**[Homework #15 (React Native)](https://bb-csuohio.blackboard.com/webapps/assignment/uploadAssignment?content_id=_5955201_1&course_id=_157272_1&group_id=&mode=view)**

**Shakeeb Rahman 2631309**

Using React Native, design a tic-tac-toe game that you will play against AI. Hence, you will need to figure out how to automatically calculate the next best move for the AI

You will need to keep track of every move (both yours and the AI’s) so that you can relay the game with the click of a button

* This means you will need to have two button (at least): one to play the game, the other to replay the game moves

**Screen/UI:**

Table

Description automatically generated with medium confidence

This is a simple Tic-Tac-Toe game built using React where you can play against an AI opponent. The AI uses the Minimax algorithm to make its moves, which ensures it's always playing the best possible move, making it a tough competitor.

How to run the app

Make sure you have Node.js and NPM (Node Package Manager) installed on your computer. You can download them from the official Node.js website.

Open your terminal (or command prompt) and navigate to the directory containing the React app (where the package.json file is located).

Run the following command to install all necessary dependencies: npm install

The Minimax algorithm works by recursively simulating all possible game states resulting from the current player's moves and their opponent's subsequent moves. It assigns a score to each game state based on the outcome (win, loss, or draw) and chooses the move that leads to the highest score for the current player (while minimizing the score for the opponent).

In the context of this Tic-Tac-Toe game, the AI opponent evaluates all possible moves on the board and chooses the one that will result in the best outcome for itself, while minimizing the chances of the human player winning.

Brief description of the Minimax algorithm

The Minimax algorithm can be summarized in the following steps:

Start from the current game state and generate all possible moves for the player (AI or human).

For each move, evaluate the new game state and assign a score based on the outcome (win, loss, or draw).

If the game state is a terminal state (a win, loss, or draw), return the score.

If the game state is not a terminal state, continue generating and evaluating game states for the other player, recursively applying the same process.

The algorithm will choose the move that leads to the maximum score for the current player, while minimizing the score for the opponent (hence the name Minimax).

By using this algorithm, the AI opponent can predict all possible outcomes of the game and make decisions that ensure the highest probability of winning or drawing.

npm start

Open your browser and go to http://localhost:3000 to view the app.

How the AI works

the Minimax algorithm predicts future moves and chooses the move that results in the most favorable outcome for the AI, while considering the best possible moves for the human player.

**When AI wins, an alert message is shown:**  
Graphical user interface

Description automatically generated

**When it’s a draw:**

Graphical user interface

Description automatically generated