one wool and one grain. | Build settlement touching road owned by player, 2 township locations away, and has 1 brick, 0 lumber, 1 wool, 1 grain | Settlement not placed | Pass | Must Have 1 Lumber |
| R6.2.0 | Settlements can be built when the player requests a township location which touches a road owned by the player, the location is at least two township locations from any township, and the player has at least one brick, one lumber, one wool and one grain. | Build settlement touching road owned by player, 2 township locations away, and has 1 brick, 1 lumber, 0 wool, 1 grain | Settlement not placed | Pass | Must Have 1 Wool |
| R6.2.0 | Settlements can be built when the player requests a township location which touches a road owned by the player, the location is at least two township locations from any township, and the player has at least one brick, one lumber, one wool and one grain. | Build settlement touching road owned by player, 2 township locations away, and has 1 brick, 1 lumber, 1 wool, 0 grain | Settlement not placed | Pass | Must Have 1 Grain |
| R6.3.0 | Cities can be built when the player requests a location with a settlement owned by the player, and the player has at least 2 ore and 3 grain resources | Build city on settlement owned by player, has 2 ore and 3 grain | City Placed | Pass |  |
| R6.3.0 | Cities can be built when the player requests a location with a settlement owned by the player, and the player has at least 2 ore and 3 grain resources | Build city on settlement owned by player, has <2 ore and 3 grain | City not placed | Pass | Must have 2 Ore |
| R6.3.0 | Cities can be built when the player requests a location with a settlement owned by the player, and the player has at least 2 ore and 3 grain resources | Build city on settlement owned by player, has 2 ore and <3 grain | City not placed | Pass | Must Have 3 Grain |
| R6.3.0 | Cities can be built when the player requests a location with a settlement owned by the player, and the player has at least 2 ore and 3 grain resources | Build city on settlement not owned by player, has 2 ore and 3 grain | City not placed | Pass | Must Own Settlement |
| R6.5.0 | Buttons are displayed when moves are available, notifications are displayed when selections or illegal moves are made. | Build City Button When Not Enough Resources | Notification Not Enough Resources | Pass |  |
| R6.5.0 | Buttons are displayed when moves are available, notifications are displayed when selections or illegal moves are made. | Build Road Button When Not Enough Resources | Notification Not Enough Resources | Pass |  |
| R6.5.0 | Buttons are displayed when moves are available, notifications are displayed when selections or illegal moves are made. | Build Settlement Button When Not Enough Resources | Notification Not Enough Resources | Pass |  |
| R6.5.0 | Buttons are displayed when moves are available, notifications are displayed when selections or illegal moves are made. | Buy Dev Card When Not Enough Resources | Notification Not Enough Resources | Pass |  |
| R6.5.0 | Buttons are displayed when moves are available, notifications are displayed when selections or illegal moves are made. | Buy Settlement | Potential Settlement Locations Visible | Pass |  |
| R6.5.0 | Buttons are displayed when moves are available, notifications are displayed when selections or illegal moves are made. | Buy City | Potential City Locations Visible | Pass | "hidden" somewhat until you mouse over |
| R6.5.0 | Buttons are displayed when moves are available, notifications are displayed when selections or illegal moves are made. | Buy Road | Potential Road Locations Visible | Pass |  |
| R7.0.0 | Game includes development card play |  |  |  |  |
| R7.1.0 | There are 25 development cards | Initialize Game | 25 Possible Development Cards | Pass | Won't let you buy 26 |
| R7.1.1 | There are 14 knight cards | Buy 25 Cards | 14 Knights | Pass |  |
| R7.1.2 | There are 5 victory point cards | Buy 25 Cards | 5 Victory Points | Pass |  |
| R7.1.3 | There are 2 road building cards | Buy 25 Cards | 2 Road Building | Pass |  |
| R7.1.4 | There are 2 monopoly cards | Buy 25 Cards | 2 Monopoly | Pass |  |
| R7.1.5 | There are 2 year of plenty cards | Buy 25 Cards | 2 Year of Plenty | Pass |  |
| R7.2.0 | The current player has access to their development cards | 1 Development Card, Play | Development Card Played | Pass |  |
| R7.3.0 | The current player can purchase a development card if they have a minimum of one grain and one ore and one wool | Pay 1 grain, 1 ore, 1 wool | Buy development Card | Pass |  |
| R7.3.1 | The current player can play development cards purchased on previous turns | Pay 1 grain, 1 ore, 1 wool, end turn. Return to same player | Can Play Development Card | Pass | Can Play Card Same Turn |
| R7.4.0 | Knight card enables the current player to move the robber to a different resource tile and claim one resource from one of the players (if any) with a settlement or a city which touches the tile. | Play Knight Card | Move Robber, Claim Resource from Players with settlements touching the new tile | Pass |  |
| R7.5.0 | Monopoly card requires the current player to select a resource, then all players must give the current player every resource they have of that type | Play Monopoly Card - Select Lumber | All other players give Lumber to current player | Pass |  |
| R7.5.0 | Monopoly card requires the current player to select a resource, then all players must give the current player every resource they have of that type | Play Monopoly Card - Select Ore | All other players give Ore to current player | Pass |  |
| R7.5.0 | Monopoly card requires the current player to select a resource, then all players must give the current player every resource they have of that type | Play Monopoly Card - Select Brick | All other players give Brick to current player | Pass |  |
| R7.5.0 | Monopoly card requires the current player to select a resource, then all players must give the current player every resource they have of that type | Play Monopoly Card - Select Grain | All other players give Grain to current player | Pass |  |
| R7.5.0 | Monopoly card requires the current player to select a resource, then all players must give the current player every resource they have of that type | Play Monopoly Card - Select Wool | All other players give Wool to current player | Pass |  |
| R7.6.0 | Year of plenty card requires the current player to select any two resources to be added to their current resources | Play Year of Plenty Card, Select Lumber | Lumber added to resources | Pass |  |
| R7.6.0 | Year of plenty card requires the current player to select any two resources to be added to their current resources | Play Year of Plenty Card, Select Ore | Ore added to resources | Pass |  |
| R7.6.0 | Year of plenty card requires the current player to select any two resources to be added to their current resources | Play Year of Plenty Card, Select Brick | Brick added to resources | Pass |  |
| R7.6.0 | Year of plenty card requires the current player to select any two resources to be added to their current resources | Play Year of Plenty Card, Select Grain | Grain added to resources | Pass |  |
| R7.6.0 | Year of plenty card requires the current player to select any two resources to be added to their current resources | Play Year of Plenty Card, Select Wool | Wool added to resources | Pass |  |
| R7.7.0 | Victory point card adds one point to the players score. | Play Victory Point Card | Add 1 Victory Point to Player's score | Pass |  |
| R7.8.0 | Road building card allows the current player to build two roads | Play Road Building Card | Current Player Builds 2 Roads | Pass |  |
| R8.0.0 | Trade: Current players are able to trade cards with other players and perform maritime trades. |  |  |  |  |
| R8.1.0 | Current player can trade cards with any other player after an agreement has been made | Select Player and Card to Give | Other Player Selects Card, They Swap | Pass |  |
| R8.2.0 | Players are able to trade any 4 resources of the same type for a single resource of their choice. | Select 4 Player Resources and 1 New Resource | Obtain new Resource, Give up 4 other resources | Fail | Not Yet Implemented, In scope for future releases |
| R8.3.0 | Current players are able to trade a resource listed on a harbor for a resource of their choosing if they have a settle located on that harbor. | Specific Harbor, Trade 1 specific resource for 1 new resource | Obtain new Resource, Give up the 1 specific harbor resource | Fail | Not Yet Implemented, In scope for future releases |
| R8.4.0 | Current players can trade 3 resources of any type for a single resource of their choosing if they have a settlement or city located on a generic harbor. | Generic Harbor, Trade 3 Resources for 1 New Resource | Obtain New Resource, Give up 3 Other Resources | Fail | Not Yet Implemented, In scope for future releases |
| R9.0.0 | Scoring is accurate |  |  |  |  |
| R9.1.0 | Each player receives one point per settlement placed | Settlement Placed | Add 1 Victory Point to Player's Score | Pass |  |
| R9.2.0 | Each player receives two points per city | City Placed | Players Score is increased by 1 (+1 for original settlement) | Pass |  |
| R9.3.0 | First player to build a continuous road of 5 segments receives two points for 'longest road' | First Player to place 5 continuous roads | Player's Score is increased by 2 - awarded longest road | Fail | Not Yet Implemented, In scope for future releases |
| R9.3.1 | If a player creates a continous road which is longer than the continuous road built by the current player with the 'longest road', the points are removed from the current longest road holder and given to the player who exceeded their road length. | Another Player builds road longer than current longest road | New player's score is increased by 2, old player's score is decreased by 2 | Fail | Not Yet Implemented, In scope for future releases |
| R9.4.0 | Players receive a point for each victory point development card played. | Victory Card Played | Player's score is increased by 1 | Pass |  |
| R9.5.0 | The first player to play 3 Knight development cards receives 2 victory points for largest army. | First Player to use 3 knight cards | Player's score is increased by 2 - awarded largest army | Pass | This is awarded at end of turn |
| R9.5.1 | If a player's number of Knight cards played exceeds the number of Knight played by the current largest army point holder, the points for largest army are taken from the current holder and added to the player who exceeded their 'army size'. | Another player uses more knights than current largest army | New player's score is increased by 2, old player's score is decreased by 2 | Pass | Able to lose largest army to another player, then gain it back |
| R9.6.0 | The first player to reach 10 points wins the game | Player reaches 10 points | Player wins the game | Pass |  |
| R10.0.0 | AI players are available |  |  |  |  |
| R10.1.0 | 1 - 3 AI players are available | Select 1, 2, or 3 AI players | AI for players 1, 2, or 3 | Fail | Not Yet Implemented, In scope for future releases |
| R10.2.0 | The AI players have correct values for points, resources and development cards. | AI players selected | AI player has the same starting conditions as users | Fail | Not Yet Implemented, In scope for future releases |
| R10.3.0 | The AI players are able to perform all functions that human players can. | AI players turn | AI player has the same player options as users. | Fail | Not Yet Implemented, In scope for future releases |

Source Code & White Box Test Results:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test Case #** | **Requirement Tested** | **Rationale** | **Input(s)** | **Expected Output** | **Passed?** | **Comments** |
|  | City Class | City Class |  |  |  |  |
| 001 | City Class | Case Purple | Purple places a city | Purple city placed | Pass |  |
| 002 | City Class | Case Orange | Orange places a city | Orange city placed | Pass |  |
| 003 | City Class | Case Green | Green places a city | Green city placed | Pass |  |
| 004 | City Class | Case Blue | Blue places a city | Blue city placed | Pass |  |
| 005 | City Class | player.addCity(this); | City ready to place, but is unused | Unused city is added to inventory | Pass | Added to inventory, then resources are returned if not placed |
| 006 | City Class | setLocation(location.getPoint()); | Select a position for the city to be built | City icon is in that location | Pass |  |
| 007 | City Class | location.getGamePiece().addToPlayerInventory(); | Select a position that already has game piece | Game piece is added to inventory | Pass |  |
| 008 | City Class | player.addPoint(); | Place a city | Player gets a point | Pass | Total of 2 from City |
| 009 | City Class | player.removeCityResources(); | Place a city | Subtract resources (2 grain 3 ore?) | Pass |  |
| 010 | City Class | player.addResource(resource); | Place a city | Gain resource based on location | Pass |  |
| 011 | City Class | player.addResource(resource); | Place a city | Gain resource based on location | Pass |  |
|  | Dice Class | Dice Class |  |  |  |  |
| 001 | Dice Class | die1 = (int)(Math.random()\*6) + 1; | Roll Dice | Dice 1 is a random value between 1 and 6 | Pass |  |
| 002 | Dice Class | die2 = (int)(Math.random()\*6) + 1; | Roll Dice | Dice 2 is a random value between 1 and 6 | Pass |  |
| 003 | Dice Class | value = NumberTokens.Value.values()[die1 + die2 - 2]; | Dice 1+2 = 2 | Number Token 2 in Array = 0 | Pass |  |
| 004 | Dice Class | value = NumberTokens.Value.values()[die1 + die2 - 2]; | Dice 1+2 = 3 | Number Token 3 in Array = 1 | Pass |  |
| 005 | Dice Class | value = NumberTokens.Value.values()[die1 + die2 - 2]; | Dice 1+2 = 4 | Number Token 4 in Array = 2 | Pass |  |
| 006 | Dice Class | value = NumberTokens.Value.values()[die1 + die2 - 2]; | Dice 1+2 = 5 | Number Token 5 in Array = 3 | Pass |  |
| 007 | Dice Class | value = NumberTokens.Value.values()[die1 + die2 - 2]; | Dice 1+2 = 6 | Number Token 6 in Array = 4 | Pass |  |
| 008 | Dice Class | value = NumberTokens.Value.values()[die1 + die2 - 3]; | Dice 1+2 = 8 | Number Token 8 in Array = 5 | Pass |  |
| 009 | Dice Class | value = NumberTokens.Value.values()[die1 + die2 - 3]; | Dice 1+2 = 9 | Number Token 9 in Array = 6 | Pass |  |
| 010 | Dice Class | value = NumberTokens.Value.values()[die1 + die2 - 3]; | Dice 1+2 = 10 | Number Token 10 in Array = 7 | Pass |  |
| 011 | Dice Class | value = NumberTokens.Value.values()[die1 + die2 - 3]; | Dice 1+2 = 11 | Number Token 11 in Array = 8 | Pass |  |
| 012 | Dice Class | value = NumberTokens.Value.values()[die1 + die2 - 3]; | Dice 1+2 = 12 | Number Token 12 in Array = 9 | Pass |  |
| 013 | Dice Class | value = null; | Dice 1+2 = 7 | null / robber moves and steals | Pass |  |
| 014 | Dice Class | return die1+die2 + " "+die1+" AND "+die2; | Roll Dice | Return Dice 1 and Dice 2 | Pass |  |
|  | Game Class | Game Class |  |  |  |  |
| 001 | Game Class | setDefaultCloseOperation(EXIT\_ON\_CLOSE); | Close Game | Exit Application | Pass |  |
| 002 | Game Class | add(gui); | Add GUI | Display GUI frame | Pass |  |
| 003 | Game Class | add(board = gui.getBoard()); | Startup | Add board to GUI | Pass |  |
| 004 | Game Class | playerManager = gui.getPlayerManager(); | Startup | Get Players | Pass |  |
| 005 | Game Class | players = playerManager.getPlayers(); | Startup | # of players set | Pass |  |
| 006 | Game Class | setVisible(true); | Startup | Display GUI | Pass |  |
| 007 | Game Class | return playerManager.getCurrentPlayer(); | Current Player = Purple | Returns Purple | Pass |  |
| 008 | Game Class | return playerManager.getCurrentPlayer(); | Current Player = Orange | Returns Orange | Pass |  |
| 009 | Game Class | return playerManager.getCurrentPlayer(); | Current Player = Green | Returns Green | Pass |  |
| 010 | Game Class | return playerManager.getCurrentPlayer(); | Current Player = Blue | Returns Blue | Pass |  |
| 011 | Game Class | Game game = new Game(); | Startup | Sets up intial game board, asks for # of players | Pass |  |
| 012 | Game Class | game.placeInitialSettlements(); | Enter initial players | Game requests players to place initial settlements | Pass |  |
| 013 | Game Class | setInitialPieces(); | Enter initial players | Game requests players to place 1st initial settlement and road | Pass |  |
| 014 | Game Class | gui.getBoard().payInitialSettlements(); | Place 1st settlements | Add resources based on those settlements | Pass |  |
| 015 | Game Class | Collections.reverse(players); | Last player places his 1st initial settlement and road | Order is reversed | Pass |  |
| 016 | Game Class | setInitialPieces(); | Last player places his 1st initial settlement and road | Game requests players to place 2nd initial settlements, in reversed order | Pass |  |
| 017 | Game Class | Collections.reverse(players); | All players placed 2nd initial settlement and road | Game returns to original order | Pass |  |
| 018 | Game Class | gui.buildButtons(); | All players placed 2nd initial settlement and road | GUI adds the build buttons | Pass |  |
| 019 | Game Class | gui.populateDevelopmentCardDeck(); | All players placed 2nd initial settlement and road | Development card deck is created | Pass |  |
| 020 | Game Class | gui.rollDice(); | All players placed 2nd initial settlement and road | First player rolls dice | Pass |  |
| 021 | Game Class | repaint(); | Roll dice for the first time | GUI displays the build buttons | Pass |  |
| 022 | Game Class | for(Player player : players){ | Entered # of players | Allows players to place 1 settlement and 1 road - see below | Pass |  |
| 023 | Game Class | gui.setBackground(player.getColor()); | Select a current player | Background matches the current player | Pass |  |
| 024 | Game Class | gui.updatePlayerInfo(); | Select a current player | Player info matches the current player | Pass |  |
| 025 | Game Class | gui.getBoard().setGamePieceReadyToPlace(player.getSettlement()); | Select a current player | Current player has ability to place a settlement | Pass |  |
| 026 | Game Class | gui.getBoard().displaySetupSettlementLocations(); | Receive a settlement to place | GUI shows options of possible settlement locations | Pass |  |
| 027 | Game Class | repaint(); | Attempt to place a settlement | Shows settlement location buttons | Pass |  |
| 028 | Game Class | while(board.getGamePieceReadyToPlace() != null){ | Settlement ready to place | Wait for player | Pass |  |
| 029 | Game Class | try{Thread.sleep(25);}catch(Exception e){} } | Settlement ready to place | Wait for player | Pass |  |
| 030 | Game Class | gui.updatePlayerInfo(); | Initial settlement is placed | Current player info is updated | Pass |  |
| 031 | Game Class | board.clearButtons(); | Initial settlement is placed | Buttons are cleared off | Pass |  |
| 032 | Game Class | board.setGamePieceReadyToPlace(player.getRoad()); | Initial settlement is placed | Current player has ability to place a road | Pass |  |
| 033 | Game Class | board.displaySetupRoadLocations(getCurrentPlayer()); | Receive a road to place | GUI shows options of possible road locations | Pass |  |
| 034 | Game Class | repaint(); | Attempt to place a road | Show road location buttons | Pass |  |
| 035 | Game Class | while(board.getGamePieceReadyToPlace() != null){ | Road ready to place | Wait for player | Pass |  |
| 036 | Game Class | try{Thread.sleep(25);}catch(Exception e){} } | Road ready to place | Wait for player | Pass |  |
| 037 | Game Class | board.clearButtons(); | Initial Road is placed | Buttons are cleared off | Pass |  |
| 038 | Game Class | playerManager.getNextPlayer(); } } | Initial Road is placed | Go to the next player | Pass |  |
| 039 | Game Class | if(nonrepeatingList.contains(item)) | Duplicate Item for List | Item not added to list | Pass |  |
| 040 | Game Class | nonrepeatingList.add(item); } | New Item for List | Item Added to List | Pass |  |
| 041 | Game Class | Collections.sort(nonrepeatingList); | Unsorted List | Sorted List | Pass |  |
|  | GamePiece Class | GamePiece Class |  |  |  |  |
| 001 | GamePiece Class | return player; } | Player = Purple | Purple | Pass |  |
| 002 | GamePiece Class | return player; } | Player = Orange | Orange | Pass |  |
| 003 | GamePiece Class | return player; } | Player = Green | Green | Pass |  |
| 004 | GamePiece Class | return player; } | Player = Blue | Blue | Pass |  |
| 005 | GamePiece Class | super.setLocation (point.x-getWidth()/2, point.y-getHeight()/2); } | Place Gamepiece | Gamepiece Placed Direct Center of Point | Pass |  |
|  | GUI Class | GUI Class |  |  |  |  |
| 001 | GUI Class | setSize((int)(500\*scale),(int)(700\*scale)); | On Startup, Scale = 1 | GUI should be 500x700 px | Pass |  |
| 002 | GUI Class | setBorder(new BevelBorder(BevelBorder.RAISED)); | On Startup | Game board has a border | Pass |  |
| 003 | GUI Class | playerLabel.setBounds((int)(190\*scale), 0, (int)(300\*scale), (int)(500\*scale)); | On Startup, Scale = 1 | Player Label Location at the top right hand side | Pass |  |
| 004 | GUI Class | playerLabel.setHorizontalAlignment(JLabel.CENTER); | On Startup | Player Label is Centered | Pass |  |
| 005 | GUI Class | add(playerLabel); | On Startup | Player Label is displayed | Pass |  |
| 006 | GUI Class | buildingCostLabel.setBounds(0, (int)(500\*scale), (int)(500\*scale), (int)(200\*scale)); | On Startup, Scale = 1 | Build Cost Label is on the bottom right hand side | Pass |  |
| 007 | GUI Class | buildingCostLabel.setHorizontalAlignment(JLabel.CENTER); | On Startup | Build Cost Label is Centered | Pass |  |
| 008 | GUI Class | add(buildingCostLabel); | On Startup | Display Build Cost Label | Pass |  |
| 009 | GUI Class | buildingCostLabel.setText("<html><p style=fontsize:"+18\*scale+"px; text-align:center><u>Build Costs</u>"+"<p style=font-size:"+14\*scale+"px>Road: 1 Brick, 1 Lumber<br>"+"Settlement: 1 Brick, 1 Lumber, 1 Grain, 1 Wool<br>City: 2 Grain, 3 Ore<br>"+"Development Card: 1 Wool, 1 Grain, 1 Ore"); | On Startup | Build Cost Label Contains Road, Settlement, City, and Development Costs | Pass |  |
| 010 | GUI Class | developmentCards.add(new KnightCard(this.board)); | Populate Development Card Deck | 14 Knight Cards | Pass |  |
| 011 | GUI Class | developmentCards.add(new VictoryPointCard()); | Populate Development Card Deck | 5 Victory Point Cards | Pass |  |
| 012 | GUI Class | developmentCards.add(new MonopolyCard(playerManager.getPlayers())); | Populate Development Card Deck | 2 Monopoly Cards | Pass |  |
| 013 | GUI Class | developmentCards.add(new RoadBuildingCard(this)); | Populate Development Card Deck | 2 Road Building Cards | Pass |  |
| 014 | GUI Class | developmentCards.add(new YearOfPlentyCard()); } } | Populate Development Card Deck | 2 Year of Plenty Cards | Pass |  |
| 015 | GUI Class | Collections.shuffle(developmentCards); } | 14 Knight, 5 Victory Point, 2 Road Building, and 2 Year of Plenty Cards | Shuffled Development Cards | Pass |  |
| 016 | GUI Class | JOptionPane.showMessageDialog(this, "Roll The Dice!", "Settlers of Catan", JOptionPane.INFORMATION\_MESSAGE); | Beginning of Turn | Message to roll the dice | Pass |  |
| 017 | GUI Class | JOptionPane.showMessageDialog(this, "You rolled a "+dice.getRoll()+".", "Settlers of Catan", JOptionPane.INFORMATION\_MESSAGE); | Roll Dice | Message returning what was rolled | Pass |  |
| 018 | GUI Class | playerManager.checkResourcesSize(scale); | 7 Rolled | Remove Excess Resources from Players if Over 7 | Pass |  |
| 019 | GUI Class | board.setGamePieceReadyToPlaceRobber(); | 7 Rolled | Robber Movable | Pass |  |
| 020 | GUI Class | board.displayTileLocations(); | 7 Rolled | Setup Potential Robber Locations | Pass |  |
| 021 | GUI Class | getParent().repaint(); } | 7 Rolled | Display Potential Robber Locations | Pass |  |
| 022 | GUI Class | board.payResourceOnRoll(dice.getValue()); | Rolled 2-6, 8-12 | Pay resources based on tile rolled | Pass |  |
| 023 | GUI Class | updatePlayerInfo(); } | Resources Paid to Player | Update player info to show new resources | Pass |  |
| 024 | GUI Class | button.setVisible(false); } | Hide buttons | Buttons hidden | Pass |  |
| 025 | GUI Class | button.setVisible(true); } | Show buttons | Buttons shown | Pass |  |
| 026 | GUI Class | button.setBounds((int)(25\*scale), (int)((25+(i++)\*50)\*scale), (int)(150\*scale), (int)(50\*scale)); | Add Buttons | New Button Size 150x50, New Button Location = 50px lower than previous | Pass |  |
| 027 | GUI Class | currentPlayer = playerManager.getCurrentPlayer(); | Player's Resources Changed | Returns Current Player | Pass |  |
| 028 | GUI Class | currentPlayer.updateResourcesInfo(); | Player's Resources Changed | Updates Player's Resources | Pass |  |
| 029 | GUI Class | playerLabel.setText(currentPlayer.getInformation(scale)); } | Player's Resources Changed | Updates Label for Player's Resources | Pass |  |
| 030 | GUI Class | handlers.add(new RoadHandler()); | Button Setup | Build Road Button | Pass |  |
| 031 | GUI Class | handlers.add(new SettlementHandler()); | Button Setup | Build Settlement Button | Pass |  |
| 032 | GUI Class | handlers.add(new CityHandler()); | Button Setup | Build City Button | Pass |  |
| 033 | GUI Class | handlers.add(new BuyDevelopmentCardHandler()); | Button Setup | Buy Development Card Button | Pass |  |
| 034 | GUI Class | handlers.add(new PlayDevelopmentCardHandler()); | Button Setup | Play Development Card Button | Pass |  |
| 035 | GUI Class | handlers.add(new TradeHandler()); | Button Setup | Trade Button | Pass |  |
| 036 | GUI Class | handlers.add(new EndTurnHandler()); | Button Setup | End Turn Button | Pass |  |
|  | GUI Class BuyDevelopmentCardHandler | public class BuyDevelopmentCardHandler extends GUIHandler |  |  |  |  |
| 001 | GUI Class BuyDevelopmentCardHandler | currentPlayer.payForDevelopmentCard(); | Buy Development Card button pressed, has required resources | Removes Required Resources from Player | Pass |  |
| 002 | GUI Class BuyDevelopmentCardHandler | developmentCards.get(0).setPlayer(currentPlayer); | Buy Development Card button pressed, has required resources | Setup Current Player to Receive Card | Pass |  |
| 003 | GUI Class BuyDevelopmentCardHandler | currentPlayer.addDevelopmentCard(developmentCards.remove(0)); } | Buy Development Card button pressed, has required resources | Add Development Card to Current Player | Pass |  |
| 004 | GUI Class BuyDevelopmentCardHandler | JOptionPane.showMessageDialog(null, "You dont have the resources, or there are no more Development Cards", "Settlers of Catan", JOptionPane.ERROR\_MESSAGE); } | Buy Development Card button pressed, does not have required resources | Popup Error Not Enough Resources | Pass |  |
| 005 | GUI Class BuyDevelopmentCardHandler | return "Buy Dev Card"; } } | Add Buy Development Card Button | "Buy Dev Card" Button Text | Pass |  |
|  | GUI Class CityHandler | public class CityHandler extends GUIHandler |  |  |  |  |
| 001 | GUI Class CityHandler | board.setGamePieceReadyToPlace(currentPlayer.getCity()); | Build City button pressed, has required resources | Game Piece Ready To Place is City | Pass |  |
| 002 | GUI Class CityHandler | board.displayCityLocations(currentPlayer); } | Build City button pressed, has required resources | Potential City Locations Displayed | Pass |  |
|  | GUI Class EndTurnHandler | public class EndTurnHandler extends GUIHandler |  |  |  |  |
| 001 | EndTurnHandler | playerManager.checkLargestArmy(currentPlayer); | End Turn, Largest Army / Not Largest | New Largest Army Awarded / No Change | Pass |  |
| 002 | EndTurnHandler | playerManager.checkForVictory(); | End Turn, 10 Victory Points | Player Won The Game | Pass |  |
| 003 | EndTurnHandler | currentPlayer = playerManager.getNextPlayer(); | End Turn Pressed | Next Player's Turn | Pass |  |
| 004 | EndTurnHandler | setBackground(currentPlayer.getColor()); | Current Player = Purple | Background Color is Purple | Pass |  |
| 005 | EndTurnHandler | setBackground(currentPlayer.getColor()); | Current Player = Orange | Background Color is Orange | Pass |  |
| 006 | EndTurnHandler | setBackground(currentPlayer.getColor()); | Current Player = Green | Background Color is Green | Pass |  |
| 007 | EndTurnHandler | setBackground(currentPlayer.getColor()); | Current Player = Blue | Background Color is Blue | Pass |  |
| 008 | EndTurnHandler | updatePlayerInfo(); | End Turn Pressed | Player's info is updated | Pass |  |
| 009 | EndTurnHandler | rollDice(); } | End Turn Pressed | Next Player's Dice is Rolled | Pass |  |
| 010 | EndTurnHandler | return "End Turn"; }  } } | Add End Turn Button | "End Turn" Button Text | Pass |  |
|  | GUI Class GUIHandler | public abstract class GUIHandler implements ActionListener { |  |  |  |  |
| 001 | GUI Class GUIHandler | JOptionPane.showMessageDialog(GUI.this, "You Must Place The Robber!!", "CATAN", JOptionPane.ERROR\_MESSAGE); | Moving Robber | Popup Message to Move Robber | Pass |  |
| 002 | GUI Class GUIHandler | board.clearButtons(); | Initiate Placing Game Piece | Button Menu Hidden | Pass |  |
| 003 | GUI Class GUIHandler | board.getGamePieceReadyToPlace().addToPlayerInventory(); | No Gamepiece Selected | Current Gamepiece is Selected | Pass |  |
| 004 | GUI Class GUIHandler | board.setGamePieceReadyToPlace(null); } | Game Piece is Selected | Game Piece Ready To Place is Empty | Pass |  |
| 005 | GUI Class GUIHandler | updatePlayerInfo(); | Game Piece Played | Player is Updated | Pass |  |
|  | GUI Class PlayDevelopmentCardHandler | public class PlayDevelopmentCardHandler extends GUIHandler |  |  |  |  |
| 001 | GUI Class PlayDevelopmentCardHandler | currentPlayer.chooseDevelopmentCard(); | Play Development Card button pressed, has development card | Player allowed to choose which development card to play | Pass |  |
| 002 | GUI Class PlayDevelopmentCardHandler | JOptionPane.showMessageDialog(null, "You dont have any Development Cards!", "Settlers of Catan", JOptionPane.ERROR\_MESSAGE); } | Play Development Card button pressed, does not have development card | Popup Error No Development Cards Found | Pass | "You don’t have any Development Cards!" |
| 003 | GUI Class PlayDevelopmentCardHandler | return "Use Dev Card"; } } | Add Use Development Card Button | "Use Dev Card" Button Text | Pass |  |
|  | GUI Class RoadHandler | public class RoadHandler extends GUIHandler |  |  |  |  |
| 001 | GUI Class RoadHandler | board.setGamePieceReadyToPlace(currentPlayer.getRoad()); | Build Road button pressed, has required resources | Game Piece Ready To Place is Road | Pass |  |
| 002 | GUI Class RoadHandler | board.displayRoadLocations(currentPlayer); } | Build Road button pressed, has required resources | Potential Road Locations Displayed | Pass |  |
| 003 | GUI Class RoadHandler | JOptionPane.showMessageDialog(board, "You are out Roads or resources!", "Settlers of Catan", JOptionPane.ERROR\_MESSAGE); } | Build Road button pressed, does not have required resources | Popup Error Not Enough Resources | Pass | "You are out Roads or resources!" |
| 004 | GUI Class RoadHandler | return "Build Road"; } } | Add Road Button | "Build Road" Button Text | Pass |  |
|  | GUI Class SettlementHandler | public class SettlementHandler extends GUIHandler |  |  |  |  |
| 001 | GUI Class SettlementHandler | board.setGamePieceReadyToPlace(currentPlayer.getSettlement()); | Build Settlement button pressed, has required resources | Game Piece Ready To Place is Settlement | Pass |  |
| 002 | GUI Class SettlementHandler | board.displaySettlementLocations(currentPlayer); } | Build Settlement button pressed, has required resources | Potential Settlement Locations Displayed | Pass |  |
| 003 | GUI Class SettlementHandler | JOptionPane.showMessageDialog(board, "You are out Settlements or resources!", "Settlers of Catan", JOptionPane.ERROR\_MESSAGE); } | Build Settlement button pressed, does not have required resources | Popup Error Not Enough Resources | Pass | "You are out Settlements or resources!" |
| 004 | GUI Class SettlementHandler | return "Build Settlement"; } } | Add Settlement Button | "Build Settlement" Button Text | Pass |  |
|  | GUI Class TradeHandler | public class TradeHandler extends GUIHandler |  |  |  |  |
| 001 | GUI Class TradeHandler | Collections.sort(currentPlayer.getResources()); | Trade Button Pressed | Current Player's Resources List is Sorted | Pass |  |
| 002 | GUI Class TradeHandler | int index = JOptionPane.showOptionDialog(null, "Choose a Resource to offer:", "Settlers of Catan", JOptionPane.YES\_NO\_OPTION, JOptionPane.QUESTION\_MESSAGE, null, currentPlayer.getResources().toArray(),null); | Trade Button Pressed | Popup Message to Choose a Trade Offer | Pass |  |
| 003 | GUI Class TradeHandler | tradeOffer.add(currentPlayer.getResources().remove(index)); | Offer Selected | Remove Offer from current Player, Add to Offer List | Pass |  |
| 004 | GUI Class TradeHandler | choice = JOptionPane.showConfirmDialog(null, "Would you like to add another resource?", "Settlers of Catan", JOptionPane.YES\_NO\_OPTION); } | Offer Selected | Popup Message to Choose if player wants to add to offer | Pass |  |
| 005 | GUI Class TradeHandler | int index = JOptionPane.showOptionDialog(null, "Request a resource:", "Settlers of Catan", JOptionPane.YES\_NO\_OPTION, JOptionPane.QUESTION\_MESSAGE, null, Tile.Resource.values(),null); | Initial Offer Selected | Popup Message to Request a Resource | Pass |  |
| 006 | GUI Class TradeHandler | tradeRequest.add(Tile.Resource.values()[index]); | Request Selected | Add Offer to Requested List | Pass |  |
| 007 | GUI Class TradeHandler | choice = JOptionPane.showConfirmDialog(null, "Would you like to request another resource?", "Settlers of Catan", JOptionPane.YES\_NO\_OPTION); } | Request Selected | Popup Message to Choose if player wants to add to request | Pass |  |
| 008 | GUI Class TradeHandler | p.getResources().remove(resource); | 2nd Trading Player has the Requested Resources | Remove Requested Resources from 2nd Trading Player | Pass |  |
| 009 | GUI Class TradeHandler | temp.add(resource); } | 2nd Trading Player removed the Requested Resources | Add requested resources to temp list | Pass |  |
| 010 | GUI Class TradeHandler | hasResourcesToTrade = false;     } | 2nd Trading Player does not have requested resources | Cannot Trade | Pass | There is no notification |
| 011 | GUI Class TradeHandler | p.addResource(resource); | 2nd Trading Player removed the Requested Resources | 2nd Trading Player is given back his temporary Requested Resource | Pass |  |
| 012 | GUI Class TradeHandler | p.updateResourcesInfo(); | 2nd Trading Player Was Given Back Resource | Resource info is updated for 2nd player | Pass |  |
| 013 | GUI Class TradeHandler | tradeResponce = JOptionPane.showConfirmDialog(null, p.getInformation(scale)+"<br><br>Trade your "+tradeRequest+" for "+tradeOffer, "Settlers of Catan", JOptionPane.INFORMATION\_MESSAGE); | Has Resources to Trade | Popup Message Confirming Trade | Pass |  |
| 014 | GUI Class TradeHandler | currentPlayer.addResource(p.removeResource(resource)); } | Trade Confirmed | Add Requested Resource to 1st Trader, Remove from 2nd Trader | Pass |  |
| 015 | GUI Class TradeHandler | p.addResource(resource); | Trade Confirmed | 2nd Trading Player is added the offered resource | Pass |  |
| 016 | GUI Class TradeHandler | currentPlayer.addResource(resource); } | Trade Confirmed | 1st Trading Player is added the Requested Resource | Pass |  |
| 017 | GUI Class TradeHandler | return "Trade"; } } | Add Trade Button | "Trade" Button Text | Pass |  |
|  | KnightCard Class | KnightCard Class |  |  |  |  |
| 001 | KnightCard Class | board.setGamePieceReadyToPlaceRobber(); | Play Knight Card | Move Robber | Pass |  |
| 002 | KnightCard Class | board.displayTileLocations(); | Play Knight Card | Show Potential Robber Locations | Pass |  |
| 003 | KnightCard Class | player.addKnight(); } | Play Knight Card | Add 1 to Knight Count | Pass |  |
|  | Location Class | Location Class |  |  |  |  |
| 001 | Location Class | point = p; } | Location Point is Passed | Point is set to the new location | Pass |  |
| 002 | Location Class | return point; } | Location Point is Requested | Location Point is Returned | Pass |  |
| 003 | Location Class | gamePiece = gp; } | Game Piece is passed | Game Piece is set to the new Piece | Pass |  |
| 004 | Location Class | gamePiece = null; } | Game Piece is Cleared | Game Piece is set to null | Pass |  |
| 005 | Location Class | return gamePiece; } | Game Piece is Requested | Game Piece is Returned | Pass |  |
| 006 | Location Class | index = i; } | Index is Passed | Index is set to the new index | Pass |  |
| 007 | Location Class | return index; } | Index is Requested | Index is Returned | Pass |  |
|  | LocationButton Class | LocationButton Class |  |  |  |  |
| 001 | LocationButton Class | location = l; } | Location is passed | Location is set to the new location | Pass |  |
| 002 | LocationButton Class | return location; } | Location is Requested | Location is Returned | Pass |  |
|  | LocationList Class | LocationList Class |  |  |  |  |
| 001 | LocationList Class | add(new Location(new Point( (int)((222+86\*(j-i))\*scale), (int)(((50+150\*i) + (j%2==0 ? 50 : 0))\*scale)))); | Gameboard Setup | Add settlement locations for top half of map | Pass |  |
| 002 | LocationList Class | add(new Location(new Point( (int)((222+86\*(j-i))\*scale), (int)(((800-150\*i) + (j%2==0 ? 0 : 50))\*scale)))); } | Gameboard Setup | Add settlement locations for bottom half of map | Pass |  |
| 003 | LocationList Class | add(new Location(new Point((int)((265+86\*(j-i))\*scale),(int)((75+150\*i)\*scale)))); | Gameboard Setup | Add diagonal road locations for top half of map | Pass |  |
| 004 | LocationList Class | add(new Location(new Point((int)((265+86\*(j-i))\*scale),(int)((825-150\*i)\*scale)))); } | Gameboard Setup | Add diagonal road locations for bottom half of map | Pass |  |
| 005 | LocationList Class | add(new Location(new Point((int)((222+172\*j-86\*i)\*scale),(int)((150+150\*i)\*scale)))); | Gameboard Setup | Add straight road locations for top half of map | Pass |  |
| 006 | LocationList Class | add(new Location(new Point((int)((222+172\*j-86\*i)\*scale),(int)((750-150\*i)\*scale)))); } | Gameboard Setup | Add straight road locations for bottom half of map | Pass |  |
|  | MonopolyCard Class | MonopolyCard Class |  |  |  |  |
| 001 | MonopolyCard Class | Tile.Resource resource = Game.showSelectionBox(null, "Choose a Resource", Tile.Resource.values()); | Chosen Resource | Resource Chosen to Take | Pass |  |
| 002 | MonopolyCard Class | iterator = p.getResources().listIterator(); | Play Monopoly Card | Takes resources from another player | Pass |  |
| 003 | MonopolyCard Class | player.addResource(resource); | Resource Taken | Add resource to owning player | Pass |  |
|  | NumberTokens Class | NumberTokens Class |  |  |  |  |
| 001 | NumberTokens Class | add(new NumberToken(Value.THREE)); | Add Number Tokens | Two "3" tokens | Pass |  |
| 002 | NumberTokens Class | add(new NumberToken(Value.FOUR)); | Add Number Tokens | Two "4" tokens | Pass |  |
| 003 | NumberTokens Class | add(new NumberToken(Value.FIVE)); | Add Number Tokens | Two "5" tokens | Pass |  |
| 004 | NumberTokens Class | add(new NumberToken(Value.NINE)); | Add Number Tokens | Two "9" tokens | Pass |  |
| 005 | NumberTokens Class | add(new NumberToken(Value.TEN)); | Add Number Tokens | Two "10" tokens | Pass |  |
| 006 | NumberTokens Class | add(new NumberToken(Value.ELEVEN)); } | Add Number Tokens | Two "11" tokens | Pass |  |
| 007 | NumberTokens Class | add(new NumberToken(Value.TWO)); | Add Number Tokens | One "2" token | Pass |  |
| 008 | NumberTokens Class | add(new NumberToken(Value.TWELVE)); | Add Number Tokens | One "12" token | Pass |  |
| 009 | NumberTokens Class | Collections.shuffle(this); | All tokens added | Shuffled Tokens | Pass |  |
| 010 | NumberTokens Class | add(0, new NumberToken(Value.EIGHT)); | Shuffled Number Tokens | First "8" token in specific location | Pass |  |
| 011 | NumberTokens Class | add(4, new NumberToken(Value.SIX)); | Shuffled Number Tokens | First "6" token in specific location | Pass |  |
| 012 | NumberTokens Class | add(14, new NumberToken(Value.SIX) ); | Shuffled Number Tokens | Second "6" token in specific location | Pass |  |
| 013 | NumberTokens Class | add(17, new NumberToken(Value.EIGHT)); } | Shuffled Number Tokens | Second "8" token in specific location | Pass |  |
| 014 | NumberTokens Class | path = "/images/numbers/Two.png"; | Number "2" Token | Path to "2" Image | Pass |  |
| 015 | NumberTokens Class | path = "/images/numbers/Three.png"; | Number "3" Token | Path to "3" Image | Pass |  |
| 016 | NumberTokens Class | path = "/images/numbers/Four.png"; | Number "4" Token | Path to "4" Image | Pass |  |
| 017 | NumberTokens Class | path = "/images/numbers/Five.png"; | Number "5" Token | Path to "5" Image | Pass |  |
| 018 | NumberTokens Class | path = "/images/numbers/Six.png"; | Number "6" Token | Path to "6" Image | Pass |  |
| 019 | NumberTokens Class | path = "/images/numbers/Eight.png"; | Number "8" Token | Path to "8" Image | Pass |  |
| 020 | NumberTokens Class | path = "/images/numbers/Nine.png"; | Number "9" Token | Path to "9" Image | Pass |  |
| 021 | NumberTokens Class | path = "/images/numbers/Ten.png"; | Number "10" Token | Path to "10" Image | Pass |  |
| 022 | NumberTokens Class | path = "/images/numbers/Eleven.png"; | Number "11" Token | Path to "11" Image | Pass |  |
| 023 | NumberTokens Class | path = "/images/numbers/Twelve.png"; } | Number "12" Token | Path to "12" Image | Pass |  |
| 024 | NumberTokens Class | JOptionPane.showMessageDialog(null, "Can not create image.", "Settler's of Catan", JOptionPane.ERROR\_MESSAGE); }} | No Image | Error Message Popup | Pass | "Can Not Create Image" - Goes to white screen |
|  | Player Class | Player Class |  |  |  |  |
| 001 | Player Class | addResource(Tile.Resource.BRICK); | Initialize Game | Add 4 Brick to each player | Pass | Used to create initial settlements and roads |
| 002 | Player Class | addResource(Tile.Resource.LUMBER); | Initialize Game | Add 4 Lumber to each player | Pass | Used to create initial settlements and roads |
| 003 | Player Class | addResource(Tile.Resource.GRAIN); | Initialize Game | Add 2 Grain to each player | Pass | Used to create initial settlements and roads |
| 004 | Player Class | addResource(Tile.Resource.WOOL); } } } | Initialize Game | Add 2 Wool to each player | Pass | Used to create initial settlements and roads |
| 005 | Player Class | return playerColor.getColor(); } | Request Player Color | Return Player Color | Pass |  |
| 006 | Player Class | return playerColor; } | Request Player Color | Return Player Color | Pass |  |
| 007 | Player Class | return score; } | Request Score | Return Score | Pass |  |
| 008 | Player Class | return settlements.remove(0); } | Request Settlement | Return Settlement | Pass |  |
| 009 | Player Class | return settlements.size(); } | Request # of Settlements | Return # of Settlements | Pass |  |
| 010 | Player Class | return cities.remove(0); } | Request City | Return City | Pass |  |
| 011 | Player Class | return cities.size(); } | Request # of Cities | Return # of Cities | Pass |  |
| 012 | Player Class | return roads.remove(0); } | Request Road | Return Road | Pass |  |
| 013 | Player Class | return roads.size(); } | Request # of Roads | Return # of Roads | Pass |  |
| 014 | Player Class | settlements.add(settlement); } | Add Settlement | Add Settlement to Settlement List | Pass |  |
| 015 | Player Class | roads.add(road); } | Add Road | Add Road to Road List | Pass |  |
| 016 | Player Class | cities.add(city); } | Add City | Add City to City List | Pass |  |
| 017 | Player Class | ++score; } | Add Point | Add Point to Score Count | Pass |  |
| 018 | Player Class | ++armySize; } | Play Knight | Add Knight to Army Size | Pass |  |
| 019 | Player Class | return armySize; } | Request Army Size | Return Army Size | Pass |  |
| 020 | Player Class | return hasLargestArmy; } | Request Has Largest Army | Return Largest Army Boolean | Pass |  |
| 021 | Player Class | score += 2; } | Player has largest army | Add 2 Score | Pass |  |
| 022 | Player Class | score -= 2; } | Player no longer has largest army | Remove 2 Score | Pass |  |
| 023 | Player Class | return resources; } | Request Resources | Return Resources | Pass |  |
| 024 | Player Class | return resources.size(); } | Request # of Resources | Return # of Resources | Pass |  |
| 025 | Player Class | resources.add(r); } | Add Resource | Add Resource to Resource List | Pass |  |
| 026 | Player Class | resources.remove(r); | Remove Resource | Remove Resource from Resource List | Pass |  |
| 027 | Player Class | if(resourceTypes.contains(resource)) | Resource Already Listed | Don't Add Duplicate | Pass |  |
| 028 | Player Class | resourceTypes.add(resource); } | Resource Not Listed | Add to Resouce Types List | Pass |  |
| 029 | Player Class | Collections.sort(resourceTypes); | List of Resource Types | Sorted List of Resource Types | Pass |  |
| 030 | Player Class | Collections.shuffle(resources); | Resources Tiles | Shuffled Resource Tiles | Pass |  |
| 031 | Player Class | return true; | brick > 0 && lumber > 0 | has road resources | Pass |  |
| 032 | Player Class | else return false; } | brick <= 0 || lumber <= 0 | does not have road resources | Pass |  |
| 033 | Player Class | resources.remove(Tile.Resource.BRICK); | build road | remove 1 brick from player's resources | Pass |  |
| 034 | Player Class | resources.remove(Tile.Resource.LUMBER); } | build road | remove 1 lumber from player's resources | Pass |  |
| 035 | Player Class | return true; | brick >0 && lumber >0 && grain > 0 && wool > 0 | has settlement resources | Pass |  |
| 036 | Player Class | return false; } | brick <= 0 || lumber <= 0 || grain <= 0 || wool <= 0 | does not have settlement resources | Pass |  |
| 037 | Player Class | resources.remove(Tile.Resource.BRICK); | build settlement | remove 1 brick from player's resources | Pass |  |
| 038 | Player Class | resources.remove(Tile.Resource.LUMBER); | build settlement | remove 1 lumber from player's resources | Pass |  |
| 039 | Player Class | resources.remove(Tile.Resource.GRAIN); | build settlement | remove 1 grain from player's resources | Pass |  |
| 040 | Player Class | resources.remove(Tile.Resource.WOOL); } | build settlement | remove 1 wool from player's resources | Pass |  |
| 041 | Player Class | return true; | ore > 2 && grain > 1 | has city resources | Pass |  |
| 042 | Player Class | return false; } | ore <=2 || grain <= 1 | does not have city resources | Pass |  |
| 043 | Player Class | resources.remove(Tile.Resource.GRAIN); | build city | remove 1st grain from player's resources | Pass |  |
| 044 | Player Class | resources.remove(Tile.Resource.GRAIN); | build city | remove 2nd grain from player's resources | Pass |  |
| 045 | Player Class | resources.remove(Tile.Resource.ORE); | build city | remove 1st ore from player's resources | Pass |  |
| 046 | Player Class | resources.remove(Tile.Resource.ORE); | build city | remove 2nd ore from player's resources | Pass |  |
| 047 | Player Class | resources.remove(Tile.Resource.ORE); } | build city | remove 3rd ore from player's resources | Pass |  |
| 048 | Player Class | return true; | ore > 0 && grain > 0 && wool > 0 | has development card resources | Pass |  |
| 049 | Player Class | return false; } | ore <= 0 || grain <= 0 || wool <= 0 | does not have development card resources | Pass |  |
| 050 | Player Class | resources.remove(Tile.Resource.ORE); | buy development card | remove 1 more from player's resources | Pass |  |
| 051 | Player Class | resources.remove(Tile.Resource.GRAIN); | buy development card | remove 1 grain from player's resources | Pass |  |
| 052 | Player Class | resources.remove(Tile.Resource.WOOL); } | buy development card | remove 1 wool from player's resources | Pass |  |
| 053 | Player Class | developmentCards.add(dc); } | buy development card | add 1 development card to player | Pass |  |
| 054 | Player Class | developmentCards.remove(index).playCard(); } | buy development card | remove 1 card from deck | Pass |  |
| 055 | Player Class | roads.add(new Road(this)); | Initialize Game Pieces | 15 Roads | Pass |  |
| 056 | Player Class | settlements.add(new Settlement(this)); | Initialize Game Pieces | 5 Settlements | Pass |  |
| 057 | Player Class | cities.add(new City(this)); } } } | Initialize Game Pieces | 4 Cities | Pass |  |
| 058 | Player Class | return "<html><p style=font-size:"+24\*scale+"px; text-align:center>"+playerColor+" Player<br>Score: "+score+"</p><br><br><p style=font-size:"+12\*scale+"px>Ore: "+ore+"<br>Brick: "+brick+"<br>Lumber: "+lumber+ | Initialize Game Board | Game Board Displays Player Color, Score, Ore, Brick, Lumber, Grain, Wool, Settlements Remaining, Cities Remaining, Roads Remaining, Development Cards, Army Size, Has Largest Army | Pass |  |
| 059 | Player Class | ore += 1; | Player has Ore in Resource List | # of Ore is Displayed | Pass |  |
| 060 | Player Class | brick += 1; | Player has Brick in Resource List | # of Brick is Displayed | Pass |  |
| 061 | Player Class | lumber += 1; | Player has Lumber in Resource List | # of Brick is Displayed | Pass |  |
| 062 | Player Class | grain += 1; | Player has Grain in Resource List | # of Grain is Displayed | Pass |  |
| 063 | Player Class | wool += 1; } } } } | Player has Wool in Resource List | # of Wool is Displayed | Pass |  |
|  | PlayerManager Class | PlayerManager Class |  |  |  |  |
| 001 | PlayerManager Class | getNumberOfPlayers(); | Request # of Players | Get # of Players | Pass |  |
| 002 | PlayerManager Class | return players.get(currentPlayerIndex); } | Request Current Player | Get Current Player | Pass |  |
| 003 | PlayerManager Class | currentPlayerIndex = (currentPlayerIndex + 1)%numberOfPlayers; | Request Next Player | Next Player becomes Current Player | Pass |  |
| 004 | PlayerManager Class | players.add(new Player(PlayerColor.values()[i])); } | 2 Players | 2 - purple, orange | Pass |  |
| 005 | PlayerManager Class | players.add(new Player(PlayerColor.values()[i])); } | 3 Players | 3 - purple, orange, green | Pass |  |
| 006 | PlayerManager Class | players.add(new Player(PlayerColor.values()[i])); } | 4 Players | 4 - purple, orange, green, blue | Pass |  |
| 007 | PlayerManager Class | return players; } | Request Listed Players | Return Players | Pass |  |
| 008 | PlayerManager Class | possibleNumbersOfPlayers.add(i); | Game Start | Possible # of Players set to 4 | Pass |  |
| 009 | PlayerManager Class | numberOfPlayers = Game.showSelectionBox(null, "Select Number Of Players", possibleNumbersOfPlayers); } | Game Start | Option to select number of players, 2, 3, or 4 | Pass |  |
| 010 | PlayerManager Class | player.updateResourcesInfo(); | Player has > 7 resources | Update Resources Counters | Pass |  |
| 011 | PlayerManager Class | Tile.Resource resourceChoice = Game.showSelectionBox(null, player.getInformation(scale)+"<br><br> Choose Resource to Remove!", player.getResources()); | If > 7 resource | Select Resource to Remove | Pass |  |
| 012 | PlayerManager Class | player.getResources().remove(resourceChoice); } } } | Remove Resource | Selected Resource Counter -1 | Pass |  |
| 013 | PlayerManager Class | JOptionPane.showMessageDialog(null, getCurrentPlayer().toString()+" PLAYER WINS!!", "Setters of Catan", JOptionPane.INFORMATION\_MESSAGE); } | Player has > 9 Victory Points | Popup Message - Player Wins | Pass |  |
| 014 | PlayerManager Class | p.setLargestArmyFalse(); | player 1 has 3, player 2 has 4 | player 2 now has largest army | Pass |  |
| 015 | PlayerManager Class | return; } } | player 1 has 4, player 2 has 3 | player 1 still has largest army | Pass |  |
| 016 | PlayerManager Class | PURPLE(new Color(250, 50, 250)), ORANGE(new Color(250, 150, 50)), GREEN(new Color(0, 255, 65)), BLUE(new Color(0, 150, 250)); | Purple, Orange, Green, Blue | Color(250, 50, 250), Color(250, 150, 50), Color(0, 255, 65), Color(0, 150, 250) | Pass |  |
| 017 | PlayerManager Class | public Color getColor() { | Player Color = Purple, Orange, Green, Blue | Returns Color Purple, Orange, Green, or Blue | Pass |  |
|  | Road Class | Road Class |  |  |  |  |
| 001 | Road Class | image = ImageIO.read(getClass().getResourceAsStream("/images/purpleRoad.png")); | purple's turn, road placed | purple road placed | Pass |  |
| 002 | Road Class | image = ImageIO.read(getClass().getResourceAsStream("/images/orangeRoad.png")); | orange's turn, road placed | orange road placed | Pass |  |
| 003 | Road Class | image = ImageIO.read(getClass().getResourceAsStream("/images/greenRoad.png")); | green's turn, road placed | green road placed | Pass |  |
| 004 | Road Class | image = ImageIO.read(getClass().getResourceAsStream("/images/blueRoad.png")); } } | blue's turn, road placed | blue road placed | Pass |  |
| 005 | Road Class | JOptionPane.showMessageDialog(null, "Can not create image.", "Settler's of Catan", JOptionPane.ERROR\_MESSAGE); } } | image not found | popup image error | Pass | "Can Not Create Image" 15 times. Then lets you play the game without the specific road |
| 006 | Road Class | player.addRoad(this); } | Add Road | Road Added to Player | Pass |  |
| 007 | Road Class | setLocation(location.getPoint()); | Place Road | Road Location is Set | Pass |  |
| 008 | Road Class | location.setGamePiece(this); | Place Road | Game Piece is set for this Location | Pass |  |
| 009 | Road Class | player.removeRoadResources(); } | Place Road | Road Cost is Removed from Player's Resources | Pass |  |
| 010 | Road Class | gx2D.rotate(-Math.PI/3, getWidth()/2, getHeight()/2); | Road Placed | Rotate Roads -60 Degrees | Pass |  |
| 011 | Road Class | gx2D.rotate(Math.PI/3, 50, 50); | Road Placed | Rotate Roads 60 Degrees | Pass |  |
|  | RoadBuildingCard Class | RoadBuildingCard Class |  |  |  |  |
| 001 | RoadBuildingCard Class | gui.hideButtons(); | Play roadbuilding card | Buttons hidden | Pass |  |
| 002 | RoadBuildingCard Class | board.displayRoadLocations(player); | Play roadbuilding card | Road locations displayed | Pass |  |
| 003 | RoadBuildingCard Class | player.addResource(Tile.Resource.BRICK); | Play roadbuilding card | Add 1 Brick to Player | Pass |  |
| 004 | RoadBuildingCard Class | player.addResource(Tile.Resource.LUMBER); | Play roadbuilding card | Add 1 Lumber to Player | Pass |  |
| 005 | RoadBuildingCard Class | board.setGamePieceReadyToPlace(player.getRoad()); | Play roadbuilding card | Game Piece Ready To Place is set to Road, Player Gets Road | Pass |  |
| 006 | RoadBuildingCard Class | gui.restoreButtons(); | Road Location Selected | Buttons Shown | Pass |  |
| 007 | RoadBuildingCard Class | gui.getParent().repaint(); } | Road Location Selected | Road Placed | Pass |  |
| 008 | RoadBuildingCard Class | board.displayRoadLocations(player); | Played roadbuilding card, placed 1st road | Road Locations Displayed | Pass |  |
| 009 | RoadBuildingCard Class | player.addResource(Tile.Resource.BRICK); | Played roadbuilding card, placed 1st road | Add 1 Brick to Player | Pass |  |
| 010 | RoadBuildingCard Class | player.addResource(Tile.Resource.LUMBER); | Played roadbuilding card, placed 1st road | Add 1 Lumber to Player | Pass |  |
| 011 | RoadBuildingCard Class | board.setGamePieceReadyToPlace(player.getRoad()); } } } } | Played roadbuilding card, placed 1st road | Game Piece Ready To Place is set to Road, Player Gets a 2nd Road | Pass |  |
|  | Robber Class | Robber Class |  |  |  |  |
| 001 | Robber Class | image = ImageIO.read(getClass().getResourceAsStream("/images/robber.png")); } | Robber Image File | Robber displayed | Pass |  |
| 002 | Robber Class | JOptionPane.showMessageDialog(null, "Can't create image", "Settlers of Catan", JOptionPane.ERROR\_MESSAGE); } } | Image not Found | popup image error | Pass | "Can't create image" then lets you play the game without a robber token |
| 003 | Robber Class | currentLocation.clearGamePiece(); | New Robber Location | Robber Removed from old location | Pass |  |
| 004 | Robber Class | setLocation(newLocation.getPoint()); | Select Robber Location | New Location is added for Robber Location | Pass |  |
| 005 | Robber Class | newLocation.setGamePiece(this); | Select Robber Location | Robber Moved to new Location | Pass |  |
| 006 | Robber Class | currentLocation = newLocation; } } | Select Robber Location | Current Robber Location is set to the new robber location | Pass |  |
|  | Settlement Class | Settlement Class |  |  |  |  |
| 001 | Settlement Class | image = ImageIO.read(getClass().getResourceAsStream("/images/purpleSettlement.png")); | Purple's Turn, Settlement Placed | Purple Settlement Placed | Pass |  |
| 002 | Settlement Class | image = ImageIO.read(getClass().getResourceAsStream("/images/orangeSettlement.png")); | Orange's Turn, Settlement Placed | Orange Settlement Placed | Pass |  |
| 003 | Settlement Class | image = ImageIO.read(getClass().getResourceAsStream("/images/greenSettlement.png")); | Green's Turn, Settlement Placed | Green Settlement Placed | Pass |  |
| 004 | Settlement Class | image = ImageIO.read(getClass().getResourceAsStream("/images/blueSettlement.png")); } } | Blue's Turn, Settlement Placed | Blue Settlement Placed | Pass |  |
| 005 | Settlement Class | JOptionPane.showMessageDialog(null, "Can not create image.", "Settler's of Catan", JOptionPane.ERROR\_MESSAGE); } } | image not found | popup image error | Pass | "Can not create image" 5 times. Then lets you play without the specific settlement color |
| 006 | Settlement Class | player.addSettlement(this); } | Build Settlement | Settlement added to player | Pass |  |
| 007 | Settlement Class | setLocation(location.getPoint()); | New Settlement Location | New Location is Added for Settlement | Pass |  |
| 008 | Settlement Class | location.setGamePiece(this); | Select Settlement Location | Settlement placed on this location | Pass |  |
| 009 | Settlement Class | player.addPoint(); | Settlement Placed | Add 1 Victory Point | Pass |  |
| 010 | Settlement Class | player.removeSettlementResources(); } | Settlement Placed | Remove Settlement Resources | Pass |  |
| 011 | Settlement Class | player.addResource(resource); } } | Settlement Placed | Add initial resource to player | Pass |  |
|  | Tile Class | Tile Class |  |  |  |  |
| 001 | Tile Class | image = ImageIO.read(getClass().getResourceAsStream("/images/desertTile.png")); } | Gameboard setup | Desert Placed | Pass |  |
| 002 | Tile Class | JOptionPane.showMessageDialog(null, "Can not create Image", "Settlers of Catan", JOptionPane.ERROR\_MESSAGE); } } | image not found | popup image error | Pass | "Can not create image" pop up a few times, then goes to white screen |
| 003 | Tile Class | image = ImageIO.read(getClass().getResourceAsStream("/images/mountainTile.png")); | Gameboard setup | Mountain Tiles placed | Pass |  |
| 004 | Tile Class | image = ImageIO.read(getClass().getResourceAsStream("/images/hillTile.png")); | Gameboard setup | Brick Tiles Placed | Pass |  |
| 005 | Tile Class | image = ImageIO.read(getClass().getResourceAsStream("/images/fieldTile.png")); | Gameboard setup | Grain Tiles Placed | Pass |  |
| 006 | Tile Class | image = ImageIO.read(getClass().getResourceAsStream("/images/forestTile.png")); | Gameboard setup | Lumber Tiles Placed | Pass |  |
| 007 | Tile Class | image = ImageIO.read(getClass().getResourceAsStream("/images/pastureTile.png")); | Gameboard setup | Wool Tiles Placed | Pass |  |
| 008 | Tile Class | JOptionPane.showMessageDialog(null, "Can not create Image", "Settlers of Catan", JOptionPane.ERROR\_MESSAGE); } } | image not found | popup image error | Pass | "Can not create image" pop up a few times, then goes to white screen |
| 009 | Tile Class | location = l; } | Tile Location Passed | Tile Location Set to this new location | Pass |  |
| 010 | Tile Class | return location; } | Request Tile Location | Tile Location Returned | Pass |  |
| 011 | Tile Class | return resource; } | Request Resource | Resource Returned | Pass |  |
| 012 | Tile Class | player.addResource(resource); } | Add Resource to Player | Resource is added to player | Pass |  |
| 013 | Tile Class | numberToken = nt; } | Number Token Passed | Number Token Set to This Number | Pass |  |
| 014 | Tile Class | return numberToken; } | Request Number Token | Number Token Returned | Pass |  |
| 015 | Tile Class | return image.getScaledInstance((int)(image.getWidth(null)\*scale),(int)(image.getHeight(null)\*scale), Image.SCALE\_SMOOTH); } | Scaled game | Scaled Image | Pass | No Scale |
| 016 | Tile Class | return numberToken.getImage(); } | Request Number Token Image | Return Number Token Image | Pass |  |
| 017 | Tile Class | return numberToken.getValue(); } } | Request Number Token Value | Return Number Token Value | Pass |  |
|  | TileList Class | TileList Class |  |  |  |  |
| 001 | TileList Class | Collections.shuffle(this); | # of each tile setup | Shuffled Tiles | Pass |  |
| 002 | TileList Class | addNumberTokens(); | Tiles Shuffled | Number Tokens Setup | Pass |  |
| 003 | TileList Class | tile.setTileLocation(new Location(tileCenters.remove(0))); } | Initialize Gameboard | Tile Centers Defined | Pass |  |
| 004 | TileList Class | add(new Tile(Tile.Resource.ORE)); | Initialize Gameboard | 3 ore tiles | Pass |  |
| 005 | TileList Class | add(new Tile(Tile.Resource.BRICK)); } | Initialize Gameboard | 3 brick tiles | Pass |  |
| 006 | TileList Class | add(new Tile(Tile.Resource.LUMBER)); | Initialize Gameboard | 4 lumber tiles | Pass |  |
| 007 | TileList Class | add(new Tile(Tile.Resource.GRAIN)); | Initialize Gameboard | 4 grain tiles | Pass |  |
| 008 | TileList Class | add(new Tile(Tile.Resource.WOOL)); } | Initialize Gameboard | 4 wool tiles | Pass |  |
| 009 | TileList Class | tile.setNumberToken(numberTokens.remove(0)); } } | Tile has a resource | Place Number Token | Pass |  |
| 010 | TileList Class | tileCenters.add(new Point((int)((308+(172\*j-86\*i))\*scale),(int)((150+150\*i)\*scale))); | Gameboard Setup | Tile Centers Defined for Top Half of Map | Pass |  |
| 011 | TileList Class | tileCenters.add(new Point((int)((308+(172\*j-86\*i))\*scale),(int)((750-150\*i)\*scale))); } } | Gameboard Setup | Tile Centers Defined for Bottom Half of Map | Pass |  |
|  | VictoryPointCard Class | VictoryPointCard Class |  |  |  |  |
| 001 | VictoryPointCard Class | player.addPoint(); } | Play Victory Card | Add 1 Victory Point | Pass |  |
|  | YearOfPlentyCard Class | YearOfPlentyCard Class |  |  |  |  |
| 001 | YearOfPlentyCard Class | Tile.Resource resource = Game.showSelectionBox(null, "Choose a Resource!", Tile.Resource.values()); | Play Year of Plenty Card | Choice of Resource | Pass |  |
| 002 | YearOfPlentyCard Class | player.addResource(resource); } } | Choice of Resource | Resource added to player | Pass |  |
|  | Board Class | Board Class |  |  |  |  |
| 001 | Board Class | setBackground(java.awt.Color.BLUE); | GUI Setup | Blue Background | Pass |  |
| 002 | Board Class | DISTANCE\_BETWEEN\_SETTLEMENT\_AND\_ADJACENT\_ROAD = 60 \* scale; | GUI Setup, 1.0 Scale | Adjacent Roads are 60px away from Settlements | Pass |  |
| 003 | Board Class | DISTANCE\_BETWEEN\_SETTLEMENTS = 110 \* scale; | GUI Setup, 1.0 Scale | Settlements are 110px away from other Settlement locations | Pass |  |
| 004 | Board Class | DISTANCE\_BETWEEN\_ROADS = 110 \* scale; | GUI Setup, 1.0 Scale | Roads are 110px away from other road locations | Pass |  |
| 005 | Board Class | DISTANCE\_BETWEEN\_TILE\_CENTERS\_AND\_SETTLEMENTS = 110 \* scale; | GUI Setup, 1.0 Scale | Tile centers and Settlements are 110px away | Pass |  |
| 006 | Board Class | placeRobberOnDesert(); | Game Start | Robber is placed on desert tile | Pass |  |
| 007 | Board Class | theRobber.place(tile.getTileLocation()); | Tile = Null (Desert) | Robber is placed on Desert Tile Location | Pass |  |
| 008 | Board Class | add(theRobber); } } | Robber placed on desert tile | Add robber to make it visible for GUI Update | Pass |  |
| 009 | Board Class | LocationButton button = new LocationButton(); | Create new Location Button | Location Button Created | Pass |  |
| 010 | Board Class | button.setButtonLocation(location); | Add Location Button | Location Button Added to Location | Pass |  |
| 011 | Board Class | button.setSize(20, 20); | Add Location Button | Button Size is 20x20 | Pass |  |
| 012 | Board Class | button.setLocation(location.getPoint().x - 10, location.getPoint().y - 10); | Add Location Button | Location Button is put in the center of location | Pass |  |
| 013 | Board Class | buttons.add(button); | Add Location Button | Location Button is added to the buttons list | Pass |  |
| 014 | Board Class | add(button); } | Add Location Button | Location Button is added to the board | Pass |  |
| 015 | Board Class | gamePieceReadyToPlace = theRobber; } | Set Game Piece Ready To Place as Robber | Robber is the Game Piece Ready To Place | Pass |  |
| 016 | Board Class | gamePieceReadyToPlace = gamePiece; } | Set Game Piece Ready To Place as Road | Road is now the Game Piece Ready To Place | Pass |  |
| 017 | Board Class | gamePieceReadyToPlace = gamePiece; } | Set Game Piece Ready To Place as City | City is now the Game Piece Ready To Place | Pass |  |
| 018 | Board Class | gamePieceReadyToPlace = gamePiece; } | Set Game Piece Ready To Place as Settlement | Settlement is now the Game Piece Ready To Place | Pass |  |
| 019 | Board Class | return gamePieceReadyToPlace; } | Request Game Piece Ready To Place | Return Game Piece Ready To Place | Pass |  |
| 020 | Board Class | return true; | Game Piece Ready To Place is Robber | Return TRUE | Pass |  |
| 021 | Board Class | return false; } | Game Piece Ready To Place is not the Robber | Return FALSE | Pass |  |
| 022 | Board Class | remove(button); | Clear Location Buttons | Location Buttons are Removed | Pass |  |
| 023 | Board Class | buttons.clear(); } | Clear Location Buttons | Location Button List is Cleared | Pass |  |
| 024 | Board Class | addButton(tile.getTileLocation()); } | Tile does not have robber | Display Tile Center Button (for Potential Robber Locations) | Pass |  |
| 025 | Board Class | boolean addButton = true; | 1st Setup Road Selected, Location is legal between other roads and settlements | Display Button to Place Game Piece at Location | Pass |  |
| 026 | Board Class | addButton = false; | 2nd Setup Road Selected, Already Road Next To Settlement | Do not Diplay Button to Place Game Piece at Location | Pass |  |
| 027 | Board Class | addButton(roadLocation1); } } | 2nd Setup Road Selected, Location Legal | Location Button added to location | Pass |  |
| 028 | Board Class | addButton(roadLocation2); } | New Road Selected, Empty Location next to player's road | Location button added to potential road location | Pass |  |
| 029 | Board Class | boolean addButton = true; | 1st Settlement Selected | Display Settlement Location Buttons | Pass |  |
| 030 | Board Class | addButton = false; | 2nd Settlement Selected, Location within another settlement | Do not Display Settlement Location Button | Pass |  |
| 031 | Board Class | addButton(settlementLocation); } } | Settlement Locations Valid | Display Settlement Location Buttons | Pass |  |
| 032 | Board Class | boolean addButton = true; | Location has a road and is within legal distance | Show Settlement Location | Pass |  |
| 033 | Board Class | addButton = false; | Settlement Selected, Location is Adjacent to another Settlement | Do Not Show Settlement Location | Pass |  |
| 034 | Board Class | addButton(settlementLocation1); } } | Settlement Selected, Location is adjacent to player owned road, is not adjacent to another settlement, and is currently empty | Add Location Button to Settlement Location | Pass |  |
| 035 | Board Class | addButton(settlementLocation); } } | City Selected, Location has settlement owned by current player | Add Location Button to Settlement Location | Pass |  |
| 036 | Board Class | settlementLocation.getGamePiece().getPlayer().addResource(tile.getResource()); } | Initial Settlement is on Brick, Wool, Grain, Ore, Lumber | Pay Brick, Wool, Grain, Ore, Lumber | Pass |  |
| 037 | Board Class | settlementLocation.getGamePiece().payResourcesToPlayer(tile.getResource()); } | Settlement on tile with currently rolled number token, Does not have Robber | Pay tile's resource | Pass |  |
| 038 | Board Class | possibleTargets.add(settlementLocation.getGamePiece().getPlayer()); | Robber placed on new tile, another player has settlement on same tile | Add that player to the list of potential targets | Pass |  |
| 039 | Board Class | Player target = Game.showSelectionBox(null, "Choose Your Target", possibleTargets); | Robber placed on new tile, another player has settlement on same tile | Popup message to choose a target | Pass |  |
| 040 | Board Class | currentPlayer.addResource(target.getRandomResource()); } } | Target Chosen | Current Player Steals Random Resource from Target | Pass |  |
| 041 | Board Class | Location eventLocation = ((LocationButton)e.getSource()).getButtonLocation(); | Location Selected | Get Location where LocationButton was pressed | Pass |  |
| 042 | Board Class | remove(eventLocation.getGamePiece()); | Place a City | Remove the Previous Settlement from Selected Location | Pass |  |
| 043 | Board Class | gamePieceReadyToPlace.place(eventLocation); | Place a City | Add City to Location | Pass |  |
| 044 | Board Class | add(gamePieceReadyToPlace); | Place Gamepiece | Add GamePiece to Board | Pass |  |
| 045 | Board Class | stealResource(eventLocation); | Place Robber in new location | Popup message to choose a target and steal resource | Pass |  |
| 046 | Board Class | gamePieceReadyToPlace = null; | Placed Gamepiece | Remove from current gamepiece | Pass |  |
| 047 | Board Class | ((Game)SwingUtilities.getWindowAncestor(this)).getGUI().updatePlayerInfo(); | Placed Gamepiece | Update Player Info | Pass |  |
| 048 | Board Class | ((Game)SwingUtilities.getWindowAncestor(this)).repaint(); } | Placed Gamepiece | Update Graphics | Pass |  |
| 049 | Board Class | gx2D.drawImage(tileImage, point.x-tileImage.getWidth(null)/2, point.y-tileImage.getHeight(null)/2,null); | GUI Setup | Tile Image Visible | Pass |  |
| 050 | Board Class | gx2D.drawImage(tokenImage, point.x-tokenImage.getWidth(null)/2, point.y-tokenImage.getHeight(null)/2,null); } } } } | GUI Setup | Number Tokens Visible | Pass |  |

# Known Bugs

Bugs and issues are common with any software application. These can be found during testing, or later in the software lifecycle. Any critical requirement of the system that fails testing will be marked as a bug that requires redesign. If an aspect of the requirements is deemed unnecessary, it can be dispositioned as in-scope for future releases. This is because the value of including a requirement can sometimes be less than the value of delivering and on-time product. Below is a list of all known bugs and issues that are in-scope for future versions:

* City Button Visibility: The button to add cities is underneath the settlements by default, and become visible only upon hovering the mouse over. This is an ease-of-use bug that will be improved, but does not normally have a large negative impact the game.
* Trade Cancelling by Window Closing: This issue involves the trading aspect of the game. When a trade is initiated, it can be called off if the player wishes. Exiting out of the trading screen should logically end the trade, but instead moves on to the next aspect of the trading. This means a trade continues with no resources. This does not negatively impact gameplay and is a user experience feature that will be addressed in future versions.
* R8.2.0 failure: players are not able to trade any 4 resources for a single resource of their choice, which is a minor feature of the original Settlers of Catan game. This is in scope for future releases.
* R8.3.0 failure: Players cannot trade a harbor resource. Harbors are an aspect of the original Settlers of Catan game, but were not able to be implemented in the initial release due to time constraints. This will be featured in a future release.
* R8.4.0 failure: Players cannot trade 3 resources for a single resource of their choice if they have a generic harbor. This is a function of the original Settlers of Catan game, but has not been implemented due to the decision to make harbors in scope for future versions.
* R9.3.0 failure: This is the requirement that involves giving a Victory Point to whichever player has the longest road. This is a feature of the original Settlers of Catan game and is in scope for future releases.
* R9.3.1 failure: A player who builds a road longer than the current longest road holder will gain a victory point, while the old ‘winner’ loses that point. This is not implemented due to the decision to release longest road in future versions.
* The following failures are related to the harbor tiles and are out of scope for current release. They will be implemented in a future version:
  + R1.6.0: There are 9 Harbor Pieces
  + R1.6.1: There is 1 harbor for each resource
  + R1.6.2: There are 4 generic harbors