Richard Stretanski
CIS 525
DLN Student

Analysis

I was tasked with creating a page where a user could play a game of tic-tac-toe with either a second player or an AI. The two game modes would be toggled between by the user. I know the assignment said to use radio buttons for this but I thought it would look a lot better if it was one switch that you could toggle between 2 options. So for this I used one checkbox that is styled like a switch. Checked is one option and unchecked is the other. The game would start upon the click of a start button and the first move would be random ('X' or 'O' going first). For this I planned on having one start button that would change into a restart button once the game started. In the mode of Player vs Computer (PvC), the AI would always be 'O' and places pieces at random. Text on the screen would always display who's turn it is. No player would be able to write over a piece that is already on the board. For this I also added a message when someone clicked on the board before clicking the start button. The amount of time that each game has been played for would be recorded and written. Once a game ending condition is met, the game would indicate it and ask if another game should be started in the form of a button. I decided for this that the created button would be the same size as the game grid and partially transparent so the user could see the results but have to click the button to play again.

For the Javascript portion of the project I decided to break it up into 8 functions and a page load listener. I used a global array to keep track of which cells were filled. The first function I created was the function that is executed when the start button is clicked. I decided to make this function do multiple things instead of breaking them up into separate functions. The start button click function cleared the board, the timer, the array, and any added css, it changed its own text to 'Restart' and the button also started the game. It called a randomizer function to decide who goes first and depending on which game mode was selected, if an AI is required to go first, then it will make the first move. I used a global game state variable to keep track of whos turn it is. For the game grid I used a table that passed a different variable value for each cell to the same function when clicked. This function executed the turns of the game and put pieces on the board. When a table cell is clicked, this function puts all the cells into an array and places the appropriate piece into the cell. Since this function is called each time a cell is clicked, I can use it for PvP X, PvP O, and PvC X, and then call the AI function that executes the AI's turn (which is only O turn). The AI turn function is recursive. It calculates a random number, checks to see if that space is open, then places an O in the space if it is, and if it is not open, it calls itself again until it finds an open space. I had another function that checks to see if a game-ending condition has been met and then if it was met, it would call a function to add game winning CSS which included a button that allows the user to play again. I also had a function that handled the timer, which was reset by the start button function.

Use-Case Diagram