Tic Tac Toe Game Using Pygame – Project Report :

Introduction :

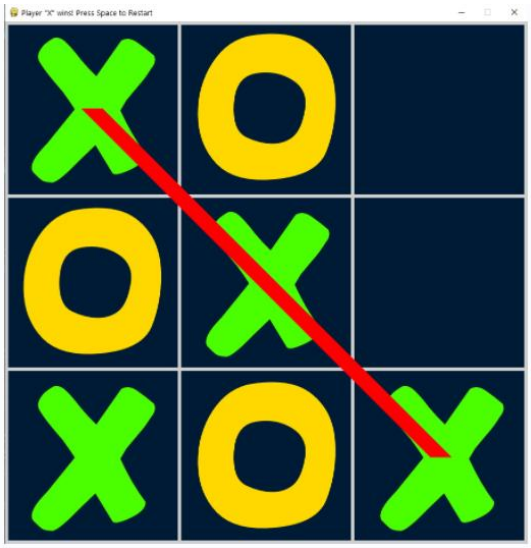
This project is a graphical implementation of the classic Tic Tac Toe game using Python and the Pygame library. The game allows two players to take turns placing their marks (X and O) on a 3x3 grid until one wins or the game ends in a draw.

Task 1: Game Window & Grid Setup

Description:

This task initializes the game window using Pygame and draws the 3x3 grid layout. It sets up the display size, colors, and background.

Screenshot:



Challenges & Solutions:

Challenge: Understanding how Pygame handles screen coordinates.

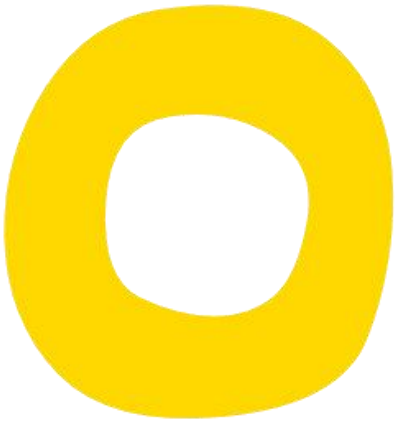
Solution: Used Pygame's built-in functions like `pygame.draw.line()` and carefully calculated cell boundaries for grid lines.

Task 2: Handling Player Input & Marking Cells

Description:

This part of the project handles player mouse clicks to mark cells with X or O. It alternates turns between two players.

Screenshot:



Challenges & Solutions:

Challenge: Preventing a player from overwriting an already marked cell.

Solution: Implemented a condition to check if a cell is empty before allowing a move.

Task 3: Checking for Win or Draw

Description:

This task checks the game board after each move to determine if a player has won or if the game is a draw.

Challenges & Solutions:

Challenge: Handling all possible win conditions (rows, columns, diagonals).

Solution: Created a function to loop through all win condition combinations and check for identical non-empty symbols.

Task 4: Game Reset Mechanism

Description:

After a game ends, this task enables the player to reset the board and start a new game using a mouse click or a keypress.

Conclusion :

Developing the Tic Tac Toe game using Pygame helped reinforce key programming concepts, including:

GUI development using Pygame

Event-driven programming

Game state management

Conditional logic and grid-based layout

Main Challenge:

The most significant challenge was managing player input correctly and syncing it with the graphical board

updates. Through testing and code refactoring, the input handling became more robust and responsive.