# Battleships - Design documentation

## Overview

The battleships application is a command line application that supports 1 or 2 players. For the one-player mode, there will be a computer player that the player will be facing. When the either player has shot all of the opposing players boats, they win the game, and if their score is better than the current highscores, the player's name will be entered and the high score will be updated. The game will go on until the user exits the application or until one of the players have won.

## Installation

Very simple. Use your command prompt to navigate to the folder with the "BattleShips.jar" file and enter *java -jar BattleShips.jar* to start the game. All of the game files are packaged into the jar file, but the other files are provided for development purposes.

## Design choices and decisions

### Overall structure

The overall structure of the software is divided using a "model-view-controller" architecture/pattern. I chose this type of separation to evenly distribute the responsibilities of the classes and parts of the software. This architecture also provides a higher modularity and allows the changing of different modules, which increases the reusability of the software components.

The class diagram can be found in the "Design" folder provided.

### Decisions

First of all I decided to do this with a command line interface, as I did not have time to find good graphical representations and implement a better graphical user interface. With the time restrictions in mind, I chose to use a CLI. I also decided to have the game randomize where you put your boats instead of you selecting where to put them. The main reason for this decision is that it is very annoying and its very clunky to do so using a command line interface. By randomizing we can jump into the game directly.

The AI ( computer player) is more A than I. Currently, the computer attempts to shoot any spot it hasn't shot yet with no regards to the previous shot. Due to time constraints I didn't

have time to figure out a good algorithm for the AI, but set up some kind of AI so the player has an option to play vs a computer if the player wishes.

Testing was done mainly with manual testing, but I made unit tests to test single component functions to ensure that the behaviour was predictable.

## Future work

For the future I would like to implement network gameplay, probably using websockets for a start. I would also like to improve the computer player to ensure that the computer player takes previous shots into account and figure out a good algorithm to make the computer more challenging. By doing that, I could also implement different difficulties.

I would like to implement a better graphical interface. By having a good user interface, I could also implement the manual adding of boats, background music and provided a better experience.