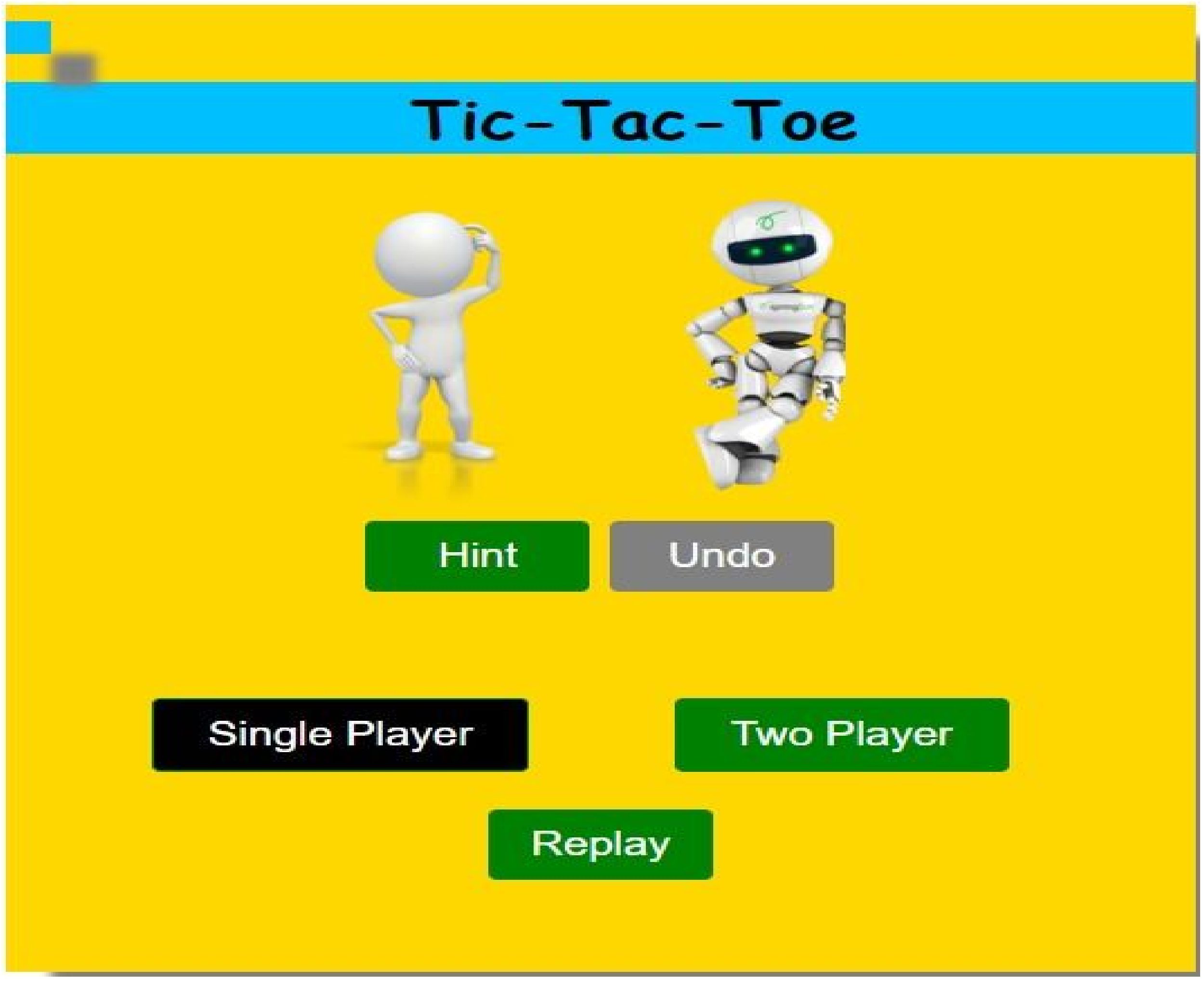


Tic-Tac-Toe

Project 2: **ENTERTAIN THE CREW**

—

Team JSS

JAHANVI PUROHIT DUPPATI SAHITHI SRIJAN MISHRA

INDEX

1.OBJECTIVE

As a part of the “Microsoft Mars-Colonization- program”, we as a team have strived to develop a tic-tac-toe game to entertain the crew.

Tic-tac-toe also called the noughts and crosses or X's and O's is a classic game played against a computer or another player. The players take turns marking the spaces in the 3x3 grid. The first one to finish placing 3 of their marks either in a horizontal or vertical or a diagonal is the winner otherwise the game ends in a draw (a tie).

2.OVERVIEW

2.1 Players

The game, in general, is a two-player game played over a 3 X 3 grid. This involves players of two types:

- A human player
- A computer

The game programmed is so designed to play either against a computer or two human players playing against each other taking alternative turns.

2.2 Game theory

A player can choose between two symbols with his opponent, usual games use “X” and “O”. If the first player chooses “X” then the second player has to play with “O” and vice versa.

A player marks any of the 3 X 3 squares with his symbol (maybe “X” or “O”) and his aim is to create a straight line horizontally or vertically or diagonally with two intentions:

a) Create a straight line before his opponent to win the game.

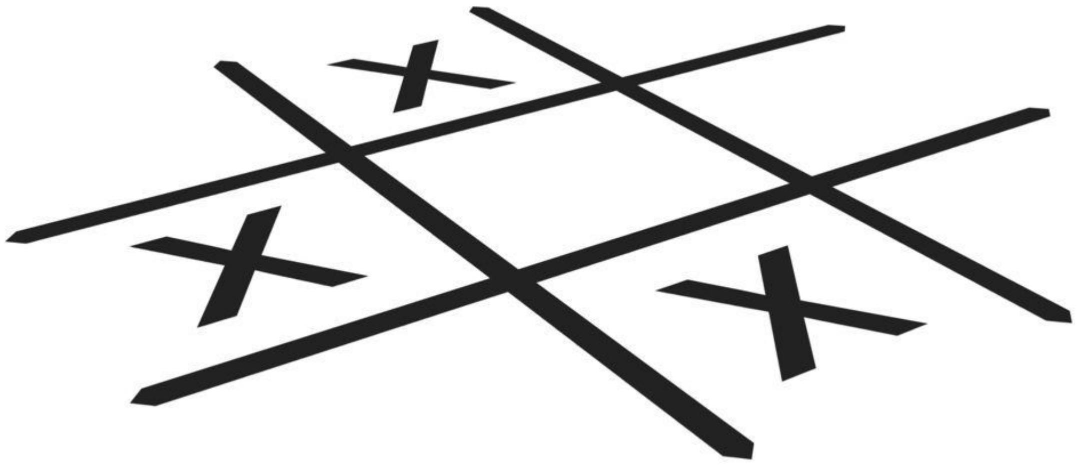
b) Restrict his opponent from creating a straight line first.
In case logically no one can create a straight line with his own symbol, the game results in a tie.

2.3 Possibility of Win

There are only three possible results in a tic-tac-toe game

- A player wins
- His opponent (human or computer) wins
- It's a tie.

3.GAME CODE



The game code is so designed in order to let the players choose either a game against the computer or a game between two. This was so enabled by creating two buttons one of single player and the other of two-player.

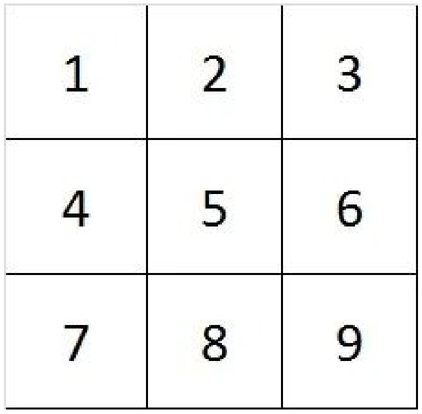
3.1 Languages used

Front end: HTML5 and CSS 3

Back end: Javascript

3.2 Winning Combinations

Name the squares in the 3 X 3 grid from 1 to 9 starting from the top-left corner square horizontally moving towards the right in each row. The figure shown depicts the description above.



1	2	3
4	5	6
7	8	9

If any player is able to draw three Xs or three Os in the following combinations then that player wins. The combinations are listed as follows:

1. 1, 2, 3
2. 4, 5, 6
3. 7, 8, 9
4. 1, 5, 9
5. 3, 5, 7
6. 1, 4, 7
7. 2, 5, 8
8. 3, 6, 9

3.3 Core Logic - The Minimax Algorithm

Minimax is a kind of backtracking algorithm that is used in decision making and game theory to find the optimal move for a player, assuming that your opponent also plays optimally.

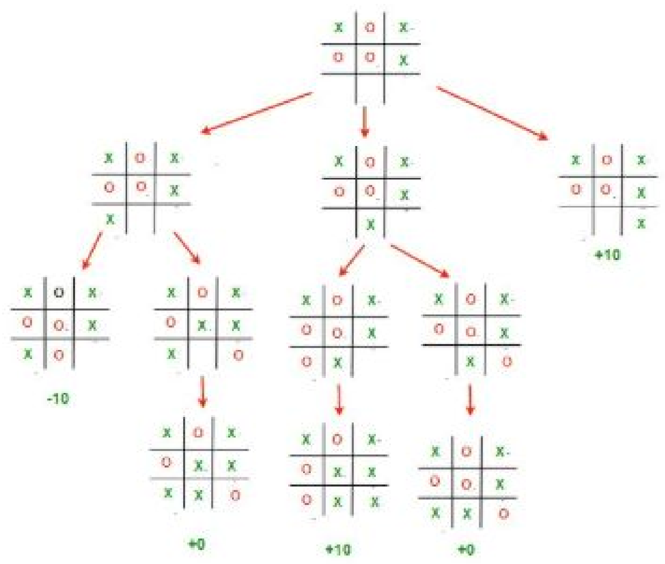
In Minimax the two players are called maximizer and minimizer. The maximizer tries to get the highest score possible while the minimizer tries to do the opposite and get the lowest score possible.

Every board state has a value associated with it. In a given state if the maximizer has upper hand then, the score of the board will tend to be some positive value. If the minimizer has the upper hand in that board state then it will tend to be some negative value. The values of the board are calculated by some heuristics which are unique for every type of game.

We try to assign values to both the players with equal chance. For example, in the code written, we have assigned values of +10 and -10 to player 1 and player 2 respectively.

These values are then assigned to the evaluating function based on the win which stores and declares the winner.

If we represent our board as a 3×3 2D character matrix, like char board[3][3]; then we have to check each row, each column and the diagonals to check if either of the players has gotten 3 in a row.



To develop the levels of depth we have slightly modified the minimax algorithm, like in the hard level we have used the minimax algorithm, and in easy level we have reversed it means the minimizer is also trying to maximise the score, whereas in medium level the computer's first move will be in the middle of the matrix (which will increase the difficulty level slightly) and rest same as the easy level.

3.4 Problems Solved

The program acts as an AI agent to deal with the problems. It uses a functional programming approach to solve the problem where certain functions act as sensors and some react to the environment as actuators.

1. **Game for a Single Player -** One person plays against the computer
2. **Game for Two Players -** Two people can play the game against each other.
3. **Hints -** The system gives a suggestion on what should be the next move of the player who has the upcoming chance.
4. **Undo Moves -** If a player after playing his chance realises that he has some better move then he can undo it and play his chance again.

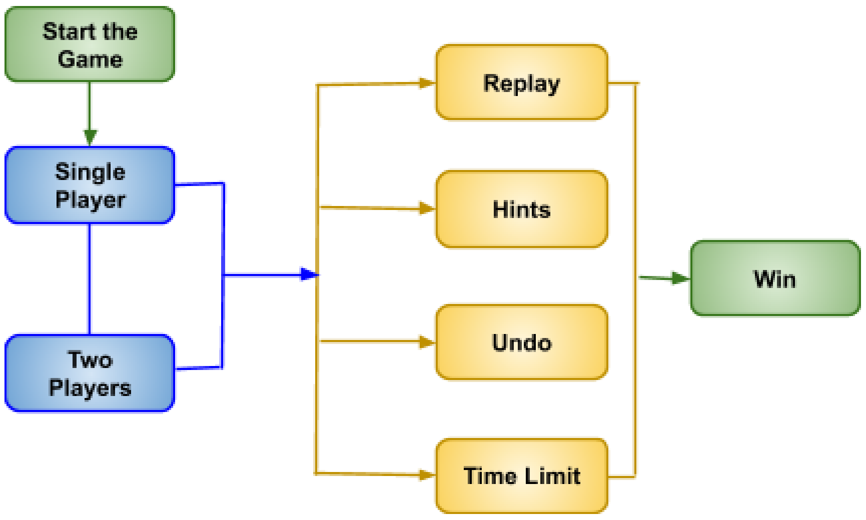
5. **Whose chance next -** This displays who has to play next in the game of two human players.
6. **Timer -** There is a time limit in which every player has to play his move or else he loses the game.
7. **Choice of Starting the Game -** The player does not have any choice to start the game. It's set by default in the two-player game.
8. **Levels of Depth -** How intelligently the computer will play in the single player game.

- ### 3.5 Problems Not Solved

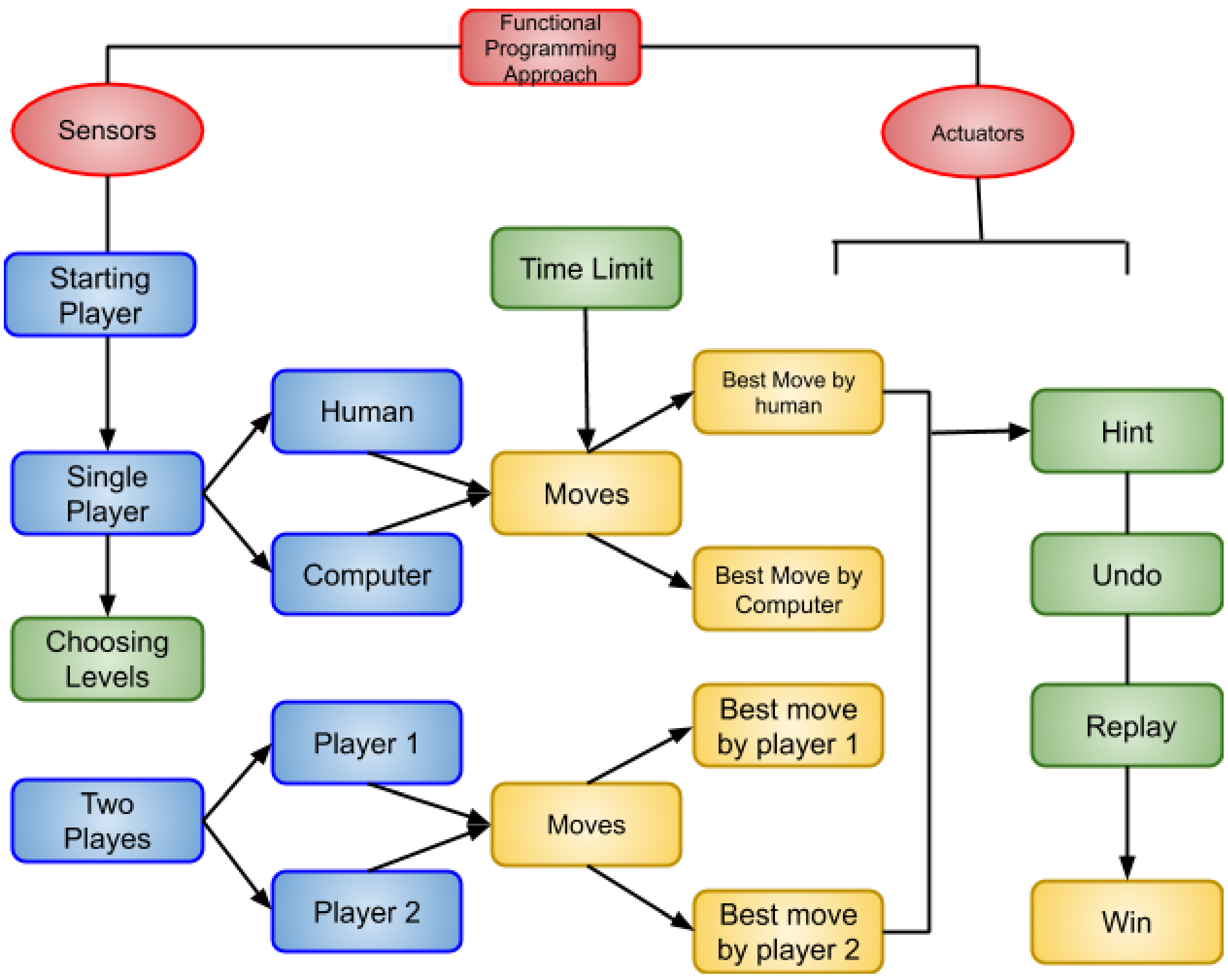
1. **Selection of Symbols -** The symbols i.e O's and X's are assigned to players by the system. They don't have any choice in that.
2. **Login and Logout -** Any anonymous user can play the game.

4. DESIGN DIAGRAMS

4.1 Low-Level Design Diagram

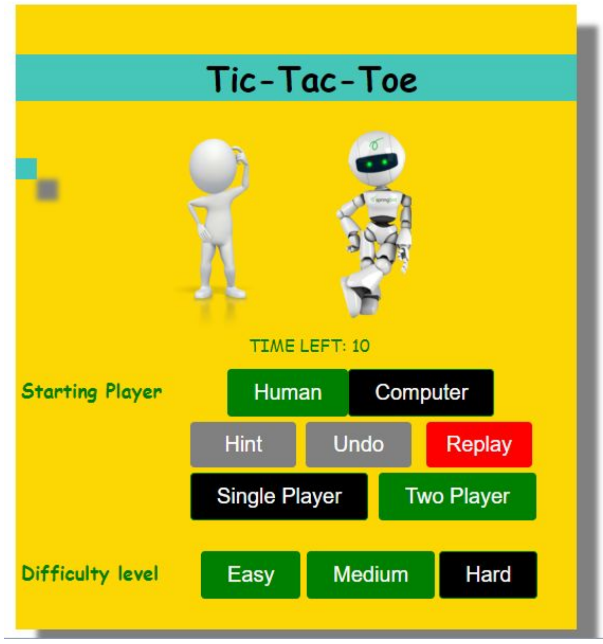


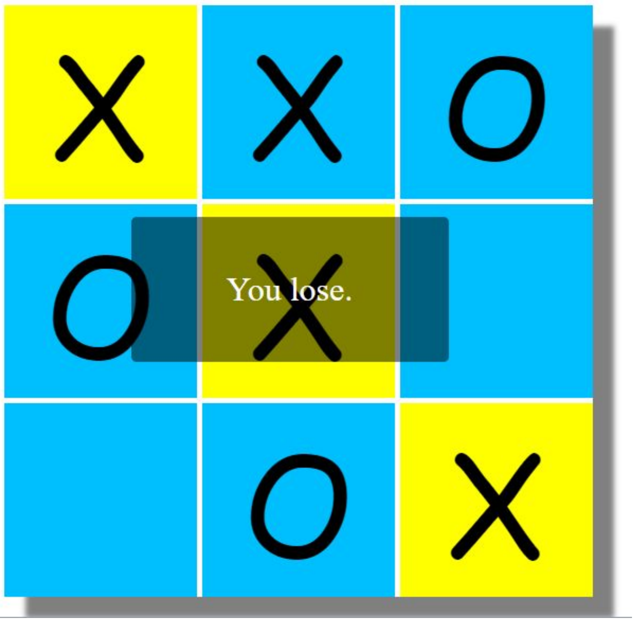
● 4.2 High-Level Design Diagram

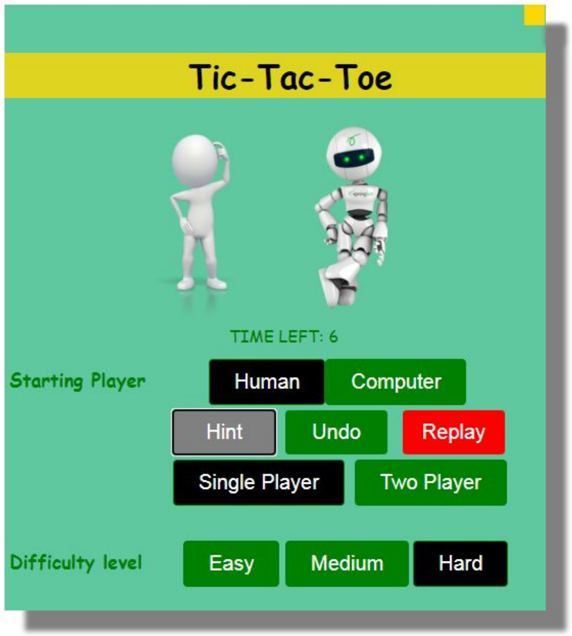


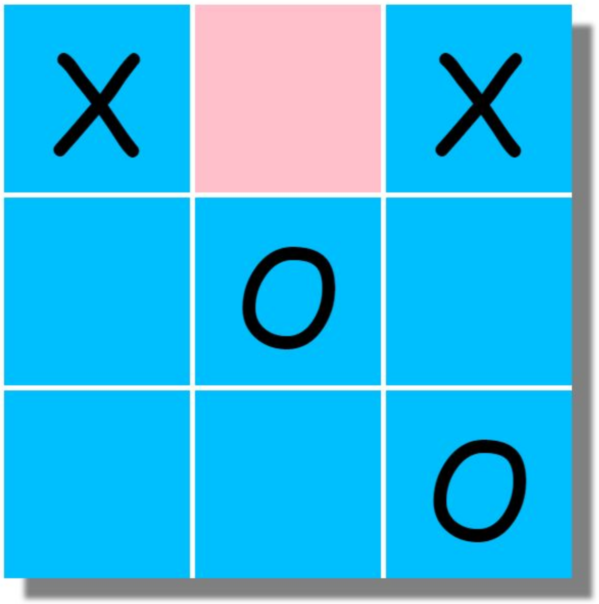
5. SCREENSHOTS FROM THE PROJECT

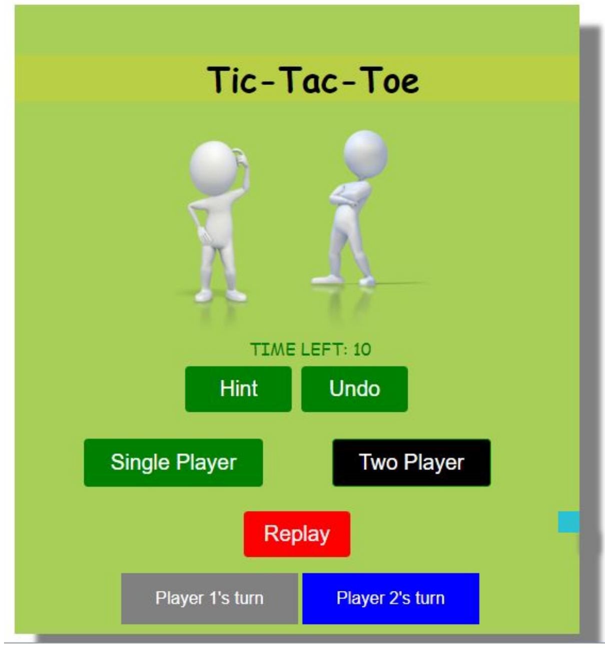
User Interface

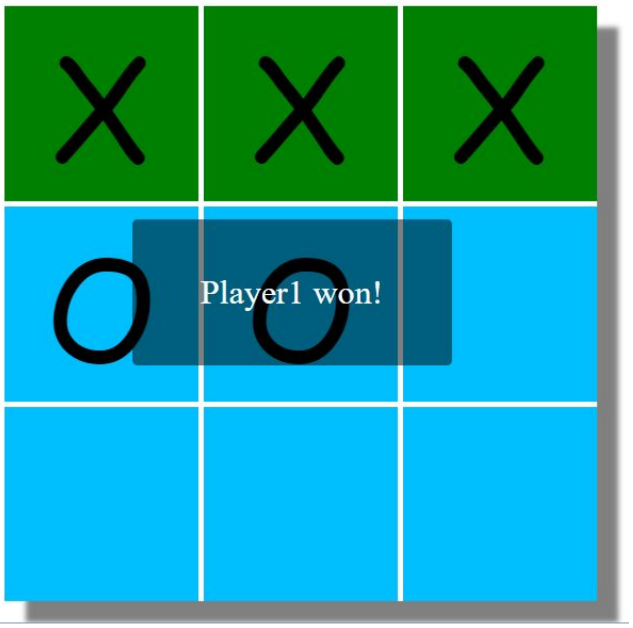

Tic-Tac-Toe
TIME LEFT: 10
Starting Player
Human
Computer
Hint
Undo
Replay
Single Player
Two Player
Difficulty level
Easy
Medium
Hard

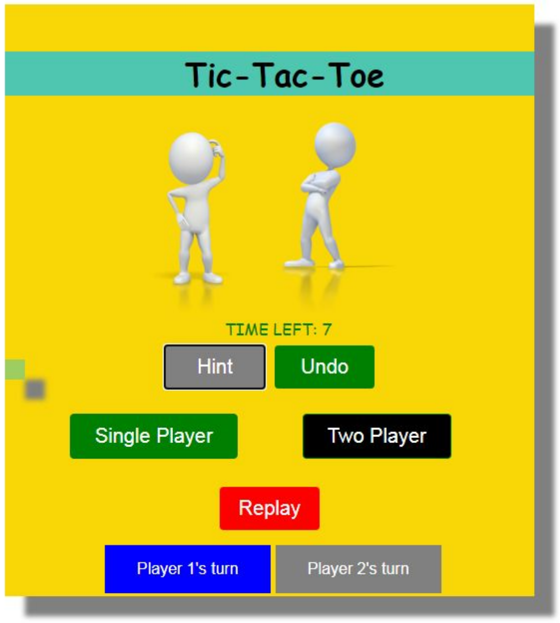

X
X
O
O
X
O
X
You lose.

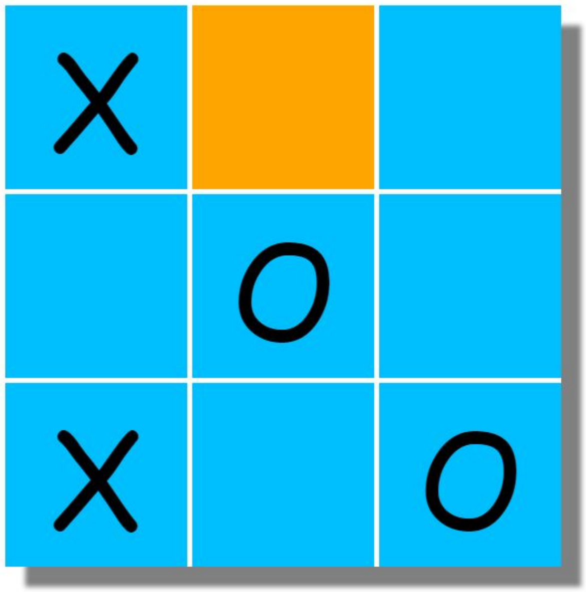

Tic-Tac-Toe
TIME LEFT: 6
Starting Player
Human
Computer
Hint
Undo
Replay
Single Player
Two Player
Difficulty level
Easy
Medium
Hard

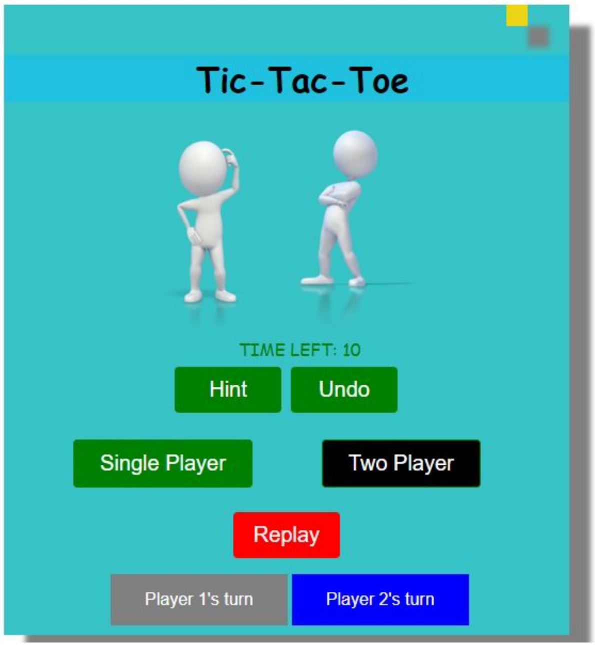

X
X
O
O

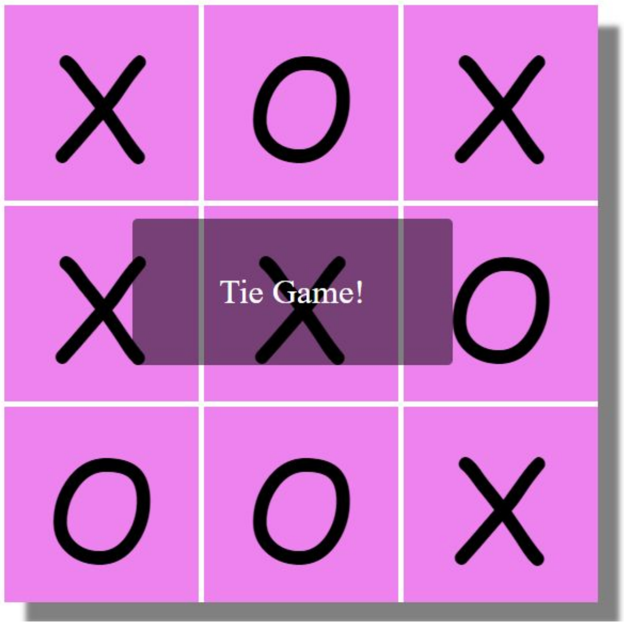

Tic-Tac-Toe
TIME LEFT: 10
Hint
Undo
Single Player
Two Player
Replay
Player 1's turn
Player 2's turn

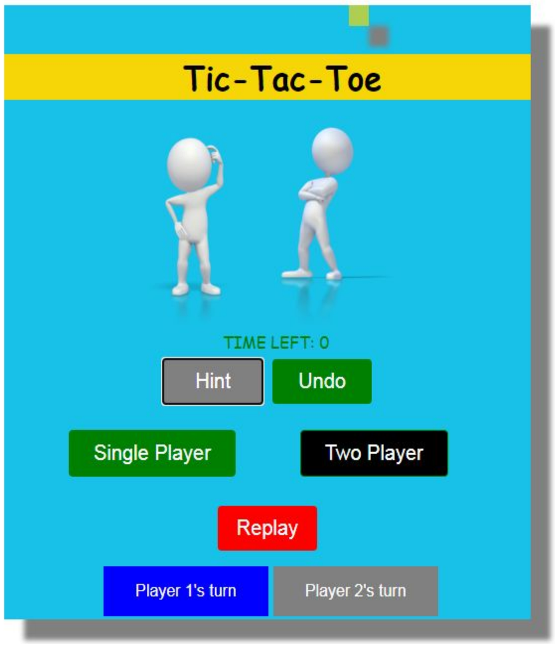

X
X
X
O
O
Player1 won!

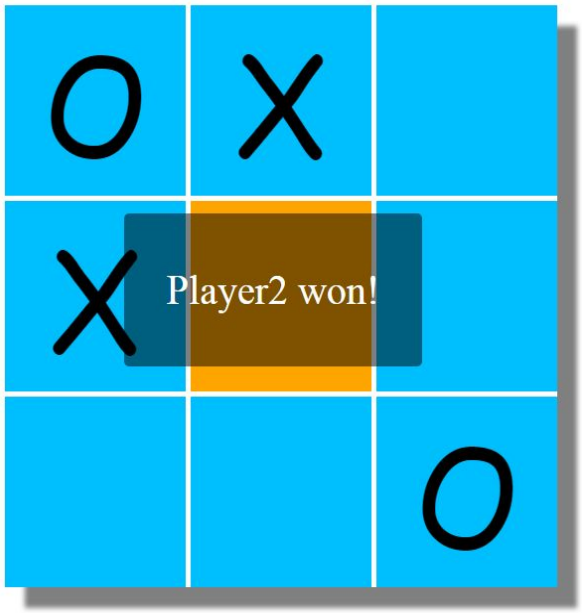

Tic-Tac-Toe
TIME LEFT: 7
Hint
Undo
Single Player
Two Player
Replay
Player 1's turn
Player 2's turn


X
O
X
O


Tic-Tac-Toe
TIME LEFT: 10
Hint
Undo
Single Player
Two Player
Replay
Player 1's turn
Player 2's turn


Tie Game!


Tic-Tac-Toe
TIME LEFT: 0
Hint
Undo
Single Player
Two Player
Replay
Player 1's turn
Player 2's turn


Player2 won!