TIC TAC TOE Quest:

Master the Board

A PROJECT REPORT

Submitted by

D.BHARATH[192210393]
N.PRANITH[192210437]

Under the guidance of
Dr. E .ANBALAGAN.

In partial fulfillment for the completion of course CSA0888-

PYTHON PROGRAMMINGFOR NATURAL LANGUAGE

PROCESSING

SIMATS ENGINEERING

THANDALAM MARCH 2024

BONAFIDE CERTIFICATE

Certified that this project report titled "**TIC TAC TOE**" is the bonafide work of "**D.BHARATH[192210393], N.PRANITH[192210437]**"who carried out the project work under my supervision as a batch. Certified further, that to the best of my knowledge the work reported herein does not form any other project report.

Date: Head of the department

ABSTRACT

The Tic Tac Toe game is a classic two-player strategy game played on a 3x3 grid. It is a simple yet engaging game where players take turns marking spaces on the grid with their respective symbols (X or O) with the objective of forming a horizontal, vertical, or diagonal line of their symbols. This project aims to implement the Tic Tac Toe game using Python programming language and the Pygame library to provide a graphical user interface.The game allows two human players to take turns playing against each other on the same device. Players can click on the grid squares to place their symbols, and the game automatically detects winning conditions and announces the winner.

Additionally, the game includes features such as a reset button to start a new game and a visual indicator of the current player's turn.The implementation of the Tic Tac Toe game serves as an excellent exercise for beginners to understand basic game development concepts, including game logic, user interaction, and graphical representation. Furthermore, it provides an enjoyable and interactive experience for players of all ages, making it a popular choice for recreational purposes.

The actual game of TIC TAC TOE could be traced back to ancient Egypt. The game was also known during that time as "Three Men's Morris". The first reference to Noughts and crosses was made in 1864 in a British novel called Can You Forgive Her. For many years the game was referred to as noughts and crosses but was changed in the 20th century by the United States to TIC TAC TOE.

The purpose of this documentation is to capture all the requirements by which the user can play a game of tic-tac-toe in Graphical User Interface as well as they can develop a logic that what is actually happening.

1.INTRODUCTION

One of the most universally played childhood games is TIC TAC TOE. An interactive TIC TAC TOE game is developed where two players will be able to play against each other in a suitable GUI by using proper mouse movements. This game will start by showing a simple display, prompt the user for a move and then prints the new board. The board looks like a large hash symbol (#)with nine slots that can each contain an X, an O, or a blank. There are two players, the X player and the O player. By default, player1 (the O player) takes the initiative. The game will end in two situations: a win, when one player gets three in a row, horizontally, vertically or diagonally. A draw, when there are no remaining places to choose and neither of them has won. Here are some rules for the game:

- The player with symbol ‘O’ goes first
- Players alternate placing X s and O s on the board until either
- one player has three in a row, horizontally, vertically or
- diagonally; orAll nine squares are filled.
- The player who can draw three X s or three O s in a row wins.
- If all nine squares are filled and none of the players have three in a row,the game is adraw.

2.PROBLEM FORMULA

3.1 DESCRIPTION OF PROBLEM DOMAIN

The problem domain for Tic Tac Toe revolves around the concept of a two-player game playedon a 3x3 grid. Tic Tac Toe, also known as Noughts and Crosses, is a simple yet engaging gamewhere players take turns marking spaces in the grid with their respective symbols, typically "X" and "O", with the objective of forming a row, column, or diagonal of their symbols.

The game begins with an empty grid, and player’s alternate turns, each marking an empty spacewith their symbol. The game continues until one player achieves the winning condition or theentire grid is filled, resulting in a draw. The winning condition occurs when a player successfully places their symbols in a line horizontally, vertically, or diagonally.

Tic Tac Toe is a classic example of a zero-sum game, where the outcome for one player is directly tied to the outcome for the other player. It is a game of perfect information, meaning that both players have complete knowledge of the game state at all times, and there is no element of chance involved

3.2 PROBLEM STATEMENT.

The problem statement for the Tic Tac Toe game project involves developing a Python programthat allows two players to play the classic game of Tic Tac Toe against each other on a computer. The primary objective is to create an interactive and functional game interface whereplayers can take turns making moves and the program accurately determines the game's outcome based on the players' moves.

Specifically, the program should address the following key aspects:

1. Game Interface: Design and implement a graphical user interface (GUI) that displays the TicTac Toe game board and allows players to interact with it. The interface should provide visualfeedback for player moves and clearly indicate the current game state.

2. Player Input: Enable players to make moves by clicking on the game board squares corresponding to their desired positions. Validate player input to ensure that moves are legal (i.e., made on empty squares) and update the game board accordingly.

3. Game Logic: Implement the rules of Tic Tac Toe to determine the game's outcome. Detect when a player has achieved a winning configuration (three symbols in a row, column, or diagonal) or when the game ends in a draw (all squares filled without a winner).

3.2 OBJECTIVE

The objectives of developing a Tic Tac Toe game can be outlined as follows:

Create a Functional Game Interface: Design and implement a graphical user interface (GUI) that provides an interactive game board where players can make moves.

Enable Player Interaction: Allow players to take turns making moves by selecting empty squares on the game board.

Implement Game Logic: Develop the underlying logic of the game to accurately determine the outcome based on player moves. This includes detecting winning configurations and identifying when the game ends in a draw.

Ensure Turn Management: Implement a mechanism to alternate turns between players,ensuring that each player gets a chance to make a move.

Provide Clear Feedback: Offer visual feedback to players for their moves and display messages indicating the current game state, such as whose turn it is or when the game ends.

Handle User Input: Validate player input to prevent illegal moves and provide error messagesor prompts when necessary.

4.IMPLEMENTATION

4.1 PYTHON

Python is an object-oriented, high level language, interpreted, dynamic and multipurpose programming language.

Python is easy to learn yet powerful and versatile scripting language which makes itattractive for Application Development

Python's syntax and dynamic typing with its interpreted nature, make it an ideallanguage for scripting and rapid application development in many areas

Python supports multiple programming patterns, including object

4.2 PY-GAME

Python is the most popular programming language or nothing wrongto say that it isthe next generation programming language.

In every emerging field in computer science, Python makes its presence activelyPython has vast libraries for various fields such as Machine Learning (Numpy, Pandas,
Matplotih), Artificial intelligence

(Pytorch, TensorFlow), and Game development

Game programming is very rewarding nowadays and it can also be used in advertising and as a teaching tool too. Game development includes mathematics, logic, physics, AL, and much more and it can be amazingly fine.In python, game programming is done in pygame and it is one of the best modules for doing so

Pygame is a cross-platform set of Python modules which is used to create videogames. It consists of computer graphics and sound libraries designed to be usedwith the Python

programming language. Pygame was officially written by Pete Shinners to replace PySDL. Pygame is suitable to create client-side applications that can be potentially wrapped in a standalone executable.

5.SOURCE CODE

```
import pygame
import sys

pygame.init()



WIDTH, HEIGHT = 300, 300

LINE_COLOR = (0, 0, 0)

BG_COLOR = (255, 255, 255)

LINE_WIDTH = 5 CELL_SIZE
= WIDTH // 3X_COLOR =
(255, 0, 0)
O_COLOR = (0, 0, 255)


screen = pygame.display.set_mode((WIDTH, HEIGHT))
pygame.display.set_caption("Tic Tac Toe")

def draw_grid():

    for i in range(1, 3):

        pygame.draw.line(screen, LINE_COLOR, (0, i * CELL_SIZE), (WIDTH,i *
CELL_SIZE), LINE_WIDTH)

        pygame.draw.line(screen, LINE_COLOR, (i * CELL_SIZE, 0), (i *
CELL_SIZE, HEIGHT), LINE_WIDTH)
```

```
def draw_XO(row, col, symbol):
    x = col * CELL_SIZE + CELL_SIZE // 2 y =
    row * CELL_SIZE + CELL_SIZE // 2if
    symbol == "X":
        pygame.draw.line(screen, X_COLOR, (x - CELL_SIZE // 4, y - CELL_SIZE // 4), (x + CELL_SIZE // 4, y + CELL_SIZE // 4), LINE_WIDTH)
        pygame.draw.line(screen, X_COLOR, (x + CELL_SIZE // 4, y - CELL_SIZE // 4), (x - CELL_SIZE // 4, y + CELL_SIZE // 4), LINE_WIDTH)
pygame.draw.circle(screen, O_COLOR, (x, y), CELL_SIZE // 4, LINE_WIDTH)


        def get_row_col_from_pos(pos):
            x, y = pos
            row = y // CELL_SIZE
            col = x // CELL_SIZE
            return row, col

        def is_winner(board, symbol):

            for row in board:
                if all(cell == symbol for cell in row):
                    return True


            for col in range(3):
                if all(board[row][col] == symbol for row in range(3)):return
                    True

            # Check diagonals
            if all(board[i][i] == symbol for i in range(3)) or all(board[i][2 - i] == symbol
```

```
for i in range(3)):
        return True

    return False


def          main():
    running = True
    board = [[" " for _ in range(3)] for _ in range(3)]current_symbol
    = "X"
    winner = None
    while                    running:
        screen.fill(BG_COLOR)
        draw_grid()


        for event in pygame.event.get():
            if event.type == pygame.QUIT:
                running          =          False
                pygame.quit()
                sys.exit()

            if event.type == pygame.MOUSEBUTTONDOWN and not winner:pos
                = pygame.mouse.get_pos()
                row, col = get_row_col_from_pos(pos)
                if board[row][col] == " ":
                    board[row][col] = current_symbol
                    if is_winner(board, current_symbol):
                        winner = current_symbol
                    else:
                        current_symbol = "O" if current_symbol == "X" else "X"
```

```
        for row in range(3):
            for col in range(3):
                if board[row][col] == "X":
                    draw_XO(row, col, "X")
                elif board[row][col] == "O":
                    draw_XO(row, col, "O")



        if winner:
            font = pygame.font.Font(None, 40)
            text = font.render(f"{winner} wins!", True, LINE_COLOR)
            text_rect = text.get_rect(center=(WIDTH // 2, HEIGHT // 2))
            screen.blit(text, text_rect)

        pygame.display.flip()

if __name__ == "__main__":
    main()
            font = pygame.font.Font(None, 40)
            text = font.render(f"{winner} wins!", True, LINE_COLOR)
            text_rect = text.get_rect(center=(WIDTH // 2, HEIGHT // 2))
            screen.blit(text, text_rect)

        pygame.display.flip()

if __name__ == "__main__":
    main()
```

6.RESULT

Output design this application "**TIC TAC TOE"** generally refers to the results and information that are generated by the system for many end-users; output is the main reason for developing the system and the basis on which they evaluate the usefulness of the application.

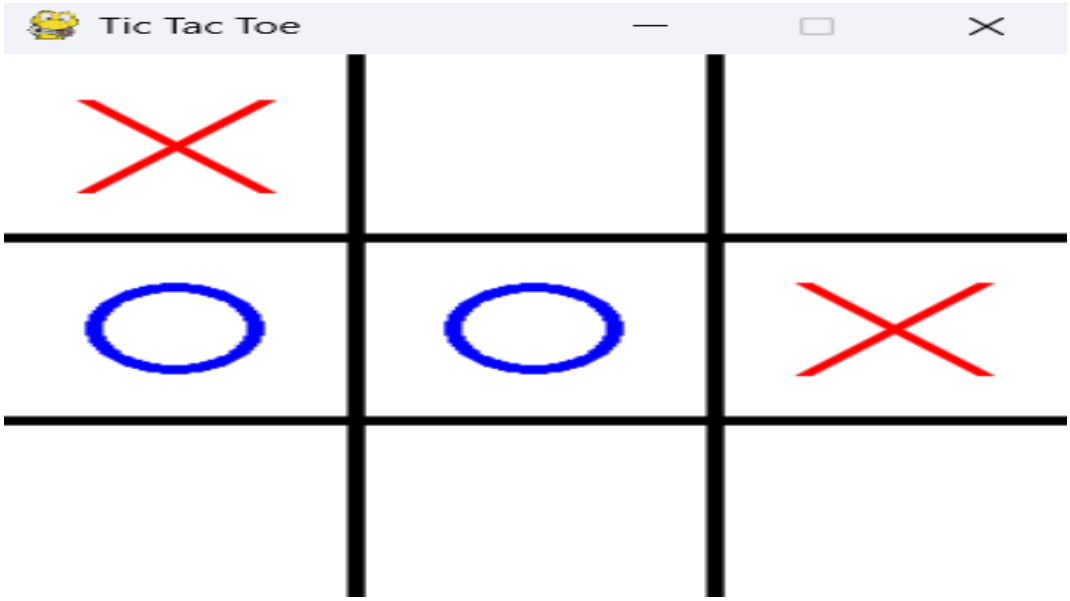

Fig 8.1 Tic Tac Toe game output

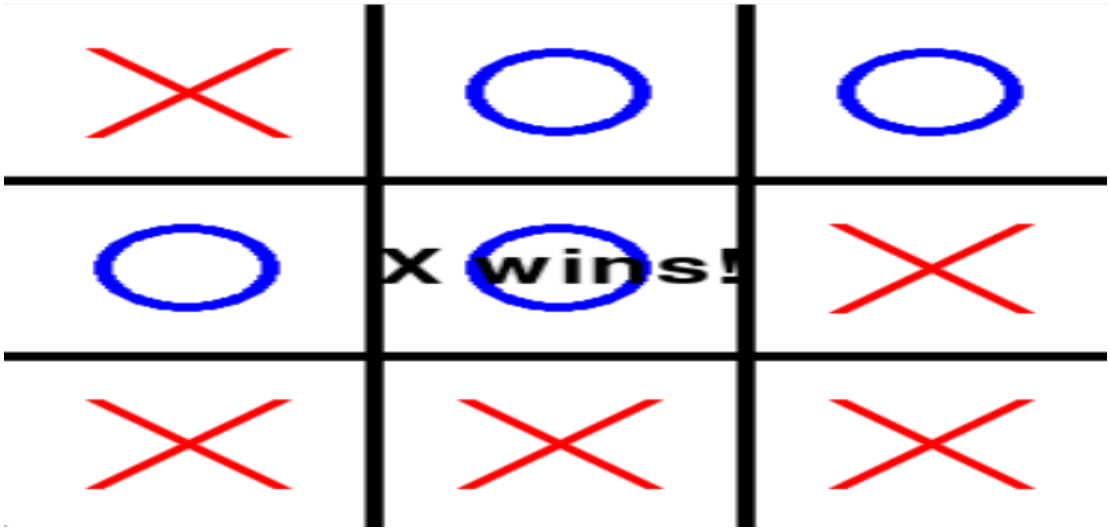

Fig 8.2 X wins

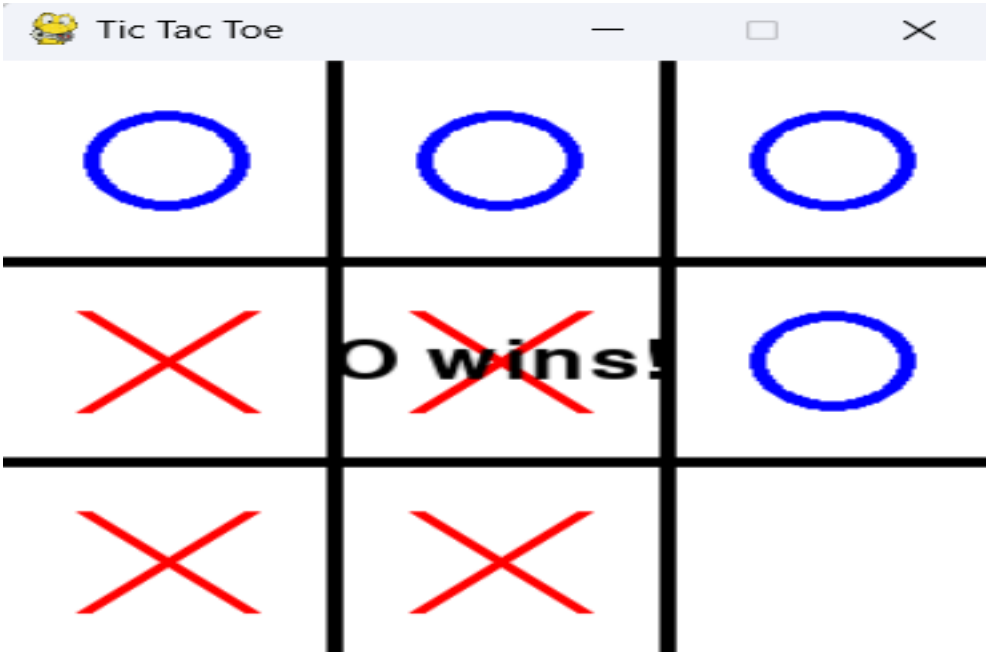

Fig 8.3 O Wins

7. REFERENCES

1. "Reinforcement Learning and Tic-Tac-Toe" by Gerald Tesauro - This paper explores using reinforcement learning techniques to train a computer to play Tic Tac Toe.

2. "Introduction to Artificial Intelligence" by Philip C. Jackson - This textbook may include a section on Tic Tac Toe as a simple example of game playing in AI.

3. "Artificial Intelligence: Foundations of Computational Agents" by David L. Poole and Alan K. Mackworth - This book may also cover Tic Tac Toe as an introductory example in the context of AI.

4. "Game Programming Patterns" by Robert Nystrom - This book could provide insights into designing and implementing Tic Tac Toe and other games.

5. "Designing Board Games: Tic-Tac-Toe" by Joseph A. DiVanna - This resource may focus specifically on the design aspects of Tic Tac Toe.

6. "Learning Python" by Mark Lutz - Many introductory Python programming books include a chapter or section on creating a Tic Tac Toe game as a beginner project.

7. "Introduction to Game Design, Prototyping, and Development: From Concept to Playable Game with Unity and C#" by Jeremy Gibson Bond - This book might cover Tic Tac Toe as a simple game example in the context of game design.

8. "Games, Puzzles, and Computation" by Robert A. Hearn - This book could offer a theoretical perspective on Tic Tac Toe and its computational complexity.

9. "Artificial Intelligence: A Modern Approach" by Stuart Russell and Peter Norvig - This classic AI textbook may include a chapter on games like Tic Tac Toe, covering both theoretical and practical aspects.

10. "Theory of Fun for Game Design" by Raph Koster - While not specifically about Tic Tac Toe, this book delves into game design principles that could be applied to designing and analyzing Tic Tac Toe and similar game