Abstract

In today's fast-changing world of technology, web-based applications have become a very important part of our daily lives. Whether it's playing games, studying on online platforms, or using tools for work, web applications are everywhere. This project report is all about designing, developing, and testing one of the most well-known and simple games — **Tic Tac Toe** — as a web-based game that runs in a browser.

The main aim of this project was to learn and apply the basics of web designing using **HTML**, **CSS**, and **JavaScript**. These three languages are the foundation of any website or web application. HTML helps to create the structure of the page, CSS is used to make the design attractive and stylish, and JavaScript brings the game to life by making it interactive and fun to play.

Tic Tac Toe is a classic two-player game where each player takes turns marking a space in a 3x3 grid until one player wins or the game ends in a draw. Although the game looks simple, it is a great way to learn about web design, user experience (UX), user interface (UI), and logical thinking. This project helped me understand how important it is to create a clear layout, easy navigation, and smooth interactions for the users.

The report explains each step taken in the project — starting from the **idea stage**, moving to the **design phase**, and then to **coding, testing, and fixing errors**. It also looks at how the game can be improved in the future, for example, by adding an automatic computer opponent, better graphics, or making it mobile-friendly so people can play it on their phones and tablets.

In the end, this project is not just about making a game. It's about learning the basics of web development in a simple and fun way. By creating a project like Tic Tac Toe, beginners can practice how to design, build, and improve a web application while focusing on important skills like problem-solving, creativity, and user satisfaction.

Introduction

Tic Tac Toe is a simple, well-known, and interesting two-player game that has been played by people of all ages for many years. The game is played on a small square grid made up of 3 rows and 3 columns, which is often called a **3x3 grid**. In this game, one player uses the symbol **'X'** and the other player uses **'O'**. Players take turns placing their symbols in any empty space on the grid. The main goal of each player is to form a

straight line of three matching symbols before the other player does. This line can be arranged in three different ways: **horizontally** (side to side), **vertically** (up and down), or **diagonally** (corner to corner). If all nine spaces on the grid are filled and neither player is able to form a straight line, the game is declared a **draw**.

The purpose of this web designing project was to bring the fun and challenge of the Tic Tac Toe game to the digital world by creating an online version that can be played in a web browser on computers, laptops, tablets, or smartphones. This project aimed to give the game a **clean, simple, and user-friendly design** so that users can enjoy the classic gameplay without any confusion or difficulties. The focus was on creating an interactive and smooth gaming experience that works well on different devices.

Through the development of this project, several essential web development concepts were practiced and applied. These include creating and styling a grid-based layout for the game board, handling user events (such as clicking a box), dynamically updating content on the webpage, managing the game state to keep track of turns and wins, and providing clear feedback messages to the players to display the winner or declare a draw.

The Tic Tac Toe game is a perfect project for beginners who are learning web design and development because it teaches important basic skills in a fun and creative way. It helps learners understand the connection between **HTML, CSS, and JavaScript**, and how these three languages work together to build interactive web applications.

This project report explains the design process, the technologies used, the step-by-step implementation, and the valuable lessons learned during the creation of this simple but educational web game. It also highlights the importance of clear design, responsive interaction, and smooth user experience while building web-based applications.

Background

The world of web-based games started growing many years ago, around the early 1990s, when websites were becoming popular and people started learning how to make them more interactive. In the beginning, web pages were very simple, and there were no fun features like animations or games. Websites were mostly plain text with a few images and links.

As time passed, new web technologies like **HTML, CSS, and JavaScript** were introduced and improved. **HTML (HyperText Markup Language)** is used to create the structure of a website, like headings, paragraphs, and buttons. **CSS (Cascading Style Sheets)** is used to

style the website, such as setting colors, fonts, sizes, and layouts. **JavaScript** is used to add interactive features like buttons that respond to clicks, forms that can check for errors, and even games.

When JavaScript became more powerful, it allowed developers to create small games directly inside a web browser, without needing to install anything. These games could run on computers, laptops, tablets, and even mobile phones, as long as the device had a browser. This opened up new opportunities for both fun and learning, and many simple games were created, including **Tic Tac Toe**.

Tic Tac Toe is one of the oldest and simplest strategy games known to people. The game is also known as **Noughts and Crosses** in Britain and some other countries. The exact origins of Tic Tac Toe are hard to track, but it has been played in different forms for hundreds of years. The rules are very easy to learn: two players take turns marking either an "X" or an "O" on a 3x3 grid. The first player to get three of their marks in a row, either horizontally, vertically, or diagonally, wins the game. If all spaces are filled and no one has three in a row, the game ends in a draw.

Even though the game is simple, it is a great example of strategic thinking and problem-solving. Players must think ahead, recognize patterns, block their opponent's moves, and plan their own path to victory. Because of these reasons, **Tic Tac Toe** is often used as an example in programming and computer science courses. It is simple enough for beginners but still teaches important lessons about logic and planning.

Many students and new developers create Tic Tac Toe as their first project when learning web development. This game helps them understand and practice important programming concepts like:

- **Loops** – for repeating actions,
- **Conditional statements** – for making decisions,
- **Arrays** – for storing game data like Xs and Os,
- **Event-driven programming** – for reacting to player clicks.

Making a Tic Tac Toe game for the web also gives students practice with designing the **user interface (UI)** and handling **user experience (UX)**. This means not only making the game work but also making it easy and fun for the player to use.

Because of all these benefits, **Tic Tac Toe** is a perfect project for anyone starting to learn web design and programming. It is a small project, but it covers many important skills that form the foundation for bigger and more complex projects in the future.

Project Objectives

The main goal of this project was to design and develop a simple, fun, and fully functional web-based version of the classic Tic Tac Toe game. To achieve this, the following clear and achievable objectives were defined:

A. **Design a Responsive and Intuitive 3x3 Grid-based Web Application**
The project aimed to create a neat and organized game board that is based on a 3x3 grid layout. The design had to be user-friendly so that players can easily understand how to play the game, no matter their age or experience level. The grid should adjust automatically to fit different screen sizes, whether on a desktop, laptop, tablet, or mobile phone.

B. **Implement Basic and Advanced JavaScript Logic for Game Rules**
The project focused on using JavaScript to write the main logic of the game. This includes checking which player's turn it is, detecting when a player has won, identifying when the game ends in a draw, and resetting the game for a new round. Both basic and slightly advanced JavaScript concepts were applied to make sure the game runs smoothly.

C. **Ensure Real-time Interaction and Immediate Feedback to Users**
The game needed to respond instantly to user actions. Whenever a player clicks on a square, the game should immediately show their chosen symbol (X or O) and update the game's status, such as displaying which player's turn it is or who won the match. This real-time interaction makes the game more engaging and fun.

D. **Utilize Minimalistic and Clean Design Principles**
The game's design followed a simple and neat visual style. The aim was to avoid unnecessary decorations and focus on providing a clean and clear playing area, which helps users concentrate on the game rather than get distracted by too many design elements.

E. **Optimize the Game's User Interface for Both Desktop and Mobile Devices**
The game was designed to be responsive and flexible, meaning it should work perfectly on various devices and screen sizes. Whether the user is playing on a large computer monitor or a small mobile phone, the game should look good and work well.

F. **Create a Modular, Scalable, and Maintainable Codebase**
The project aimed to write clean and organized code by dividing it into small, easy-to-manage parts (called modules). This makes the code easier to understand, allows future developers to add new features, and ensures the game

can grow or be improved over time.

G. **Test the Game Under Various Conditions to Ensure Reliability**
The game was tested many times under different conditions, such as different browsers, devices, and screen sizes, to make sure it works correctly in all situations. This helps make the game reliable and bug-free for all users.

H. **Propose and Document Potential Future Improvements**
Lastly, the project included planning for possible new features and improvements that could be added in the future. Writing down these ideas helps guide future updates and encourages continuous learning and development.

Code

HTML FILE

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Tic Tac Toe</title>
                                                                    <link
href="https://fonts.googleapis.com/css2?family=SF+Pro+Display:wght@400;600
&display=swap" rel="stylesheet">
  <link rel="stylesheet" href="styles.css">
</head>
<body>
  <div class="container">
    <h1>Tic Tac Toe</h1>
    <div class="board" id="board">
      <div class="cell" data-index="0"></div>
      <div class="cell" data-index="1"></div>
      <div class="cell" data-index="2"></div>
      <div class="cell" data-index="3"></div>
      <div class="cell" data-index="4"></div>
```

```
            <div class="cell" data-index="5"></div>
            <div class="cell" data-index="6"></div>
            <div class="cell" data-index="7"></div>
            <div class="cell" data-index="8"></div>
        </div>
        <button id="resetButton">Reset Game</button>
        <div class="status" id="status"></div>
    </div>
    <script src="script.js"></script>
</body>
</html>
```

CSS FILE

```
body {
    font-family: 'SF Pro Display', sans-serif;
    background-color: #f0f0f0;
    display: flex;
    justify-content: center;
    align-items: center;
    height: 100vh;
    margin: 0;
}
.container {
    text-align: center;
    background: white;
    border-radius: 10px;
    box-shadow: 0 4px 20px rgba(0, 0, 0, 0.1);
    padding: 20px;
}
h1 {
    margin-bottom: 20px;
    color: #333;
}
.board {
    display: grid;
    grid-template-columns: repeat(3, 100px);
    grid-template-rows: repeat(3, 100px);
    gap: 5px;
```

```
    margin-bottom: 20px;
}
.cell {
    width: 100px;
    height: 100px;
    background-color: #e0e0e0;
    display: flex;
    justify-content: center;
    align-items: center;
    font-size: 2rem;
    cursor: pointer;
    transition: background-color 0.3s;
}
.cell:hover {
    background-color: #d0d0d0;
}
button {
    padding: 10px 20px;
    font-size: 1rem;
    border: none;
    border-radius: 5px;
    background-color: #007bff;
    color: white;
    cursor: pointer;
    transition: background-color 0.3s;
}
button:hover {
    background-color: #0056b3;
}
.status {
    margin-top: 20px;
    font-size: 1.2rem;
    color: #333;
}
```

JAVASCRIPT FILE

```
const board = document.getElementById('board');
const cells = document.querySelectorAll('.cell');
```

```
const statusDisplay = document.getElementById('status');
const resetButton = document.getElementById('resetButton');
let currentPlayer = 'X';
let gameActive = true;
let gameState = ['', '', '', '', '', '', '', '', ''];
const winningConditions = [
    [0, 1, 2],
    [3, 4, 5],
    [6, 7, 8],
    [0, 3, 6],
    [1, 4, 7],
    [2, 5, 8],
    [0, 4, 8],
    [2, 4, 6]
];
function handleCellClick(event) {
    const clickedCell = event.target;
    const clickedCellIndex = parseInt(clickedCell.getAttribute('data-index'));
    if (gameState[clickedCellIndex] !== '' || !gameActive) return;
    gameState[clickedCellIndex] = currentPlayer;
    clickedCell.innerHTML = currentPlayer;
    checkResult();
}
function checkResult() {
    let roundWon = false;
    for (let i = 0; i < winningConditions.length; i++) {
        const [a, b, c] = winningConditions[i];
        if (gameState[a] === '' || gameState[b] === '' || gameState[c] === '') continue;
        if (gameState[a] === gameState[b] && gameState[a] === gameState[c]) {
            roundWon = true;
            break;
        }
    }
    if (roundWon) {
        statusDisplay.innerHTML = `Player ${currentPlayer} wins!`;
        gameActive = false;
        return;
    }
    if (!gameState.includes('')) {
```

```
        statusDisplay.innerHTML = `It's a draw!`;
        gameActive = false;
        return;
    }
    currentPlayer = currentPlayer === 'X' ? 'O' : 'X';
    statusDisplay.innerHTML = `It's ${currentPlayer}'s turn`;
}
function handleResetGame() {
    gameActive = true;
    currentPlayer = 'X';
    gameState = ['', '', '', '', '', '', '', '', ''];
    statusDisplay.innerHTML = `It's ${currentPlayer}'s turn`;
    cells.forEach(cell => cell.innerHTML = '');
}
cells.forEach(cell => cell.addEventListener('click', handleCellClick));
resetButton.addEventListener('click', handleResetGame);
```

Code

This project has been an exciting and insightful journey into the world of web development and game design. Working on the **Tic Tac Toe** game allowed me to dive into many important areas of web programming, including **modular programming**, **user-centric design**, **event handling**, and **responsive styling**. These concepts were not only essential for making the game functional but also helped me understand how games and websites work behind the scenes.

Throughout this project, I was able to practice coding by breaking down the game into smaller parts, making it easier to manage and improve. This process of **modular programming** helped me learn how to write clean, organized code that can be reused in different parts of the game. By focusing on the **user experience**, I made sure that the game was easy to play, visually appealing, and worked well on different screen sizes, whether on a computer or mobile phone.

Another important aspect was **event handling**. This involved programming the game to respond to user actions, like when a player clicks a spot on the board to place their mark. It was satisfying to see how the game reacted to each move, whether it was a win, draw, or an ongoing game. Additionally, using **responsive styling** ensured that the game

layout adapted to different screen sizes, creating a smooth experience for users, no matter what device they were using.

The project also involved a lot of **testing and debugging**, which were key steps in making sure the game worked properly. It was sometimes challenging to find and fix bugs, but these challenges helped me develop problem-solving skills and a deeper understanding of how to make things work as expected.

In addition to technical skills, the project helped me build soft skills like **collaboration**, since I worked through some ideas with peers and sought feedback, and **design thinking**, which taught me how to focus on making the game easy and enjoyable for users.

Overall, this project was an excellent opportunity to improve my programming skills and gain hands-on experience with real-world web development. It has laid a strong foundation for me to tackle more complex projects in the future, and I feel more confident in my ability to design, code, and troubleshoot interactive web applications.