Software Requirements Specification (SRS) Document

1. Introduction

- **Purpose:**
  - This document provides a detailed description of the requirements for the Battleship game application.
- **Scope:**
  - The application aims to simulate the classic Battleship game where players place ships on a grid and attempt to sink each other's ships by guessing their locations.

2. Overall Description

- **Product Perspective:**
  - The Battleship state tracking API will serve as a standalone component that can be integrated into larger game implementations or used independently for developing Battleship game logic.
- **Product Functions:**
  - The API will support the following functionalities:
    - Create a game board
    - Add a ship
    - Attack ship

3. Specific Requirements

- Create a board
- Add Ship
- Attack Ship
- There should be state tracker without any backend support
- Unit Testing

3.1 Functional Requirements

The functional requirements are explained below:

3.1.1 Create Board

- **Description:**
  - The API shall provide a function to create a game board.
- **Inputs:**
  - Board Size: An integer value representing the size of the game board.
- **Outputs:**
  - Success/Failure message.
- **Exceptions:**
  - If the board size is zero, the system should throw an exception with an appropriate error message.

- **Dependencies:**
    - IBattleShipService: Interface defining the methods for creating and managing the game board.
- **API Endpoint:** (POST) https://priyankajbattleshipcodingtest.azurewebsites.net/api/CreateBoard/{BoardSize}

3.1.2 Add Ship

- **Description:**
    - The API shall provide a function to add a battleship to the game board.
- **Inputs:**
    - TotalShips: An integer value representing the number of ships to add.
- **Outputs:**
    - Success/Failure message.
- **Exceptions:**
    - If the total number of ships is zero, the system should throw an exception with an appropriate error message.
    - If the total number of ships exceeds the board size, the system should throw an exception with an appropriate error message.
- **Dependencies:**
    - IBattleShipService: Interface defining the methods for adding and managing ships on the game board.
- **API Endpoint:** (POST) https://priyankajbattleshipcodingtest.azurewebsites.net/api/CreateShip/{TotalShips}

3.1.3 Attack Ship

- **Description:**
    - Users should be able to attack a specific position on the game board.
- **Inputs:**
    - Position: A string representing the position on the game board to attack.
- **Outputs:**
    - Success/Failure message.
- **Exceptions:**
    - If the specified position is invalid, the system should throw an exception with an appropriate error message.
- **Dependencies:**
    - IBattleShipService: Interface defining the method for attacking ships on the game board.
- **API Endpoint:** (POST) https://priyankajbattleshipcodingtest.azurewebsites.net/api/AttackShip/{Position}

3.1.4 Restart Game

- **Description:**
    - Users should be able to restart the game and reset the game board.
- **Inputs:**
    - None
- **Outputs:**
    - Success message.
- **Dependencies:**

    - IBattleShipService: Interface defining the method for restarting the game.
- **API Endpoint:** (GET) https://priyankajbattleshipcodingtest.azurewebsites.net/api/RestartGame

3.1.5 Display Board

- **Description:**
    - Users should be able to view the current state of the game board.
- **Inputs:**
    - None
- **Outputs:**
    - Matrix representation of the game board.
- **Dependencies:**
    - IBattleShipService: Interface defining the method for retrieving the game board state.
- **API Endpoint:** (GET) https://priyankajbattleshipcodingtest.azurewebsites.net/api/DisplayBoard

3.2 Non-Functional Requirements

- **Performance:**
    - The application should respond to user actions within a reasonable time frame to provide a smooth gaming experience.
- **Reliability:**
    - The application should handle errors gracefully and provide informative error messages to users.
- **Security:**
    - The application should implement appropriate security measures to protect user data and prevent unauthorized access.
- **Usability:**
    - The application should have an intuitive user interface and provide clear instructions for users to interact with the game.

4. Conclusion

- The Battleship state tracking API provides essential functionality for managing game state in a Battleship game for a single player. By adhering to the specified requirements, developers can seamlessly integrate this API into their game implementations, enabling players to enjoy the classic Battleship experience with accurate state tracking and attack resolution.