Software Requirements Specification (SRS)

1. Introduction

Purpose:

The purpose of this Software Requirements Specification (SRS) document is to describe the functional and non-functional requirements for the Tic-Tac-Toe game. This document will provide a detailed description of the game, system behavior, and performance requirements to ensure all stakeholders have a clear understanding of the system's capabilities and limitations.

Scope:

This document covers the requirements for the Tic-Tac-Toe game application. The game will be a two-player game or single player vs AI that can be played on a single device. It will include features such as player authentication, game history tracking, and a replay feature.

2. Overall Description

Product Functions:

- User Registration and Login

- Start a New Game

- choose one vs one or one vs AI

- Play Tic-Tac-Toe

- Display Game History

- Replay Previous Games

- Show Game Statistics

Operating Environment:

- The application will run on desktop platforms with Qt framework support (Windows, macOS, Linux) and will be developed using C++.

- The database will be implemented using SQLite.

- Internet connection is not required as the game is a standalone application.

3. Game Rules

Objective:

The objective of Tic-Tac-Toe is to be the first player to form a horizontal, vertical, or diagonal line of three of one's own markers.

Players:

- The game is played by two players.

- Players take turns placing their markers on the board.

- One player uses "X" markers, and the other player uses "O" markers. The game start by "X" as default in our game

Board:

- The game is played on a 3x3 grid.

- The board starts empty at the beginning of the game.

Gameplay

1. Starting the Game:

- The game randomly selects which player goes first.

- The first player places their marker ("X") in one of the nine empty spaces on the grid.

2. Turns:

- Players take turns to place their marker in an empty cell on the grid.

- A player cannot place a marker in a cell that is already occupied.

3. Winning the Game:

- A player wins the game if they successfully place three of their markers in a horizontal, vertical, or diagonal row.

- The game ends immediately when a player achieves this condition, and the winning player is declared.

4. Draw:

- The game ends in a draw if all nine cells are filled and neither player has achieved a line of three markers.

- A message indicating the draw is displayed, and the game is reset for a new round.

Special Conditions

- AI Player:

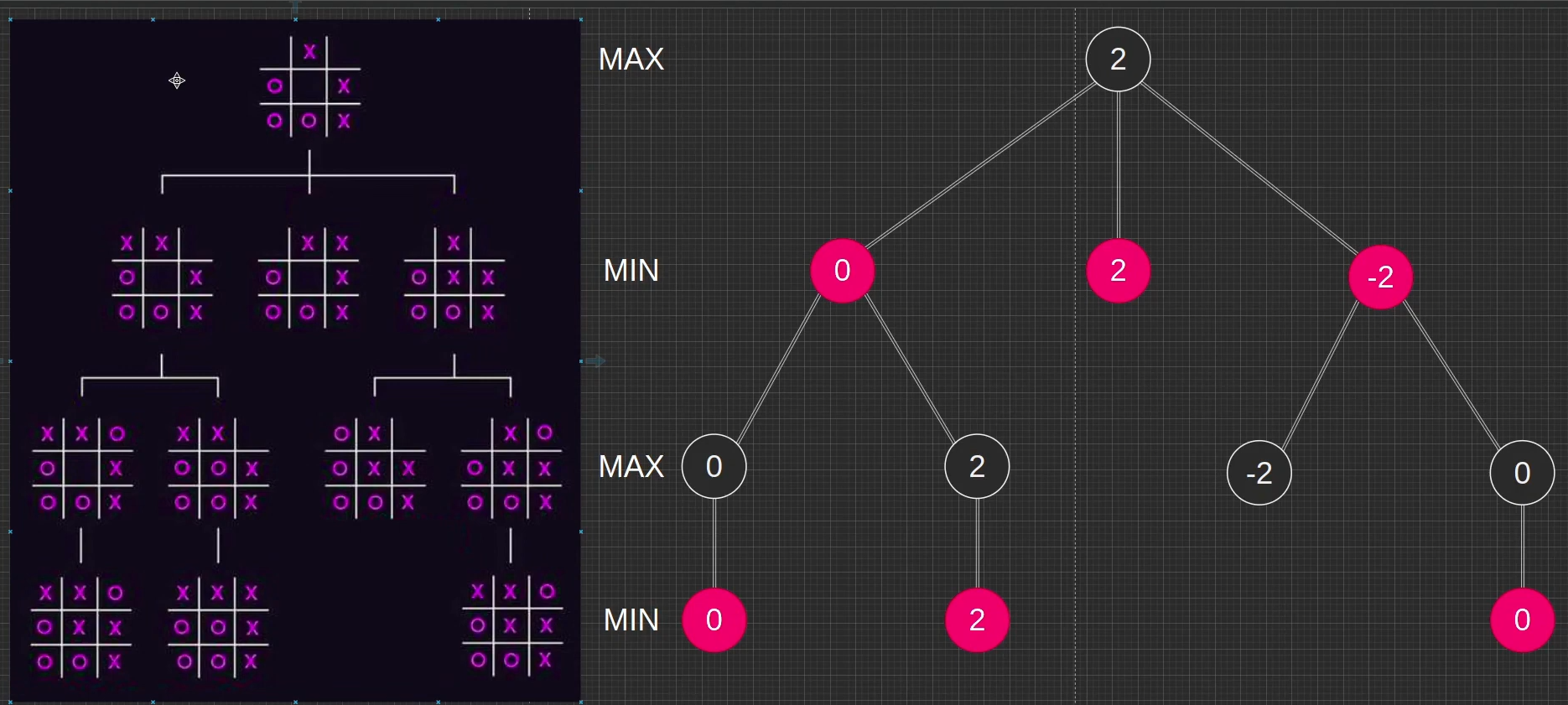
- If playing against the computer (AI), the AI will automatically place markers on its turn.

- The AI's moves are determined by a predefined algorithm designed to challenge the human player.

- The AI always uses the marker "O".

- The AI Player has a special ID of 123456 for identification in the game records.

- we implemented function of AI by Minimax Algorithm



- Game Replay:

- Players can view the history of previously played games.

- The history includes details such as the players involved, the winner, and a replay of the moves made during the game.

- For games played against the AI, the AI's moves are shown as being made by "AI Player".

Game Interface

- The game interface includes a 3x3 grid where players can click to place their markers.

- A message area displays whose turn it is, the winner of the game, or if the game ended in a draw.

Resetting the Game

- A "Reset" button allows players to clear the current game board and start a new game.

- Resetting the game will not affect the game history or player scores.

Starting a New Game

- A "New Game" button starts a fresh game, resetting the board and player turns.

- The new game can be configured to be either player vs. player or player vs. AI.

- Starting a new game will reset the game board and player turns, but will not clear the game history.

Undoing Moves

- An "Undo" button allows players to undo the last move made.

- Only the most recent move can be undone.

- The undo feature can be used to correct mistakes or rethink strategies.

- Undoing a move updates the game board and allows the same player to make a new move.

- The undo functionality is not available once the game has been won or declared a draw.

4. Specific Requirements

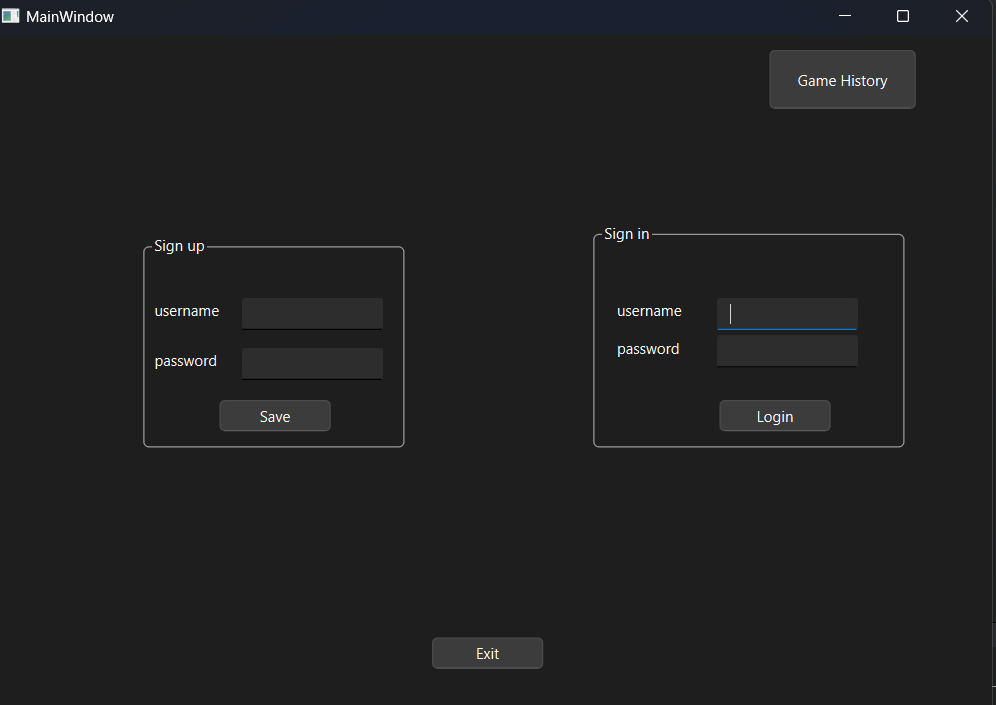
Functional Requirements

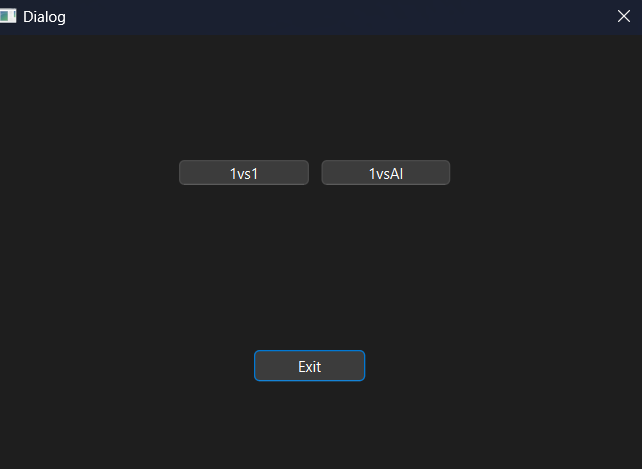
1- User Registration and Login

- Description: Users must be able to register with a username and password if user have account in data you can Login

- Inputs: Username, Password

- Outputs: Registration confirmation, Login confirmation

- Error Handling: Display error message for invalid inputs or existing username.



2- Start a New Game

- Description: Users must be able to start a new game.

- Inputs: Player1 and Player2 selection for

2 mods (one vs one, one vs AI)

- Outputs: Game board initialization

3- Play Tic-Tac-Toe

- Description: Users must be able to play the game following standard Tic-Tac-Toe rules.

- Inputs: User clicks on the game board cells

- Outputs: Updated game board, game status (ongoing, win, draw)

- Error Handling: Prevent clicks on already occupied cells.

4- Display Game History

- Description Users must be able to view a list of past games.

- Inputs: User request for game history you should verify login for User

- Outputs: List of past games with details (players, winner)

5- Replay Previous Games

- Description: Users must be able to replay past games step by step.

- Inputs: User selection of a game to replay

- Output: Sequential display of game steps

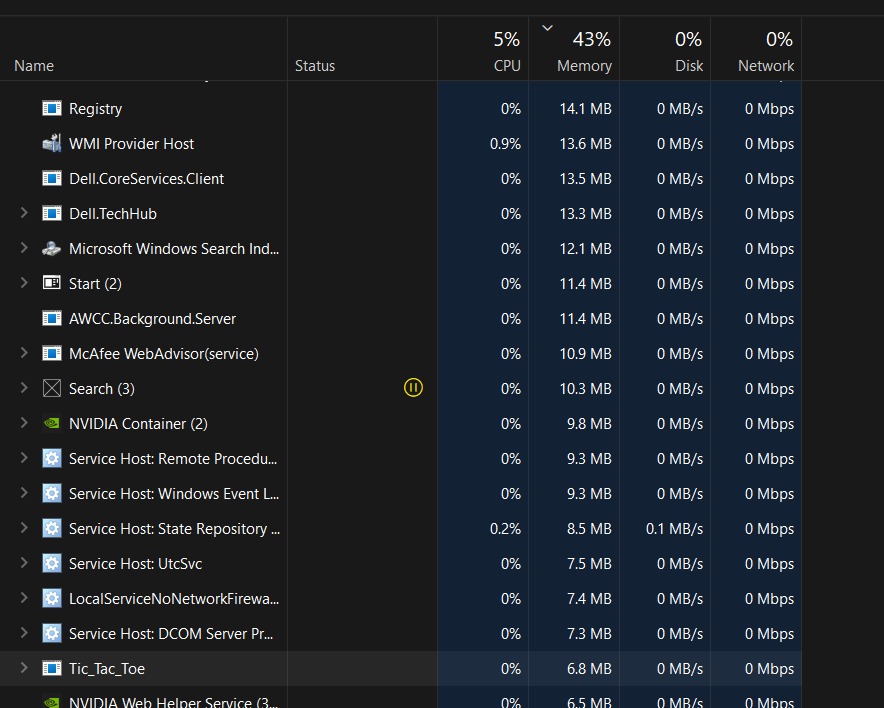
Non-Functional Requirements (Performance Requirements)

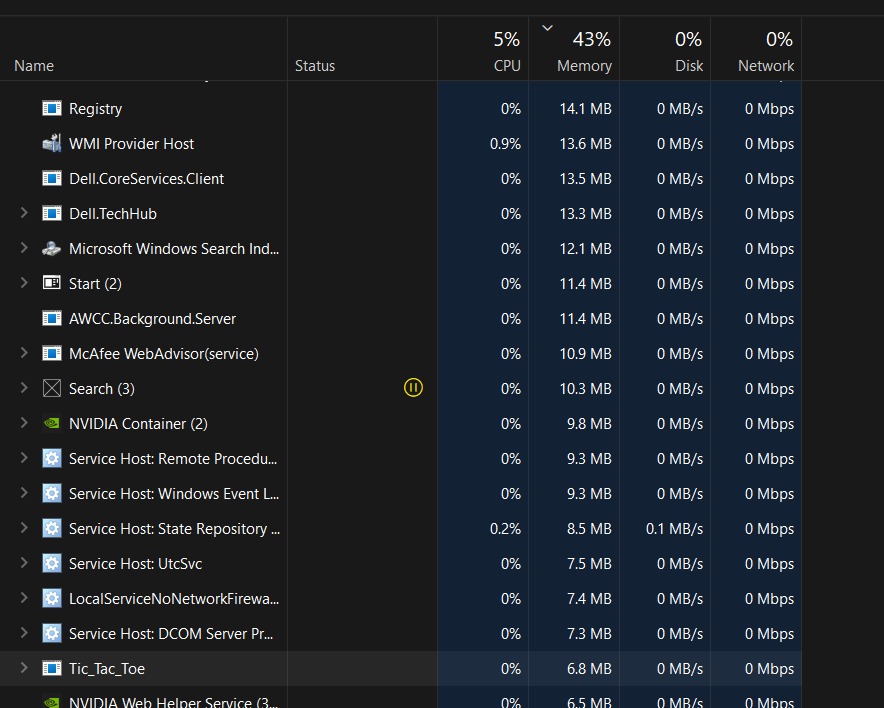
1- Response Time

- for login 🡪 Response time: 1722 milliseconds

- for game history 🡪 Response time: 25 milliseconds

2- Resource Utilization

 - The game should run efficiently on devices with at least the following specifications:



- Storage: 170 KB free disk space

- The application should not consume more than 100 MB of RAM during normal operation.

3- Scalability

- The application should be able to handle game histories for up to 10,000 games without a significant degradation in performance.

- The system should maintain a responsive user interface even as the number of stored games increases.

4- Reliability

- The application should have an uptime of 99.9%, excluding scheduled maintenance.

- Game state should be correctly saved and retrievable without loss of data.

5- Load Handling

- The system should handle up to 100 concurrent users without any noticeable slowdown in performance.

- The application should manage concurrent access to the game history database efficiently to avoid conflicts and ensure data integrity.

6- Error Handling

- The system should handle unexpected errors gracefully, providing meaningful error messages to the user and logging details for further investigation.

- Errors should not cause the application to crash. Instead, the user should be guided to restart the game or return to a stable state.

7- Security

- User data, including game history and personal information, should be securely stored and transmitted using encryption.

- Implement password hashing using a strong cryptographic hash function to securely store and manage user passwords.

- Ensure that the password hashing algorithm is computationally efficient to minimize processing overhead during authentication and registration processes.

8- Data Consistency

- The application should ensure that game states and histories are consistently and accurately saved in the database.

- Changes to game states should be atomic, ensuring that partial updates do not corrupt the data.

9- Compatibility

- The game should be compatible with the following operating systems:

- Windows 7 and later

- The application run on both 32-bit and 64-bit systems.

5. Other Requirements

Database

- The database schema should be well-defined and optimized for performance.

- Tables: Users, Games, GameSteps

User Interface

- The UI should be designed to be responsive and user-friendly.

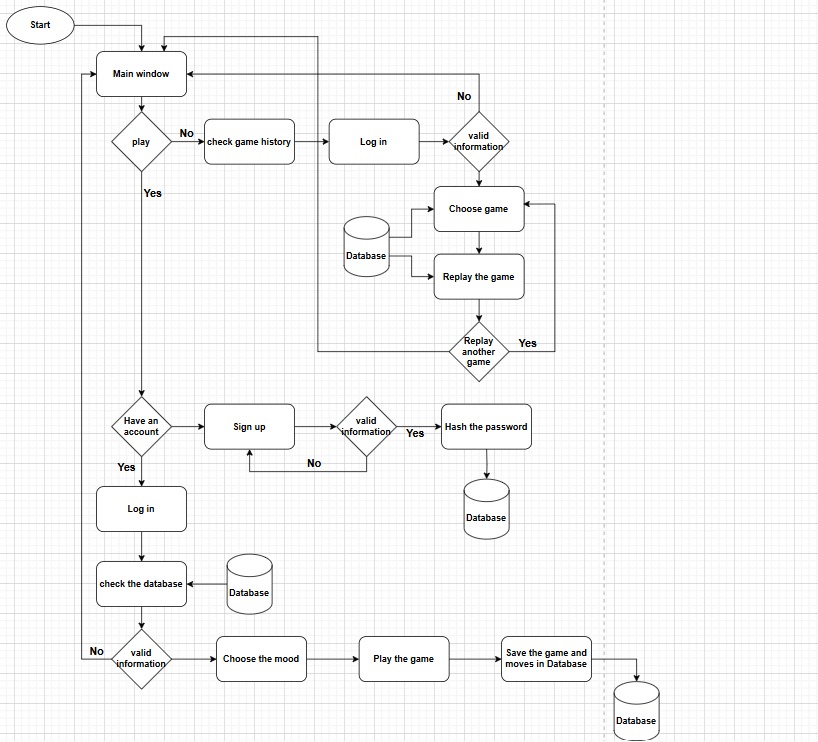
- The main screen should display Sign up and Login and Game History.

- Start Window game for choose any mod you want to play.

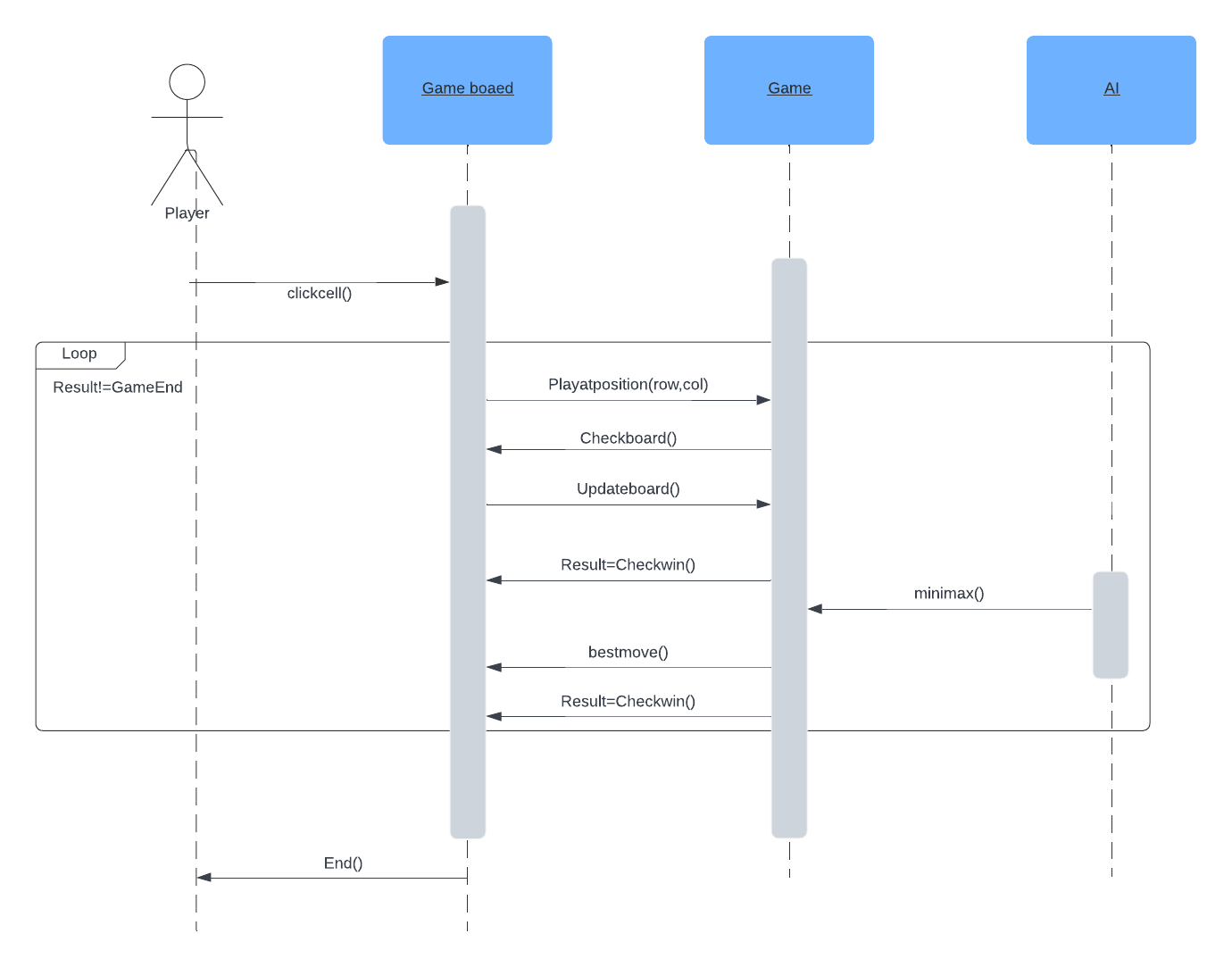
- the game window to show the game board and game controls.

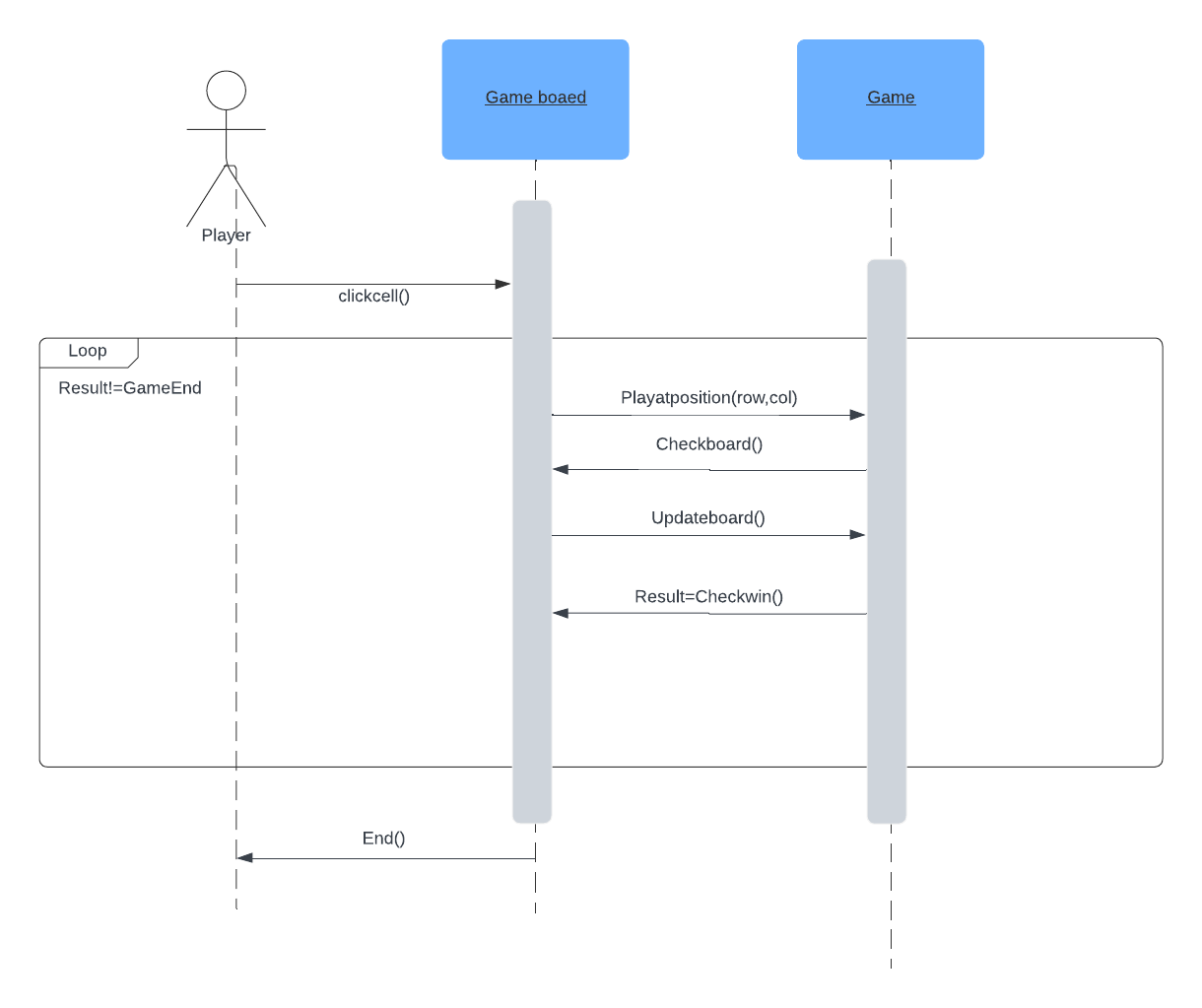
Software Design Specification (SDS):

Flow chart:



Sequence Diagram :

one vs AI:

one vs one: