## 1) Introduction and Problem:

Tic-Tac-Toe is a very simple two player game. So only two players can play at a time. This game is also known as Noughts and Crosses O(s) and X(s) game. One player plays with X and the other player plays with O. In this game we have a board consisting of a 3X3 grid. The number of grids may be increased. This Game in C# is a simple GUI-based strategy board game that is very easy to understand and use. This Tic Tac Toe Game is a project that provides a fun moment that can be played with your friends. This game provides some learning references to help those students who want to pursue their game developer career.

## 2) Technology:

This Game in C# is a simple GUI-based strategy board game that is very easy to understand and use. This is a simple multiplayer game. Game is an interesting and addictive game C#, In the most rapid and convenient way to create your user interface is to do so visually, using the Windows Forms Designer and Toolbox. Windows Forms controls are reusable components that encapsulate user interface functionality and are used in client-side Windows based applications.

## 3) Customise in current project:

Tic-Tac-Toe Game Application is changed from pencil-paper game to a computer game Where 2 players sit at a computer and play the same game.

## 4) Functionalities:

The game is developed for full-time entertainment and enthusiasm. It teaches the Gamer to be alert at every situation he/she faces, because if the Gamer is not fully alert and notices the saucer fire, he/she must be hit by the saucer-bombs. Though the proposed game is an action game, it doesn't involve direct violence. No zombie killing, animal killing, or human killing is performed in the game. So, it can also be viewed as a non-violence game. Kids can also play this game, because the design of the game is very simple, controlling the game is very easy – pressing some neighboring keys of the keyboard.

Some Rules are as Follow:

- Traditionally the first player plays with "X". So, you can decide who wants to go "X" and who wants to go with "O".
- Only one player can play at a time.

- If any of the players have filled a square, then the other player and the same player cannot override that square.
- There are only two conditions that may be match will be draw or may be win.
- The player that succeeds in placing three respective marks (X or O) in a horizontal, vertical, or diagonal row wins the game.

## 5) Module Distribution:

### Admin Module:

- Admin of the system can update the player's score.
- Admin of the system can maintain the record of player and high scores.

### Player Module:

- The players can see the game table and start playing.
- They can place appropriate Crosses or Noughts on the table.
- Players can view the result at end and view their Scores.

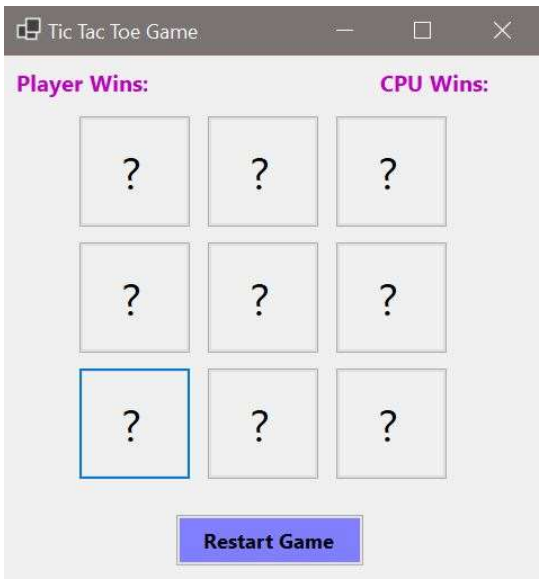
## 6) Code:

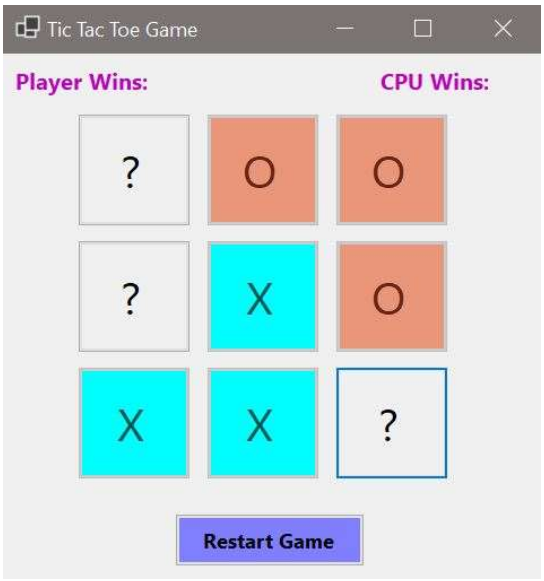
```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace Tic_Tac_Toe_Game
{
    public partial class Form1 : Form
    {


        public enum Player
        {
            X, O
        }

        Player currentPlayer;
        Random random = new Random();
        int playerWinCount = 0;
        int CPUWinCount = 0;
        List<Button> buttons;

        public Form1()
        {
            InitializeComponent();
            RestartGame();
        }
```

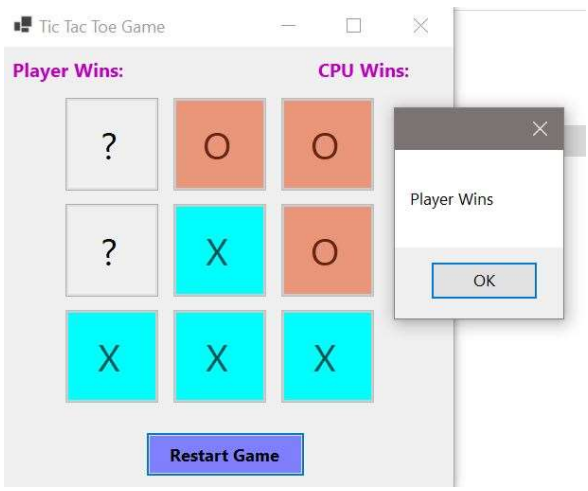
```
private void button9_click(Object sender, EventArgs e)
{

}

private void CPUmove(object sender, EventArgs e)
{
    if (buttons.Count > 0)
    {
        int index = random.Next(buttons.Count);
        buttons[index].Enabled = false;
        currentPlayer = Player.O;
        buttons[index].Text = currentPlayer.ToString();
        buttons[index].BackColor = Color.DarkSalmon;
        buttons.RemoveAt(index);
        CheckGame();
        CPUTimer.Stop();


    }
}

private void PlayerClickButton(object sender, EventArgs e)
{
    var button = (Button)sender;

    currentPlayer = Player.X;
    button.Text = currentPlayer.ToString();
    button.Enabled = false;
    button.BackColor = Color.Cyan;
    buttons.Remove(button);
    CheckGame();
    CPUTimer.Start();


}

private void RestartGame(object sender, EventArgs e)
{
    RestartGame();
}

private void CheckGame()
{
    if (button1.Text == "X" && button2.Text == "X" && button3.Text == "X"
        || button4.Text == "X" && button5.Text == "X" && button6.Text == "X"
        || button7.Text == "X" && button8.Text == "X" && button9.Text == "X"
        || button1.Text == "X" && button4.Text == "X" && button7.Text == "X"
        || button2.Text == "X" && button5.Text == "X" && button8.Text == "X"
        || button3.Text == "X" && button6.Text == "X" && button9.Text == "X"
        || button1.Text == "X" && button5.Text == "X" && button9.Text == "X"
        || button3.Text == "X" && button5.Text == "X" && button7.Text == "X"
        )
    {
        CPUTimer.Stop();
        MessageBox.Show("Player Wins");
        playerWinCount++;
        label1.Text = "Player Wins:" + playerWinCount;
        RestartGame();
    }
    else if (button1.Text == "O" && button2.Text == "O" && button3.Text == "O"
```

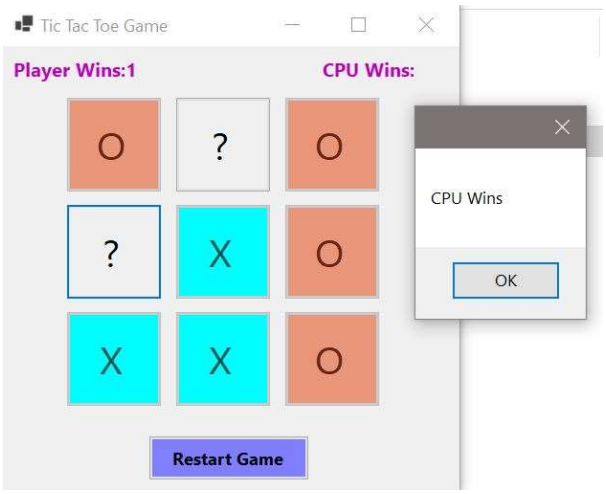
```
                || button4.Text == "O" && button5.Text == "O" && button6.Text == "O"
                || button7.Text == "O" && button8.Text == "O" && button9.Text == "O"
                || button1.Text == "O" && button4.Text == "O" && button7.Text == "O"
                || button2.Text == "O" && button5.Text == "O" && button8.Text == "O"
                || button3.Text == "O" && button6.Text == "O" && button9.Text == "O"
                || button1.Text == "O" && button5.Text == "O" && button9.Text == "O"
                || button3.Text == "O" && button5.Text == "O" && button7.Text == "O"
                )
            {
                CPUTimer.Stop();
                MessageBox.Show("CPU Wins");
                CPUWinCount++;
                label2.Text = "CPU Wins:" + CPUWinCount;
                RestartGame();
            }


        }

        private void RestartGame()
        {
            buttons = new List<Button> { button1, button2, button3, button4, button5, button6,
button6, button7, button8, button9 };
            foreach (Button x in buttons)
            {
                x.Enabled = true;
                x.Text = "?";
                x.BackColor = DefaultBackColor;

            }
        }
        private void Form1_Load(Object sender, EventArgs e)
        {

        }
    }
}
```

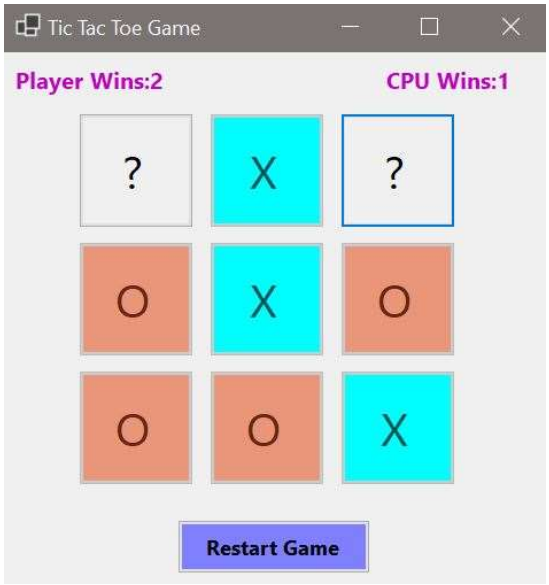
## 7) Interfaces:






## 8) Conclusion:

The Tic Tac Toe game is the most familiar among all the age groups. Intelligence can be a property of any purpose-driven decision maker. This basic idea has been suggested many times. An algorithm of playing Tic Tac Toe has been presented and tested to show that works in an efficient way. Overall, the system works without any bugs. The player who succeeded in placing three respective marks in a horizontal, vertical, or diagonal row wins the game.