

Project Proposal

TEAM NAME –

CODE BLOOD

PORJECT NAME –

Tic Tac Toe Game

TEAM MEMBERS –

Aman	19103013
Rohit Bajaj	19103017
Arzoo Goyal	18103087
Geetika Bansal	18103028
Piyush	

ABOUT PROJECT

TIC TAC TOE is a (one 3X3 board) very popular game. This game is played between two players. The first player makes move with the circle(0) or cross(X). The player, who has first formed a horizontal, vertical or diagonal sequence first is a winner. This game is implemented using the minimax algorithm.

Although this is a simple project but we can make this project big and quite learning by adding different modes of playing (Man vs Computer, Man vs Man, 2-player online game). Still working on this project.

WHAT WE WILL DO IN PROJECT

In this project we will make a game of tic tac toe which will be played between user and computer.

Technologies to be used:

Bootstrap, HTML, CSS and javascript.

Learning Outcomes:

Every team member has learned many new technical skills. They are :

- Python
- HTML + CSS
- Bootstrap
- Javascript

Code Pics:

```

1 <html>
2   <head>
3     <meta charset="UTF-8">
4     <title>Tic Tac Toe</title>
5     <link rel="stylesheet" href="style.css">
6   </head>
7   <body style="background-color: #d9dee2">
8     <h1>Tic Tac Toe Game</h1>
9     <h3>Man Vs Computer</h3>
10    <table>
11      <tr>
12        <td class="cell" id="0"></td>
13        <td class="cell" id="1"></td>
14        <td class="cell" id="2"></td>
15      </tr>
16      <tr>
17        <td class="cell" id="3"></td>
18        <td class="cell" id="4"></td>
19        <td class="cell" id="5"></td>
20      </tr>
21      <tr>
22        <td class="cell" id="6"></td>
23        &td class="cell" id="7"></td>
24        &td class="cell" id="8"></td>
25      </tr>
26    </table>
27
28    <div class="endgame">
29      <div class="text"></div>
30    </div>
31    <button class="btn" style="margin: 80px 830px 0px; font-size: 40px;">
32
33    <script src="script.js"></script>
34  </body>
35</html>

```

```

1 var origBoard;
2 const huPlayer = 'O';
3 const aiPlayer = 'X';
4 const winCompos = [
5   [0, 1, 2],
6   [3, 4, 5],
7   [6, 7, 8],
8   [0, 3, 6],
9   [1, 4, 7],
10  [2, 5, 8],
11  [0, 4, 8],
12  [6, 4, 2]
13 ]
14
15 const cells = document.querySelectorAll('.cell');
16 startGame();
17
18 function startGame() {
19   document.querySelector(".endgame").style.display = "none";
20   origBoard = Array.from(Array(9).keys());
21   for (var i = 0; i < cells.length; i++) {
22     cells[i].innerText = '';
23     cells[i].style.removeProperty('background-color');
24     cells[i].addEventListener('click', turnClick, false);
25   }
26 }
27
28 function turnClick(square) {
29   if (typeof origBoard[square.target.id] == 'number') {
30     turn(square.target.id, huPlayer);
31     if (!checkWin(origBoard, huPlayer) && !checkTie()) turn(bestSpot(
32   )
33 }
34
35 function turn(squareId, player) {
36   origBoard[squareId] = player;
37   document.getElementById(squareId).innerText = player;
38   let gameWon = checkWin(origBoard, player);
39   if (gameWon) gameOver(gameWon);
40 }
41
42 function checkWin(board, player) {
43   let plays = board.reduce((a, e, i) =>
44     (e === player) ? a.concat(i) : a, []);
45   let gameWon = null;
46   for (let [index, win] of winCompos.entries()) {
47     if (win.every(elem => plays.indexOf(elem) > -1)) {

```

```

1 td {
2   border: 2px solid #333;
3   height: 150px;
4   width: 150px;
5   text-align: center;
6   vertical-align: middle;
7   font-family: "arial", cursive, sans serif;
8   font-size: 100px;
9   cursor: pointer;
10 }
11
12 table {
13   border-collapse: collapse;
14   position: absolute;
15   left: 50%;
16   margin-left: -230px;
17   top: 250px;
18 }
19
20 table tr:first-child td {
21   border-top: 0;
22 }
23
24 table tr:last-child td {
25   border-bottom: 0;
26 }
27
28 table tr td:first-child {
29   border-left: 0;
30 }
31
32 table tr td:last-child {
33   border-right: 0;
34 }
35
36 .endgame {
37   display: none;
38   width: 200px;
39   top: 300px;
40   background-color: rgba(205, 133, 63, 0.8);
41   position: absolute;
42   left: 50%;
43   margin-top: 100px;
44   margin-left: -100px;
45   padding-top: 50px;
46   padding-bottom: 50px;
47   text-align: center;
48   color: black;

```

```

47   if (win.every(elem => plays.indexOf(elem) > -1)) {
48     gameWon = {index: index, player: player};
49     break;
50   }
51 }
52 return gameWon;
53
54
55 function gameOver(gameWon) {
56   for (let index of winCompos[gameWon.index]) {
57     document.getElementById(index).style.backgroundColor =
58       gameWon.player == huPlayer ? "blue" : "orange";
59   }
60   for (var i = 0; i < cells.length; i++) {
61     cells[i].removeEventListener('click', turnClick, false);
62   }
63   declareWinner(gameWon.player == huPlayer ? "You win!" : "You lose.");
64 }
65
66
67 function declareWinner(who) {
68   document.querySelector(".endgame").style.display = "block";
69   document.querySelector(".endgame .text").innerText = who;
70 }
71
72 function emptySquares() {
73   return origBoard.filter(s => typeof s == 'number');
74 }
75
76 function bestSpot() {
77   return minimax(origBoard, aiPlayer).index;
78 }
79
80 function checkTie() {
81   if (emptySquares().length == 0) {
82     for (var i = 0; i < cells.length; i++) {
83       cells[i].style.backgroundColor = "green";
84       cells[i].removeEventListener('click', turnClick, false);
85     }
86     declareWinner("Tie Game!")
87     return true;
88   }
89   return false;
90 }
91
92 function minimax(newBoard, player) {
93   var availSpots = emptySquares();

```

```
# style.css > .endgame
24 table tr:last-child td {
25   border-bottom: 0;
26 }
27
28 table tr td:first-child {
29   border-left: 0;
30 }
31
32 table tr td:last-child {
33   border-right: 0;
34 }
35
36 .endgame {
37   display: none;
38   width: 200px;
39   top: 300px;
40   background-color: rgba(205,133,63,0.8);
41   position: absolute;
42   left: 50%;
43   margin-top: 100px;
44   margin-left: -100px;
45   padding-top: 50px;
46   padding-bottom: 50px;
47   text-align: center;
48   color: black;
49   font-size: 2em;
50 }
51
52 h1 {
53   position: absolute;
54   left: 50%;
55   margin-top: 50px;
56   margin-left: -220px;
57   text-align: center;
58   font-size: 3em;
59 }
60
61 h3 {
62   position: absolute;
63   left: 50%;
64   margin-top: 150px;
65   margin-left: -90px;
66   text-align: center;
67 }

JS script.js > checkTie
94
95 if (checkWin(newBoard, huPlayer)) {
96   return {score: -10};
97 } else if (checkWin(newBoard, aiPlayer)) {
98   return {score: 10};
99 } else if (availSpots.length === 0) {
100   return {score: 0};
101 }
102
103 var moves = [];
104 for (var i = 0; i < availSpots.length; i++) {
105   var move = {};
106   move.index = newBoard[availSpots[i]];
107   newBoard[availSpots[i]] = player;
108
109   if (player == aiPlayer) {
110     var result = minimax(newBoard, huPlayer);
111     move.score = result.score;
112   } else {
113     var result = minimax(newBoard, aiPlayer);
114     move.score = result.score;
115   }
116
117   newBoard[availSpots[i]] = move.index;
118   moves.push(move);
119 }
120
121 var bestMove;
122 if (player == aiPlayer) {
123   var bestScore = -10000;
124   for (var i = 0; i < moves.length; i++) {
125     if (moves[i].score > bestScore) {
126       bestScore = moves[i].score;
127       bestMove = i;
128     }
129   }
130 } else {
131   var bestScore = 10000;
132   for (var i = 0; i < moves.length; i++) {
133     if (moves[i].score < bestScore) {
134       bestScore = moves[i].score;
135       bestMove = i;
136     }
137   }
138 }
139
140 return moves[bestMove];
141 }
```

Game Pics:





