# Basics of Data Structures and Algorithms – PETV70

(SUMMER TRAINING PROGRAM)

## TOPIC: Tic-Tac-Toe Game

**LINK: https://tic-tac-toe-dsa.netlify.app/**


Submitted by:

Name: Ambika Raj

Registration number: 12311532

Section: 9PV17


Submitted to:

Sudha Shanker Prasad / Harjeet Kaur



**LOVELY PROFESSIONAL UNIVERSITY**

# ABSTRACT

This project is a creative and interactive implementation of the classic Tic-Tac-Toe game, developed using HTML, CSS, and JavaScript. The core logic utilizes Data Structures and Algorithms (DSA) concepts such as arrays, stack, and conditional logic.

Key features include undo functionality using a stack, player vs computer mode, winning cell highlights, confetti on victory, fun taunt messages, and a responsive UI. The project demonstrates how DSA can be applied in real-world applications like game development while ensuring an engaging user experience.

The project also focuses on clean UI/UX design, logical state handling, and creative enhancements, making it both technically sound and visually appealing.

# OBJECTIVES

- To apply DSA concepts in a real-world mini project
- To develop a functional and interactive web game
- To implement stack-based undo logic
- To practice array manipulation and condition checks
- To enhance UI with user feedback and engagement

# TECHNOLOGIES USED

| Tool / Technology | Purpose / Use Case |
|---|---|
| HTML | Used to structure the layout of the Tic-Tac-Toe game board, buttons, and interface elements. |
| CSS | Responsible for styling the game — including the neon theme, hover effects, button designs, and layout responsiveness. |
| JavaScript | Implements game logic: handling player turns, win detection, game state, undo functionality, and user interaction. |
| DSA Concepts | Applied stack (for undo), arrays (for board), pattern-matching (for win detection), and condition-based state logic. |

# DSA CONCEPTS APPLIED

1. Array (board representation)
    - Used for: Representing the 3x3 Tic-Tac-Toe board using a simple 1D array of 9 elements.
    - Each index (0 to 8) represents one of the 9 cells on the grid.
    - Players moves are stored in the array (X or O), and this array is used to check winning conditions and manage game state.

2. Stack (for undo features)
    - Used for: Implementing Undo functionality using the Last In First Out (LIFO) principle — a perfect example of stack usage in real applications.
    - Before each player move (in computer mode), the current board state is saved.
    - When the user clicks "Undo", the most recent state is popped and restored.
    - Undo is limited to one time per game (to add game balance and reinforce stack concept clearly).

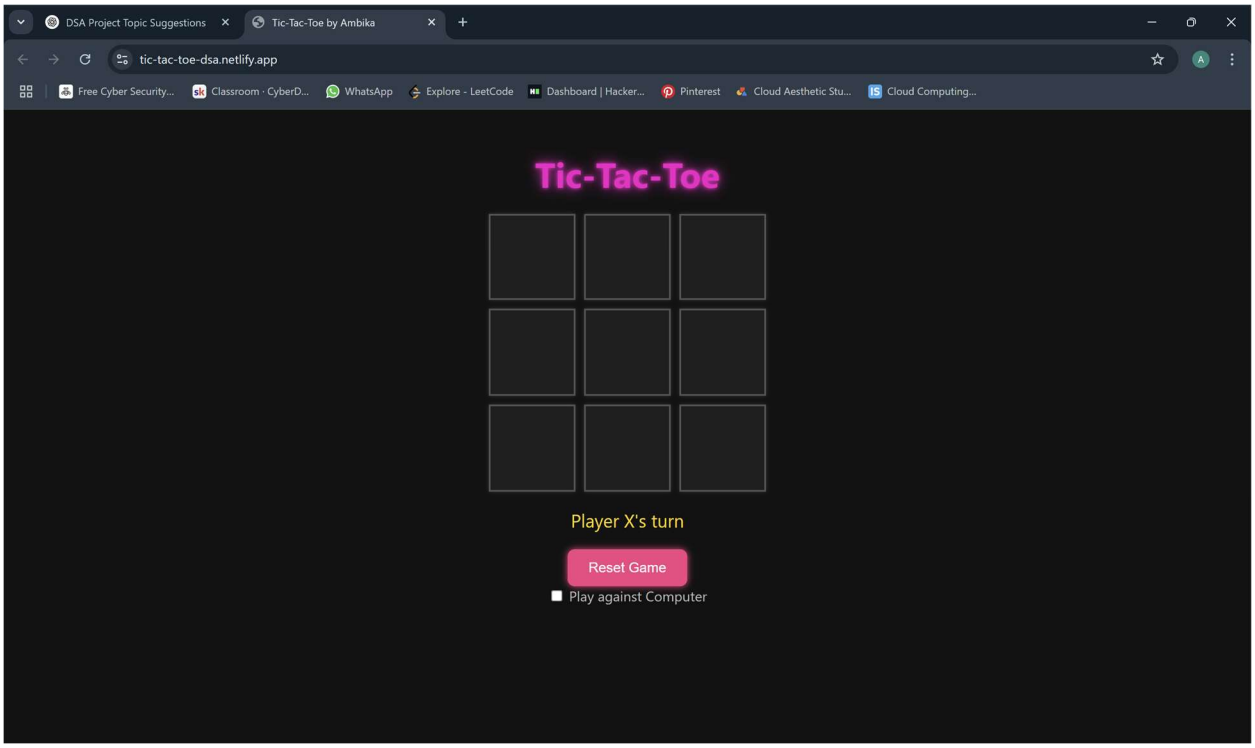
3. Map (for taunt memory)
    - Used for: Handling flexible key value data for taunt memory.
    - Stores the last taunt shown to avoid repeating it immediately.

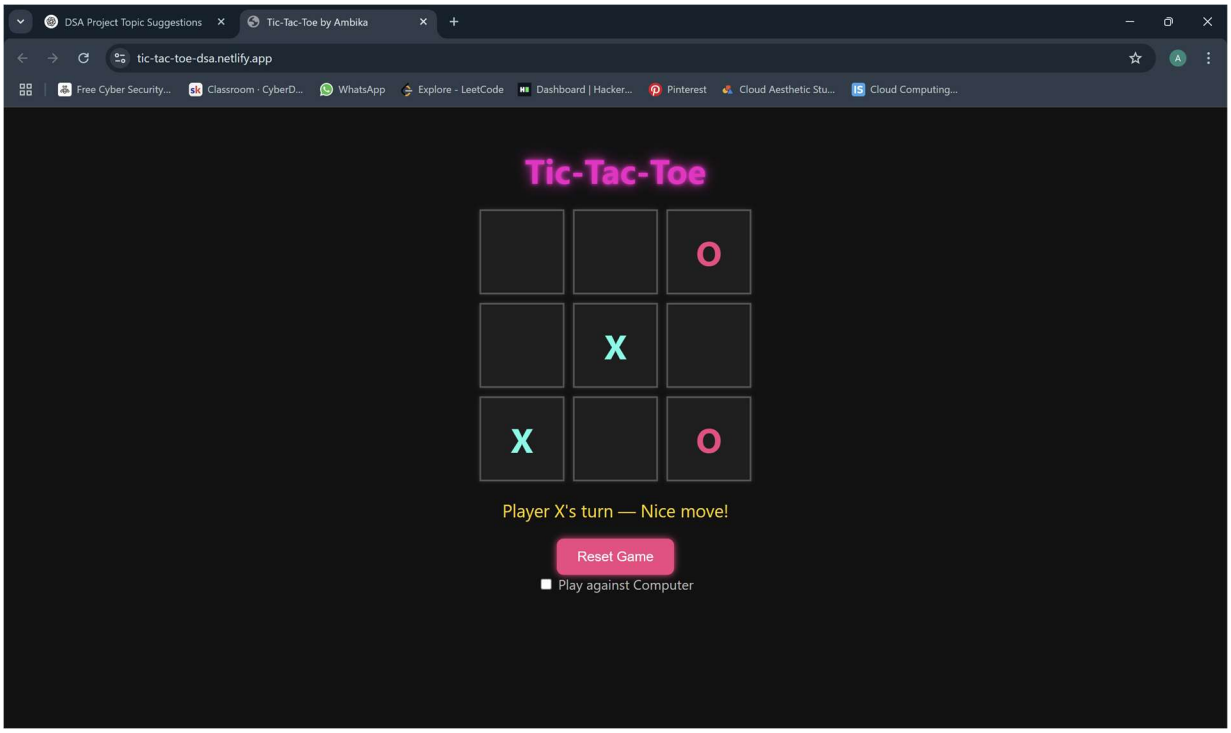
4. Pattern Matching (win logic)
    - Used for: Checking if a player has won by comparing the board against predefined win index combinations.
    - Win conditions are stored in a static array, each move triggers a check through this array to determine if any condition is satisfied by the current player's symbols.

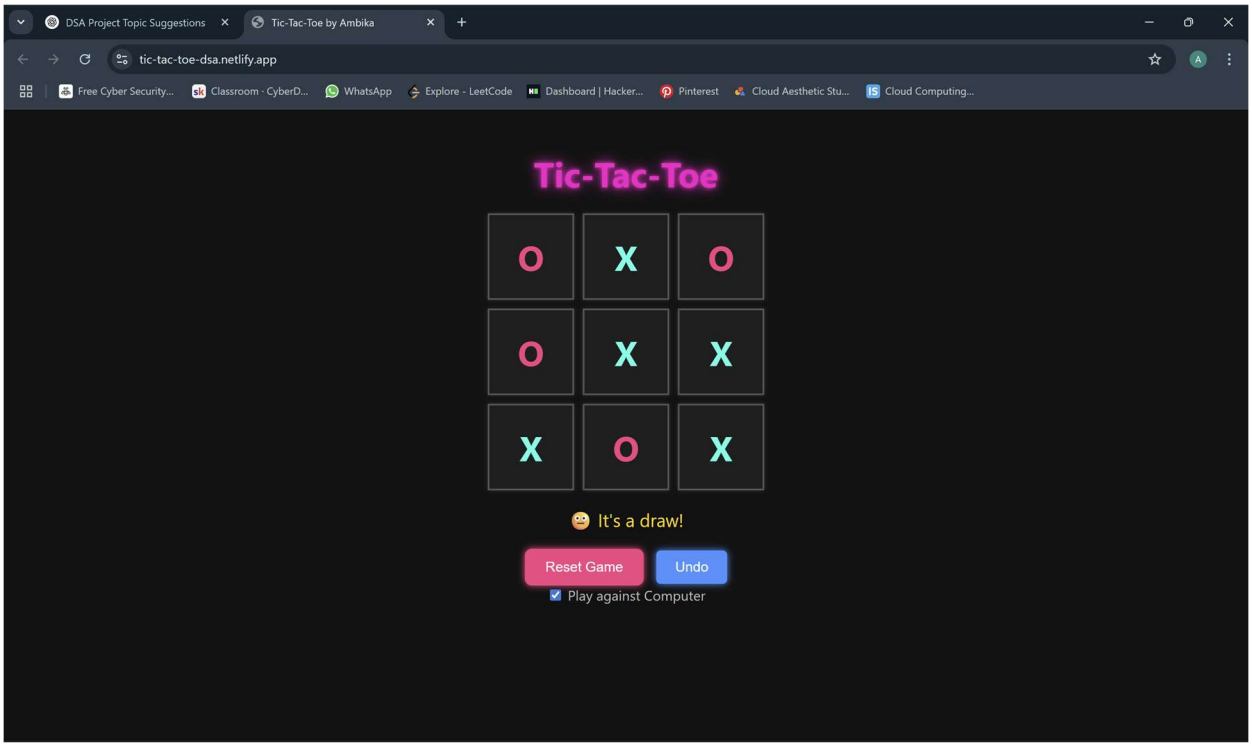
# KEY FEATURES

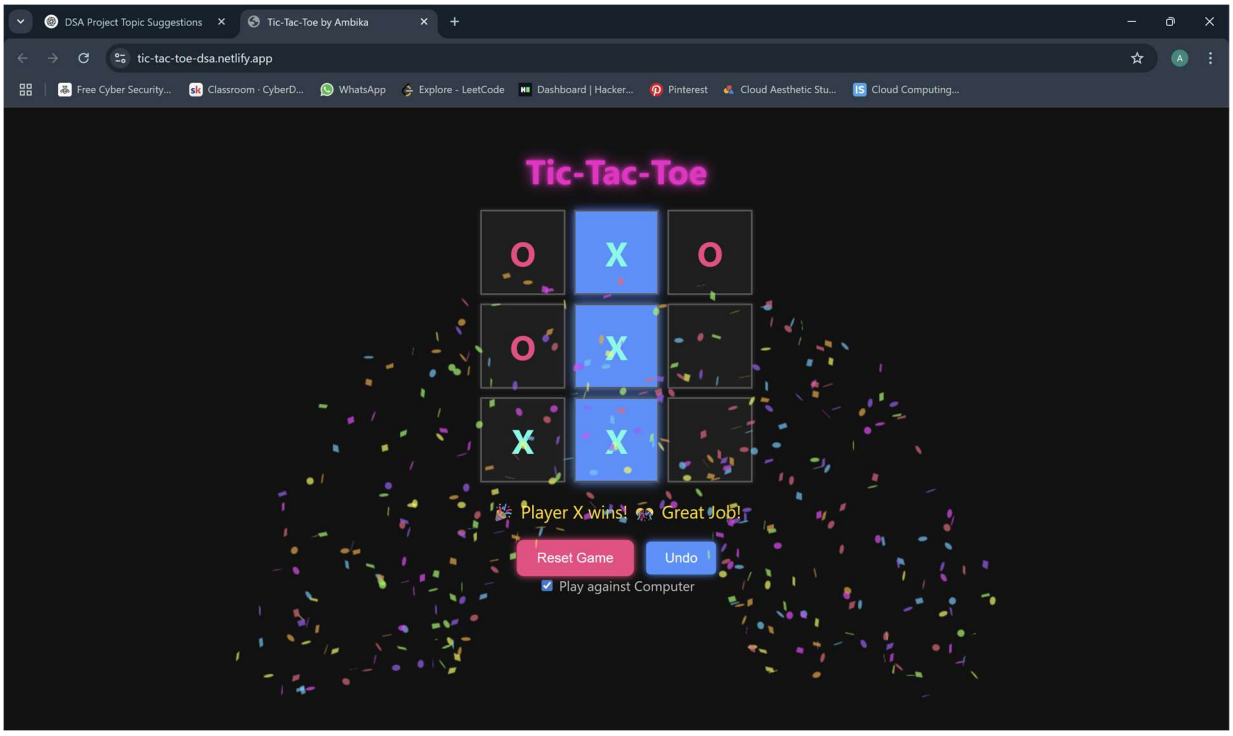
- Player vs Player and Player vs Computer modes
- Undo move option (for Player vs Computer only)
- Confetti animation on win
- Taunt messages after each move
- Highlighted winning cells
- Reset button to reset the game
- Responsive and interactive cell grid

# SCREENSHOTS




DSA Project Topic Suggestions
Tic-Tac-Toe by Ambika
tic-tac-toe-dsa.netlify.app
Free Cyber Security...
Classroom - CyberD...
WhatsApp
Explore - LeetCode
Dashboard | Hacker...
Pinterest
Cloud Aesthetic Stu...
Cloud Computing...
Tic-Tac-Toe
O X O
O X X
X O X
It's a draw!
Reset Game
Undo
Play against Computer


DSA Project Topic Suggestions
Tic-Tac-Toe by Ambika
tic-tac-toe-dsa.netlify.app
Free Cyber Security...
Classroom - CyberD...
WhatsApp
Explore - LeetCode
Dashboard | Hacker...
Pinterest
Cloud Aesthetic Stu...
Cloud Computing...
Tic-Tac-Toe
O X O
O X
X X
Player X wins! Great Job!
Reset Game
Undo
Play against Computer

# OUTCOME & LEARNING

- Learned how to apply DSA practically through stack and array usage.
- Improved JavaScript programming and DOM manipulation.
- Understood the role of UX/UI in engagement.
- Understood how to deploy projects and share via live links.

# FUTURE ENHANCEMENTS

- Add minimax algorithm for smarter computer opponent
- Add leaderboard using local storage or backend
- Allow symbol customization
- Add move animation or sound effects

# CONCLUSION

This project served as an excellent learning experience in combining theoretical DSA knowledge with practical implementation. By using arrays and stack effectively, and enhancing user experience with visual elements, the project bridges the gap between logic and creativity.

# PROJECT LINK

Link: https://tic-tac-toe-dsa.netlify.app/