

# Software Requirements Specification

---

Battleship

Project Team 4

Kaitlin Dosch  
Connor May  
Dominic Ravagnani  
Diane Smith  
Julia VanLandingham

# Table of Contents

<b>Table of Contents</b>	<b>2</b>
<b>Introduction</b>	<b>3</b>
Purpose	3
Intended audience	3
Project Scope	3
References	3
<b>Overall Description</b>	<b>4</b>
Product functions	4
User Characteristics	5
Dependencies	5
<b>Interfaces</b>	<b>6</b>
User interfaces	6
Hardware interfaces	10
Software Interfaces	10
Communications interfaces	10
<b>Functional Requirements</b>	<b>10</b>
GUI Requirements	10
Game Mechanics Requirements	13
Network Requirements	15
<b>Non-Functional Requirements</b>	<b>16</b>
Performance requirements	16
Security requirements	16
Software quality requirements	16

# Introduction

This section gives a scope description and overview of everything included in this SRS document. Also, the purpose of this document is described and a list of abbreviations and definitions is provided.

## Purpose

The purpose of this document is to give a detailed description of the requirements for an online Battleship game software. It will define the purpose and complete declaration for the development of the system. It will also explain the constraints on the system, interfaces, and interactions with external applications.

## Intended audience

This project is completed in partial fulfillment of the requirements for Otterbein University's COMP 3100 course. This has been implemented under the guidance of a university professor. This product is useful for the entertainment of 1-2 people. This document is primarily intended to be proposed for the product's approval before and a reference during the development process.

## Project Scope

The purpose of the online Battleship game is to create a way for a person to play a game of Battleship by themselves or for two people who are unable to play in person (either due to not having the physical game or not being in the same physical location) to play a game of Battleship together via the internet. The system is based on the board game Battleship with all the same rules and objectives taken from nmsu.edu (outlined in the Overall Description section).

## References

“Battleship Board Game Rules: How Do You Play Battleship?” *How Do You Play It*, [howdoyouplayit.com/battleship-board-game-rules-play/](http://howdoyouplayit.com/battleship-board-game-rules-play/).

# Overall Description

This section gives an overview of the whole system. The basic functions of the system as well as how the system interacts with other systems and software.

## Product functions

Our software will be modeling the board game Battleship using the following set of rules (see resources above):

### Game Objective

The object of Battleship is to try and sink all of the other players before they sink all of your ships. All of the other player's ships are somewhere on his/her board. You try and hit them by calling out the coordinates of one of the squares on the board. The other player also tries to hit your ships by calling out coordinates. Neither you nor the other player can see the other's board so you must try to guess where they are. Each board in the physical game has two grids: the lower (horizontal) section for the player's ships and the upper part (vertical during play) for recording the player's guesses.

### Starting a New Game

Each player places the 5 ships somewhere on their board. The ships can only be placed vertically or horizontally. Diagonal placement is not allowed. No part of a ship may hang off the edge of the board. Ships may not overlap with each other.

Once the guessing begins, the players may not move the ships.

The 5 ships are Carrier (occupies 5 spaces), Battleship (4), Cruiser (3), Submarine (3), and Destroyer (2).

### Playing the Game

Player's take turns to guess by calling out the coordinates. The opponent responds with "hit" or "miss" as appropriate. Both players should mark their board with pegs: red for a hit, white for miss. For example, if you call out F6 and your opponent does not have any ship located at F6, your opponent would respond with "miss". You record the miss F6 by placing a white peg on the lower part of your board at F6. Your opponent records the miss by placing it.

When all of the squares that one of your ships occupy have been hit, the ship will be sunk. You should announce "hit and sunk". In the physical game, a red peg is placed on the top edge of the vertical board to indicate a sunk ship.

As soon as all of one player's ships have been sunk, the game ends. The player with the remaining ships is pronounced the winner of the game.

Specifically, our software will be modeling the above game with the ability to play without a physical board using a computer with the options to play against a computer opponent or another user.

## User Characteristics

We expect that the user has basic computer function knowledge and skills as well as some experience using simple GUIs. The user will be able to obtain the required information about their public IP address. We assume that two users that wish to play against each other will be able to communicate outside of this program in order to pass information to obtain network connection.

## Dependencies

Our product requires a device capable of running a java application as well as connecting and communicating with another device over the network.

# Interfaces

This section contains all the interface requirements for the system. This includes a description of the user interface, hardware interface, software interface, and communication interface along with diagrams of what the user interface will look like.

## User interfaces

When the product is first launched a menu screen will appear with the name of the game across the top of the screen and three options for the user to choose: “Play against computer”, “Play against human”, and “Rules”. If the rules option is selected a rules screen will pop up with a summary of the rules and a “back” button. If the “Play against human” option is selected a networking connection screen will be shown with two buttons: “Host” and “Join”. If the “Host” button is selected the host’s IP address will be displayed. If the “Join” button is selected an input screen will be displayed for inputting the host’s IP address.

(menu screen and networking screen)



Networking View

BATTLESHIP - HOST
Your IP Address is:
<input type="text" value="192.168.1.1"/>
Waiting for player to connect

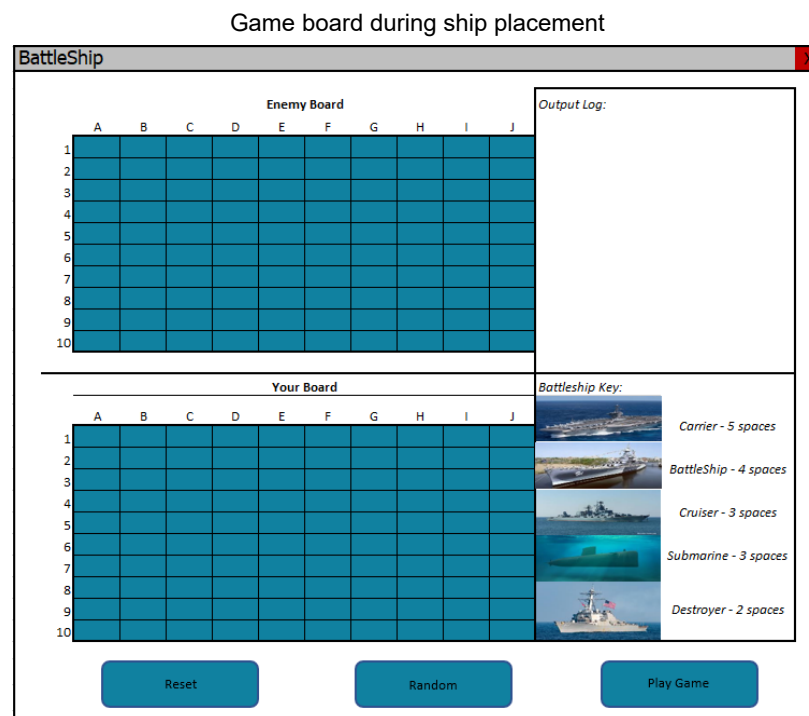
Host view

BATTLESHIP
Enter the hosts IP address:
<input type="text"/>

Player to join

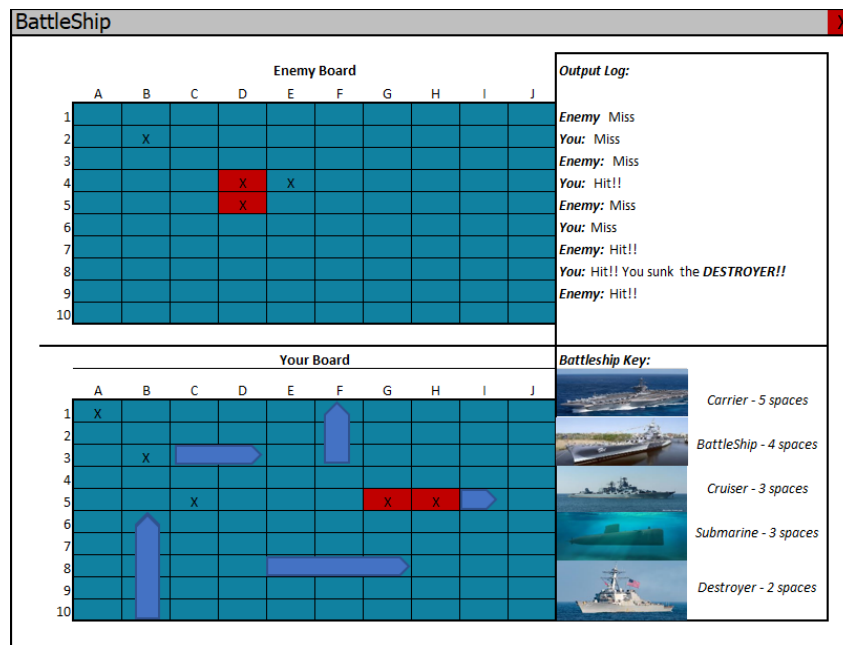
Once the connection has been completed or if the “Play against computer” option was selected then the game screen will appear for the placement of ships. Two game boards will be depicted with one on top of the other along the left-hand side of the screen. The top board depicts the enemy’s board while the bottom is the player’s board. In the top right-hand corner of the screen an output log will display the results of a player’s guessing actions when the game is active. Directly below the output log, there is a bank where ships are stored along with their name and length. The user can select a ship for placement by clicking on a ship. Upon selection, the user will then choose the starting tile for the ship and then the ending tile for the ship. The user can only select tiles on their board at this point.

While the user is placing ships there are three buttons: “Reset”, “Random”, and “Play Game”. The “Reset” button allows the user to return all the ships to the bank and restart the ship placement process. Otherwise, a user cannot move a ship after placement without resetting all ships. The “Random” button allows the user to get a random ship placement and they can continue to select this option as many times as they like. Once all five ships have been placed the “Play Game” button will be able to be pressed and that will start the game.



Once the game has started the buttons for ship placement will disappear but the rest of the screen will look the same. In the ship bank the ships can still be selected however this only makes the informational popup appear. The user can only select tiles on the enemy’s board at this point. When the user selects a tile to drop a bomb onto either a hit or a miss will be registered. If a hit is registered then the tile will turn red with an X and an explosion icon will appear. If a miss is registered the tile will remain blue but will have an X. For all actions, the outcome of the action will be printed to the user’s output log.

Game board during gameplay

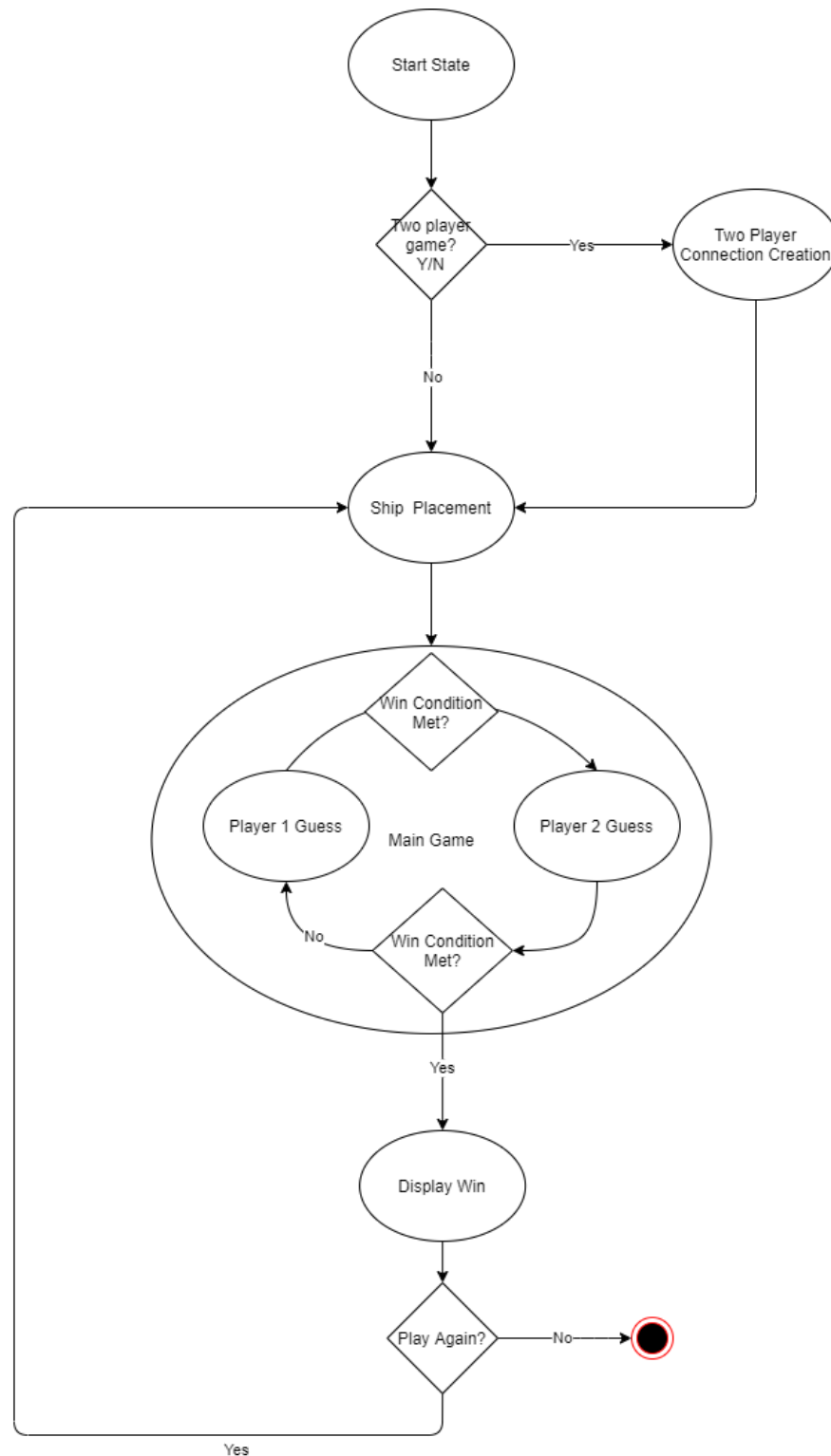


When a player wins “You win” is displayed across their board and “You lose” is displayed for the other player across their board. Two buttons will appear at the bottom of the screen: “Play again” and “Quit”. When the “Quit” button is selected the program will close the program for all users. When the “Play again” option is selected the game will begin again back at the ship placement screen. If there are two players the system will wait until both players have chosen to play again to start a new game.

Game Board - Win view







Activity Diagram

Above is an activity diagram showing different actions and decisions the player can make. This is another representation of the windows we have discussed, showing how the different decisions will lead to different windows and actions.

## Hardware interfaces

There are no specific hardware requirements for this software, though we expect that users would run the software on a windows machine.

## Software Interfaces

There are no specific outside software requirements for this product.

## Communications interfaces

The software will need to communicate through a TCP socket protocol. Our user will need to be able to find their public IP address to be able to use the multiplayer.

# Functional Requirements

This section describes the features required to be implemented. Each requirement will be labeled with a number ID and name. The first number will be the section it is in and the second will be the number of the actual requirement.

## GUI Requirements

### **ID: FR1.1**

Display Start Screen

Description: Display start screen

Dependencies: None

### **ID: FR1.2**

Rules

Description: Display rules screen by user clicking button

Dependencies: FR1.1

### **ID: FR1.3**

Select Opponent

Description: Display options to user to select two player or singleplayer mode

Dependencies: FR1.1

### **ID: FR1.4**

Display network screen

Description: Display network screen for host/join options

Dependencies: FR1.3

**ID: FR1.5**

Display IP address:

Description: Display the host's IP address

Dependencies: FR1.4, FR3.1

**ID: FR1.6**

Input IP address:

Description: Display input screen for IP address input

Dependencies: FR1.4

**ID: FR1.7**

Connect to other user

Description: Connect to other user

Dependencies: FR1.5, FR1.6

**ID: FR1.8**

Display connection status

Description: Display the status of the connection to other user

Dependencies: FR1.7

**ID: FR1.9**

Display ship placement screen

Description: Display the ship placement screen

Dependencies: FR1.3, FR1.7

**ID: FR1.10**

Output text to log

Description: Write text to output log

Dependencies: None

**ID: FR1.11**

Select a Ship to place

Description: User selects which type of ship they would like to place

Dependencies: FR1.9

**ID: FR1.12**

Select a start coordinate

Description: User selects the beginning tile for their ship

Dependencies: FR1.11, FR2.5

**ID: FR1.13**

Select an end coordinate

Description: User selects the ending tile for their ship

Dependencies: FR1.11, FR2.5

**ID: FR1.14**

Display opponent quits game message

Description: Display message that the opponent quit

Dependencies: FR2.16, FR3.3

**ID: FR1.15**

Confirm an individual ship's placement

Description: User confirms a ship's placement before moving on to another ship

Dependencies: FR1.12, FR1.13

**ID: FR1.16**

Display placed ship

Description: Display ship on board after it has been placed

Dependencies: FR1.15

**ID: FR1.17**

Reset ships

Description: Remove all ships from the board

Dependencies: None

**ID: FR1.18**

Display random ship placement

Description: Display a random legal placement of ships

Dependencies: FR2.14

**ID: FR1.19**

Confirm final ship placements

Description: User confirms the final placement of all ships

Dependencies: FR1.15

**ID: FR1.20**

Game screen

Description: Display game screen

Dependencies: FR1.19

**ID: FR1.21**

Display hit or miss

Description: Update the appropriate tile to display the hit or miss

Dependencies: FR2.7

**ID: FR1.22**

Output result of guess in log

Description: Update all results of the user's guess in the log

Dependencies: FR1.10, FR2.7

**ID: FR1.23**

Display Win/Lose Screen

Description: Display win or lose screen appropriately.

Dependencies: FR2.13

**ID: FR1.24**

Wait for both players play again

Description: Wait for both players to select "Play again" at the end of a game in two player mode

Dependencies: None

## Game Mechanics Requirements

**ID: FR2.1**

Initialize an AI opponent

Description: Initialize a computer generated opponent

Dependencies: None

**ID: FR2.2**

Track player turns

Description: Track which player's turn it is

Dependencies: None

**ID: FR2.3**

Start ship placement phase

Description: Start the ship placement process

Dependencies: None

**ID: FR2.4**

Generate random ship placement

Description: Generate a random legal placement of all ships

Dependencies: FR2.10

**ID: FR2.5**

Check if a ship's placement is legal

Description: Verify that a ship's placement is legal

Dependencies: FR2.10

**ID: FR2.6**

Clear stored ships

Description: Reset all ships current placements

Dependencies: FR2.10

**ID: FR2.7**

Check for hit/miss

Description: Check if the guess was a hit or a miss

Dependencies: FR2.10

**ID: FR2.8**

Disable game board

Description: Disable all the buttons of the game board

Dependencies: None

**ID: FR2.9**

Enable game board

Description: Enable appropriate buttons of the game board

Dependencies: FR2.11

**ID: FR2.10**

Store ship placements

Description: Store the positions of every placed ship

Dependencies: FR1.15

**ID: FR2.11**

Store hits/misses

Description: Store the location of all hits and misses

Dependencies: FR2.7

**ID: FR2.12**

Check for sunk ship

Description: Check if all tiles of a ship have been hit

Dependencies: FR2.10, FR2.11

**ID: FR2.13**

Check if a player has won

Description: If all of one player's ships have been sunk, the other player wins

Dependencies: FR2.12

**ID: FR2.14**

Quit

Description: Quit the program

Dependencies: FR3.3

**ID: FR2.15**

Opponent quits game

Description: Return to start screen

Dependencies: None

## Network Requirements

**ID: FR3.1**

Get IP address

Description: Obtain host IP address

Dependencies: None

**ID: FR3.2**

Connect

Description: Client connect to host

Dependencies: FR3.1

# Non-Functional Requirements

## Performance requirements

All actions should execute almost instantaneously. After all information is input, connection between host and client should occur in less than 0.5 seconds.

## Security requirements

This is a low security application.

## Software quality requirements

The software needs to have consistent format and user experience from window to window. Code should be easily readable and well documented including comments throughout as well as method and class comments.