Tic tac toe

Nombre:

Tic tac toe

¿Como se realizó?

Primero previsualice lo que quería hacer en un algoritmo u seudocódigo,

Posterior a ello comencé a primeramente realizar el cuerpo en HTML, y luego a ello procedí a darle un diseño un poco minimalista con css Grid, ya por ultimo le di vida al juego utilizando JavaScript.

Código o maquetación utilizado en HTML:

```
<!doctype html>
<html lang="en">

<head>
  <meta charset="UTF-8">
  <meta name="viewport"
    content="width=device-width, user-scalable=no, initial-scale=1.0,
maximum-scale=1.0, minimum-scale=1.0">
  <meta http-equiv="X-UA-Compatible" content="ie=edge">
  <title>Game: Tic Tac Toe</title>
  <link rel="stylesheet" href="style.css">
</head>

<body>
  <div class="container">
    <h1>Tic Tac Toe</h1>
    <div class="game-container">
      <div class="game-cell"></div>
      <div class="game-cell"></div>
      <div class="game-cell"></div>
      <div class="game-cell"></div>
      <div class="game-cell"></div>
      <div class="game-cell"></div>
      <div class="game-cell"></div>
      <div class="game-cell"></div>
      <div class="game-cell"></div>
    </div>
    <h2 class="game-notification"></h2>
    <button class="game-restart">Restablecer</button>
  </div>
  <script src="main.js"></script>
```

```
</body>

</html>
```

Estilo utilizado con CSS:

```
body {
    font-family: Arial, sans-serif;
    margin: 0;
    padding: 0;
    display: flex;
    justify-content: center;
    align-items: center;
    height: 100vh;
    background-color: #f0f0f0;
}

.container {
    text-align: center;
    background-color: #fff;
    padding: 20px;
    border-radius: 8px;
    box-shadow: 0 2px 4px rgba(0, 0, 0, 0.1);
}

.game-container {
    display: grid;
    grid-template-columns: repeat(3, 100px);
    grid-gap: 10px;
    margin-top: 20px;
}

.game-cell {
    width: 100px;
    height: 100px;
    background-color: #ccc;
    border-radius: 8px;
    display: flex;
    justify-content: center;
    align-items: center;
    font-size: 2em;
}

.game-notification {
    margin-top: 20px;
    color: #333;
}

.game-restart {
    margin-top: 20px;
    padding: 10px 20px;
```

```
    background-color: #007bff;
    color: #fff;
    border: none;
    border-radius: 4px;
    cursor: pointer;
    transition: background-color 0.3s ease;
  }

  .game-restart:hover {
    background-color: #0056b3;
  }
```

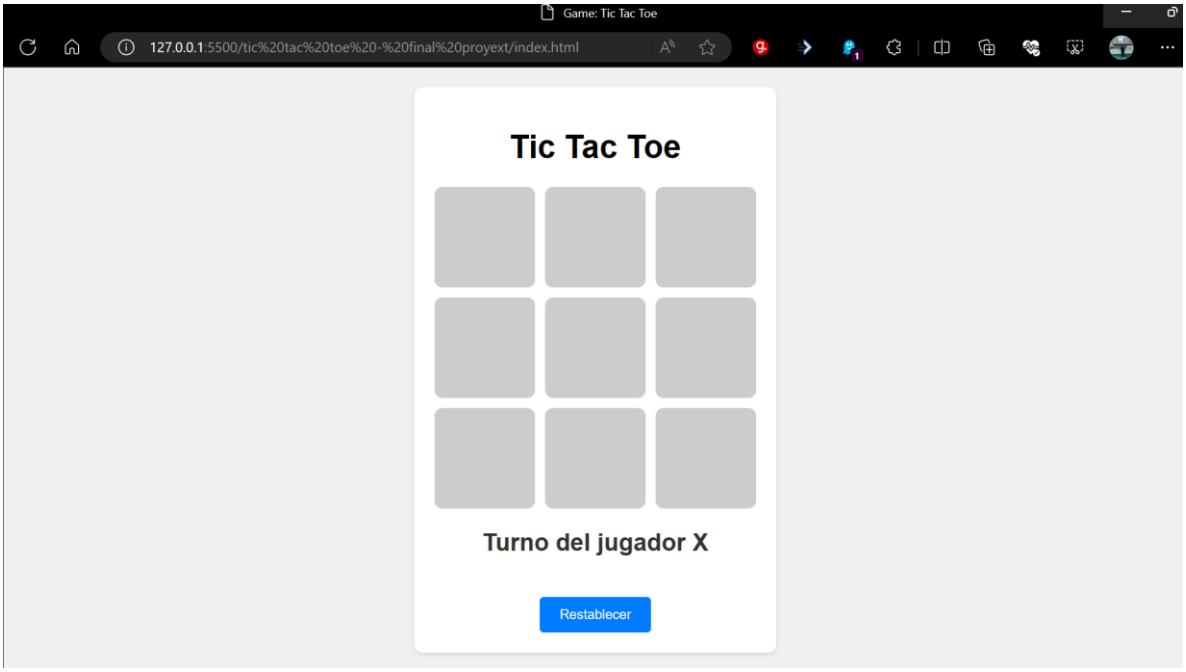
Código utilizado con JS para darle vida al juego, documentación agregada:

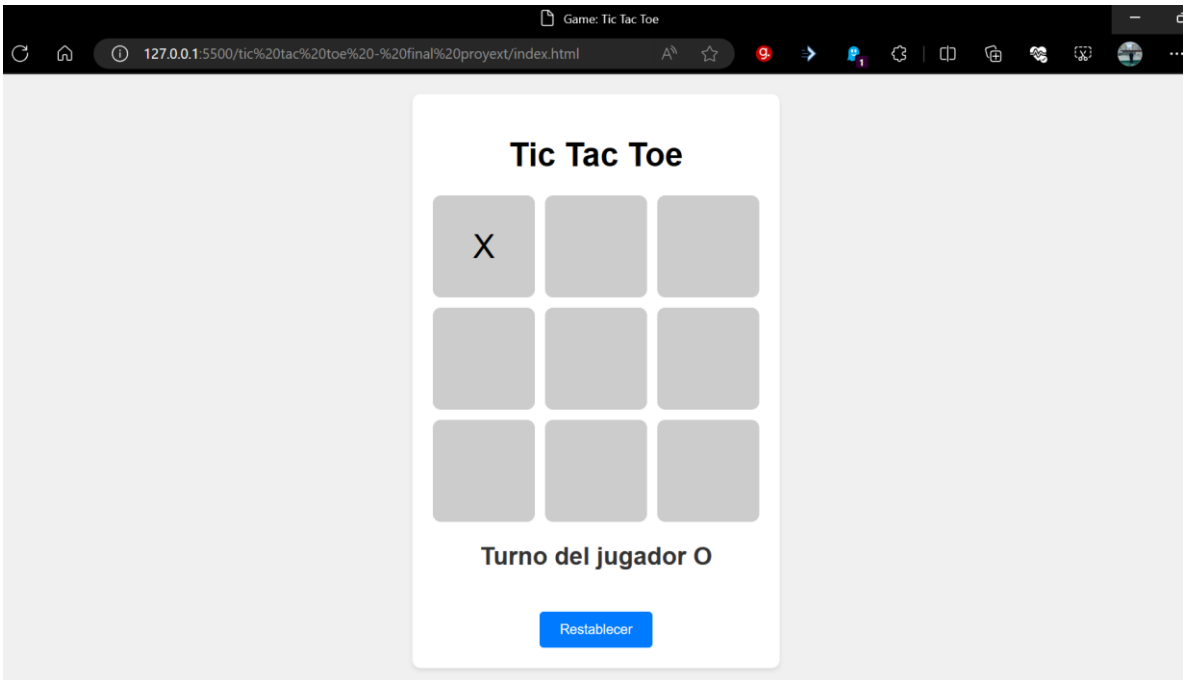
```
// ==================== CONSTANTE ==================== //
const STATUS_DISPLAY = document.querySelector('.game-notification'),
  GAME_STATE = ["", "", "", "", "", "", "", "", ""],
  WINNINGS = [
    [0, 1, 2],
    [3, 4, 5],
    [6, 7, 8],
    [0, 3, 6],
    [1, 4, 7],
    [2, 5, 8],
    [0, 4, 8],
    [2, 4, 6]
  ],
  WIN_MESSAGE = () => `El jugador ${currentPlayer} ha ganado!`,
  DRAW_MESSAGE = () => `El juego ha terminado en empate!`,
  CURRENT_PLAYER_TURN = () => `Turno del jugador ${currentPlayer}`

// ==================== VARIABLES ==================== //
let gameActive = true,
  currentPlayer = "O"

// ==================== FUNCTIONS ==================== //

function main() {
  handleStatusDisplay(CURRENT_PLAYER_TURN())
  listeners()
}

function listeners() {
  document.querySelector('.game-container').addEventListener('click',
handleCellClick)
  document.querySelector('.game-restart').addEventListener('click',
handleRestartGame)
}

function handleStatusDisplay(message) {
```

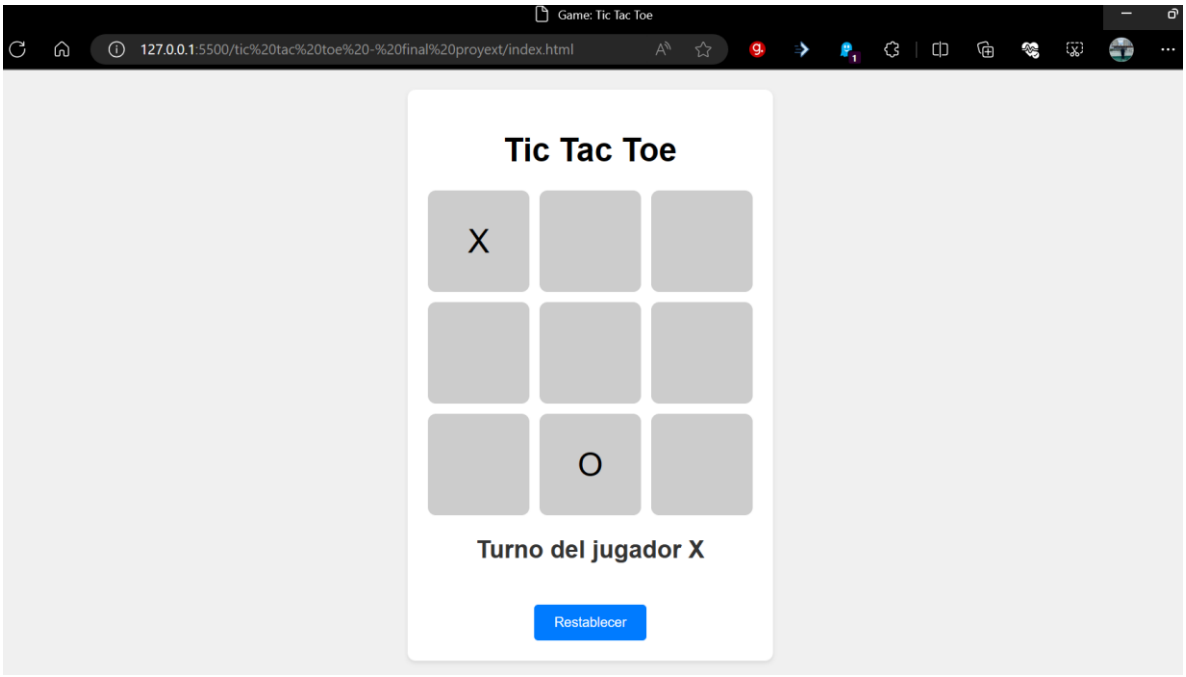
```
37   STATUS_DISPLAY.innerHTML = message
38 }
39
40 function handleRestartGame() {
41   gameActive = true
42   currentPlayer = "X"
43   restartGameState()
44   handleStatusDisplay(CURRENT_PLAYER_TURN())
45   document.querySelectorAll('.game-cell').forEach(cell =>
46 cell.innerHTML = "")
47 }
48
49 function handleCellClick(clickedCellEvent /** Type Event **/) {
50   const clickedCell = clickedCellEvent.target
51   if (clickedCell.classList.contains('game-cell')) {
52     const clickedCellIndex =
53 Array.from(clickedCell.parentNode.children).indexOf(clickedCell)
54     if (GAME_STATE[clickedCellIndex] !== '' || !gameActive) {
55       return false
56     }
57
58     handleCellPlayed(clickedCell, clickedCellIndex)
59     handleResultValidation()
60   }
61 }
62
63 function handleCellPlayed(clickedCell /** object HTML **/,
64 clickedCellIndex) {
65   GAME_STATE[clickedCellIndex] = currentPlayer // Agrega en la
66 posición correspondiente el valor ya sea "X" u "O" en el estado
67 actual del juego
68   clickedCell.innerHTML = currentPlayer // Agrega en el HTML el valor
69 del jugador
70 }
71
72 function handleResultValidation() {
73   let roundWon = false
74   for (let i = 0; i < WINNINGS.length; i++) { // Itera cada uno de
75 las posibles combinaciones ganadores
76     const winCondition = WINNINGS[i] // Guarda la combinación por
77 ejemplo: [0, 1, 2]
78     let position1 = GAME_STATE[winCondition[0]],
79       position2 = GAME_STATE[winCondition[1]],
80       position3 = GAME_STATE[winCondition[2]] // Almacena el valor
81 del estado actual del juego según las posiciones de winCondition
82
83     if (position1 === '' || position2 === '' || position3 === '') {
84       continue; // Si hay algún valor vacio nadie ha ganado aún
85     }
86     if (position1 === position2 && position2 === position3) {
87       roundWon = true // Si todas las posiciones coinciden entonces,
88 dicho jugador ha ganado la partida
```

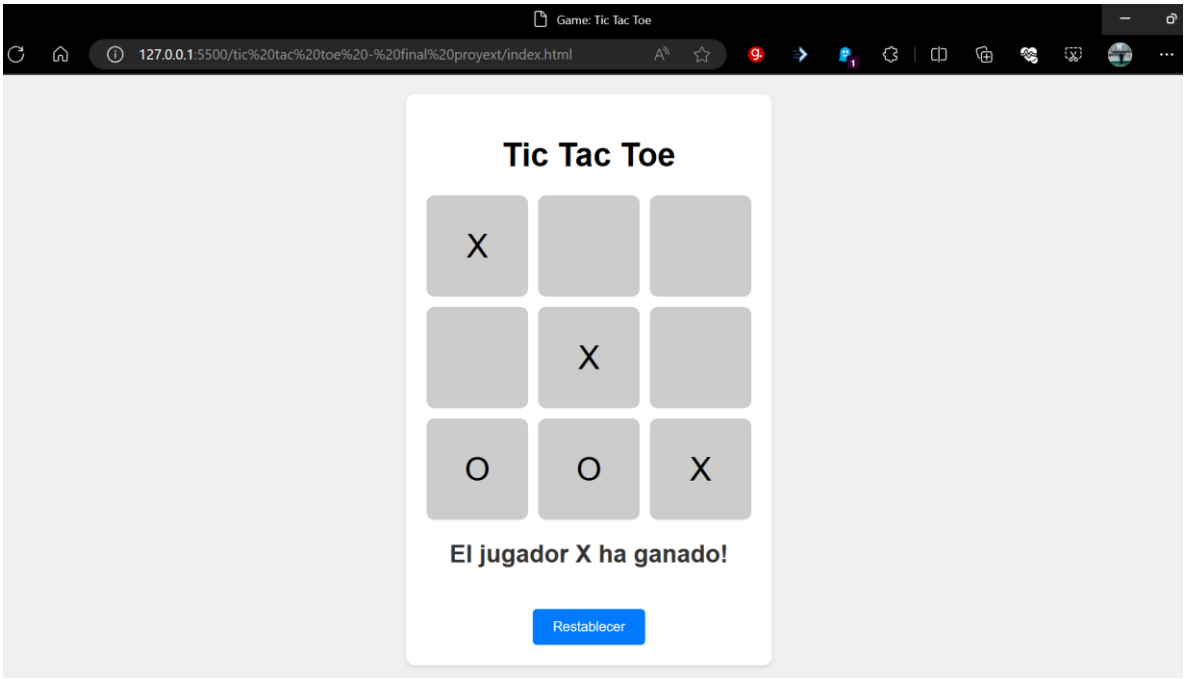
```
      break
    }
  }

  if (roundWon) {
    handleStatusDisplay(WIN_MESSAGE())
    gameActive = false
    return
  }

  let roundDraw = !GAME_STATE.includes("") // Si todas las celdas tienen valor y la sentencia anterior fue falsa entonces es empate
  if (roundDraw) {
    handleStatusDisplay(DRAW_MESSAGE())
    gameActive = false
    return
  }

  handlePlayerChange()
}

function handlePlayerChange() {
  currentPlayer = currentPlayer === "X" ? "O" : "X"
  handleStatusDisplay(CURRENT_PLAYER_TURN())
}

function restartGameState() {
  let i = GAME_STATE.length
  while (i--) {
    GAME_STATE[i] = ''
  }
}

main()
```

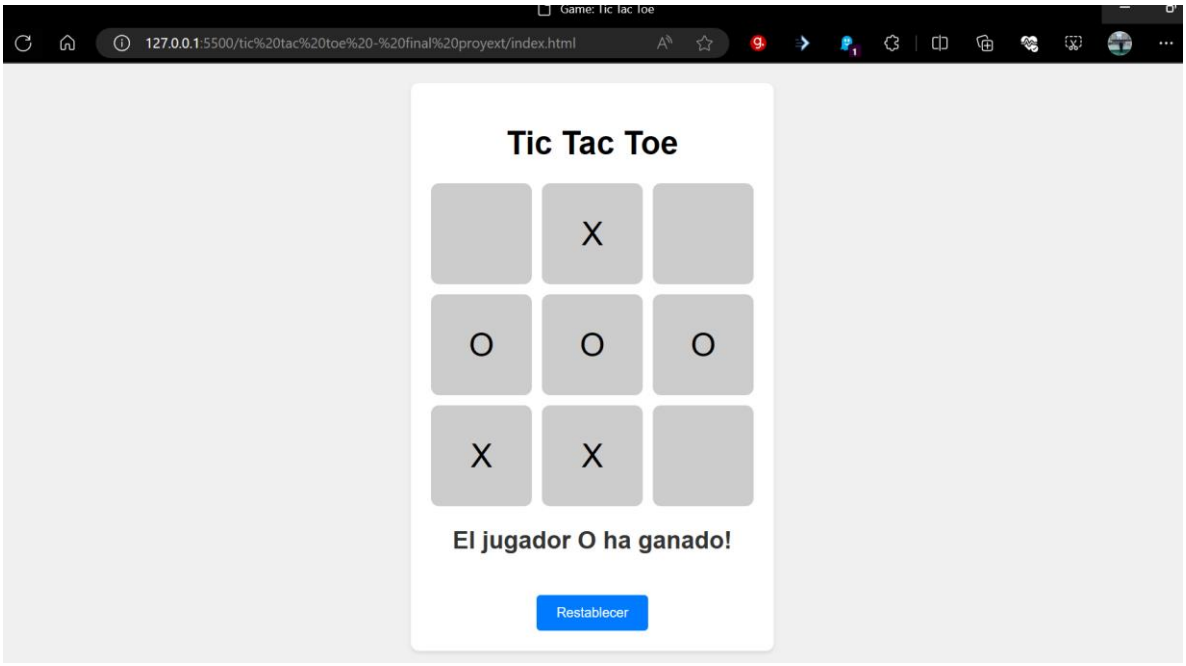
ANEXOS:

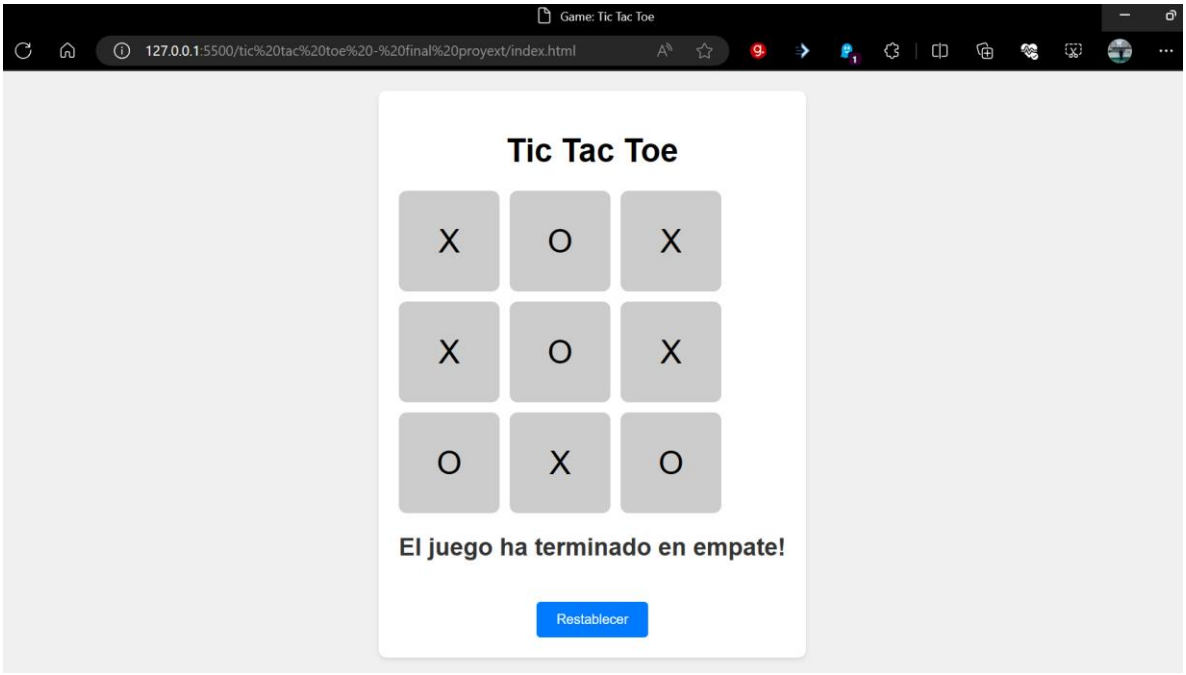

Game: Tic Tac Toe
127.0.0.1:5500/tic%20tac%20toe%20-%20final%20proyext/index.html
Tic Tac Toe
Turno del jugador X
Restablecer


Game: Tic Tac Toe
127.0.0.1:5500/tic%20tac%20toe%20-%20final%20proyext/index.html
Tic Tac Toe
X
Turno del jugador O
Restablecer


Game: Tic Tac Toe
127.0.0.1:5500/tic%20tac%20toe%20-%20final%20proyext/index.html
Tic Tac Toe
X
O
Turno del jugador X
Restablecer


Game: Tic Tac Toe
127.0.0.1:5500/tic%20tac%20toe%20-%20final%20proyext/index.html
Tic Tac Toe
X
X
O
O
X
El jugador X ha ganado!
Restablecer


Game: Tic Tac Toe
127.0.0.1:5500/tic%20tac%20toe%20-%20final%20proyext/index.html
Tic Tac Toe
X
O O O
X X
El jugador O ha ganado!
Restablecer


Game: Tic Tac Toe
127.0.0.1:5500/tic%20tac%20toe%20-%20final%20proyext/index.html
Tic Tac Toe
X O X
X O X
O X O
El juego ha terminado en empate!
Restablecer