TIC-TAC-TOE GAME

A PROJECT REPORT

Submitted by

YUVARAJ P (2303811710421188)

in partial fulfillment of requirements for the award of the course

CGB1201 - JAVA PROGRAMMING

In

COMPUTER SCIENCE AND ENGINEERING

K. RAMAKRISHNAN COLLEGE OF TECHNOLOGY

(An Autonomous Institution, affiliated to Anna University Chennai and Approved by AICTE, New Delhi)

SAMAYAPURAM – 621 112

NOVEMBER- 2024

K. RAMAKRISHNAN COLLEGE OF TECHNOLOGY (AUTONOMOUS)

SAMAYAPURAM – 621 112

BONAFIDE CERTIFICATE

Certified that this project report on **"TIC-TAC-TOE GAME"** is the bonafide work of **YUVARAJ P (2303811710421188)** who carried out the project work during the academic year 2024 - 2025 under my supervision.

CGB1201-JAVA PROGRAMMING
Dr.A.DELPHIN CAROLINA RANI M.E.,Ph.D.,
HEAD OF THE DEPARTMENT
PROFESSOR

CGB1201-JAVA PROGRAMMING
Mr.MALARMANNAN A, M.E.,
SUPERVISOR
ASSISTANTP ROFESSOR

SIGNATURE	**SIGNATURE**
Mrs.A.Delphin Carolina Rani, M.E.,Ph.D.,	Mr. A. Malarmannan, M.E.,
HEAD OF THE DEPARTMENT	**SUPERVISOR**
PROFESSOR	ASSISTANT PROFESSOR
Department of CSE	Department of CSE
K.Ramakrishnan College of Technology (Autonomous)	K.Ramakrishnan College of Technology (Autonomous)
Samayapuram–621112.	Samayapuram–621112.

Submitted for the viva-voce examination held on 06-12-2024

CGB1201-JAVA PROGRAMMING
Mr. M. SARAVANAN, M.E.,
INTERNAL EXAMINER
ASSISTANT PROFESSOR

CGB1201-JAVA PROGRAMMING
Mr.R. KARTHIK, M.E.,
EXTERNAL EXAMINER
ASSISTANT PROFESSOR
8138-SCE, TRICHY.

06/12/2024

INTERNAL EXAMINER EXTERNAL EXAMINER

DECLARATION

I declare that the project report on **"TIC-TAC-TOE GAME"** is the result of original work done by us and best of our knowledge, similar work has not been submitted to **"ANNA UNIVERSITY CHENNAI"** for the requirement of Degree of **BACHELOR OF ENGINEERING**. This project report is submitted on the partial fulfilment of the requirement of the completion of the course **CGB1201 - JAVA PROGRAMMING.**

.

Signature

P. Yuvaraj

YUVARAJ P

Place: Samayapuram

Date: 06-12-2024

ACKNOWLEDGEMENT

It is with great pride that I express our gratitude and in-debt to our institution "**K.Ramakrishnan College of Technology (Autonomous)**", for providing us with the opportunity to do this project.

I glad to credit honourable chairman **Dr. K. RAMAKRISHNAN**, **B.E.,** for having provided for the facilities during the course of our study in college.

I would like to express our sincere thanks to our beloved Executive Director **Dr. S. KUPPUSAMY, MBA, Ph.D.,** for forwarding to our project and offering adequate duration in completing our project.

I would like to thank **Dr. N. VASUDEVAN, M.Tech., Ph.D.,** Principal, who gave opportunity to frame the project the full satisfaction.

I whole heartily thanks to **Dr. A. DELPHIN CAROLINA RANI, M.E.,Ph.D.,** Head of the department, **COMPUTER SCIENCE AND ENGINEERING** for providing her encourage pursuing this project.

I express our deep expression and sincere gratitude to our project supervisor **MR. A. MALARMANNAN, M.E.,** Department of **COMPUTER SCIENCE AND ENGINEERING,** for his incalculable suggestions, creativity, assistance and patience which motivated us to carry out this project.

I render our sincere thanks to Course Coordinator and other staff members for providing valuable information during the course.

I wish to express our special thanks to the officials and Lab Technicians of our departments who rendered their help during the period of the work progress.

VISION OF THE INSTITUTION

To serve the society by offering top-notch technical education on par with global standards

MISSION OF THE INSTITUTION

- Be a center of excellence for technical education in emerging technologies by exceeding the needs of the industry and society.
- Be an institute with world class research facilities
- Be an institute nurturing talent and enhancing the competency of students to transform them as all-round personality respecting moral and ethical values

VISION OF DEPARTMENT

To be a center of eminence in creating competent software professionals with research and innovative skills.

MISSION OF DEPARTMENT

M1: Industry Specific: To nurture students in working with various hardware and software platforms inclined with the best practices of industry.

M2: Research: To prepare students for research-oriented activities.

M3: Society: To empower students with the required skills to solve complex technological problems of society.

PROGRAM EDUCATIONAL OBJECTIVES

1. PEO1: Domain Knowledge

To produce graduates who have strong foundation of knowledge and skills in the field of Computer Science and Engineering.

2. PEO2: Employability Skills and Research

To produce graduates who are employable in industries/public sector/research organizations or work as an entrepreneur.

3. PEO3: Ethics and Values

To develop leadership skills and ethically collaborate with society to tackle real-world challenges.

PROGRAM SPECIFIC OUTCOMES (PSOs)

PSO 1: Domain Knowledge

To analyze, design and develop computing solutions by applying foundational concepts of Computer Science and Engineering.

PSO 2: Quality Software

To apply software engineering principles and practices for developing quality software for scientific and business applications.

PSO 3: Innovation Ideas

To adapt to emerging Information and Communication Technologies (ICT) to innovate ideas and solutions to existing/novel problems

PROGRAM OUTCOMES (POs)

Engineering students will be able to:

1. **Engineering knowledge:** Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
2. **Problem analysis:** Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences
3. **Design/development of solutions:** Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations
4. **Conduct investigations of complex problems:** Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions

5. **Modern tool usage:** Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations
6. **The engineer and society:** Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice
7. **Environment and sustainability:** Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development
8. **Ethics:** Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.
9. **Individual and team work:** Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.
10. **Communication:** Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.
11. **Project management and finance:** Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.
12. **Life-long learning:** Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

ABSTRACT

The Tic Tac Toe Game is a logic-driven application designed to deliver an engaging and interactive gaming experience. Built on the foundation of Java programming, the project employs Java Swing and AWT to create a robust graphical user interface (GUI) that ensures smooth and responsive gameplay. The game features two primary modes: Player vs Player (PvP), where two users alternate turns to strategize their moves, and Player vs Computer (PvC), which incorporates an AI opponent with randomized move logic. These features make the game accessible and enjoyable for a diverse range of users, encouraging both strategy and engagement.

This project emphasizes real-time interactions and dynamic feedback. The intuitive GUI provides players with clear and immediate updates on game states, such as turns, wins, or ties, ensuring a seamless flow throughout. The dynamic scoreboard enhances competitiveness by tracking player performance after each round. In the PvC mode, the AI opponent generates randomized moves, adding unpredictability and challenge to single-player gameplay. The combination of real-time feedback, randomized logic, and smooth interactions makes the game highly replayable and mentally stimulating.

Overall, the Tic Tac Toe Game successfully modernizes the traditional paper-and-pencil game into a digital format, offering accessibility and enjoyment across various age groups. The system highlights the use of object-oriented principles in Java, ensuring a modular and scalable design. With its focus on engagement, usability, and innovation, this project provides a fun, thought-provoking, and competitive gaming experience, demonstrating how classic games can be reimagined for the digital age.

ABSTRACT WITH POs AND PSOs MAPPING

CO 5 : BUILD JAVA APPLICATIONS FOR SOLVING REAL-TIME PROBLEMS.

ABSTRACT	POs MAPPED	PSOs MAPPED
The Tic Tac Toe Game is a logic-based application developed using Java Swing and AWT to provide an interactive gameplay experience. The game includes two modes: Player vs Player (PvP), where two users alternate turns on a shared interface, and Player vs Computer (PvC), where a single player competes against an AI opponent with randomized moves.The project features a responsive and intuitive GUI, with dynamic score tracking and real-time feedback on game states such as turns, wins, and ties. The PvC mode includes a randomized AI to ensure challenge and unpredictability, making the game more engaging and replayable for players of all ages.This project demonstrates the application of object-oriented programming principles in Java, offering a fun and accessible experience while maintaining smooth gameplay. The Tic Tac Toe Game successfully modernizes a classic game for the digital age, enhancing its accessibility and enjoyment.	**PO1 -3** **PO2 -3** **PO3 -3** **PO4 -3** **PO5 -3** **PO6 -3** **PO7 -3** **PO8 -3** **PO9 -3** **PO10 -3** **PO11-3** **PO12 -3**	**PSO1 -3** **PSO2 -3** **PSO3 -3**

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

1.1Objective

The primary objective of the Tic Tac Toe Game project is to develop an engaging and interactive platform that transforms the classic pencil-and-paper game into a digital experience. The system incorporates two gameplay modes: Player vs Player (PvP), where two users alternate turns, and Player vs Computer (PvC), where a single player competes against an AI opponent. The PvC mode uses randomized move logic, ensuring unpredictability and providing a challenging experience for the player. The aim is to create a fun and strategic game environment that appeals to users of all ages.

The game is built using Java Swing and AWT, enabling a visually appealing and responsive graphical user interface (GUI). The 3x3 grid board allows players to seamlessly take turns placing their symbols (X or O), while the application dynamically tracks and validates game states, including turns, wins, and ties. A built-in dynamic scoreboard further enhances engagement by tracking player performance over multiple rounds. By emphasizing real-time updates and smooth gameplay, this project delivers a user-friendly solution that promotes critical thinking, decision-making, and competitive play.

Ultimately, this project aims to modernize the classic Tic Tac Toe game by blending simplicity with innovation. The system’s focus on intuitive design, strategic depth, and engaging gameplay ensures a memorable experience for both casual players and enthusiasts. The project also demonstrates the power of Java’s object-oriented programming principles, emphasizing modularity, reusability, and scalability.

1.2Overview

The Tic Tac Toe Game is a strategic application developed using Java Swing and AWT to provide an intuitive graphical interface and smooth gameplay. The game features two modes: Player vs Player, allowing two users to alternate turns, and Player vs Computer, where the AI uses randomized moves to provide a competitive experience. The platform ensures fairness through comprehensive game logic that validates moves, detects wins or ties, and alternates turns dynamically. With its interactive design, the system tracks player performance using a real-time scoreboard. The project aims to enhance the traditional game with a modern digital approach, making it accessible, engaging, and entertaining.

1.3 Java Programming Concepts

This project leverages key Java programming concepts to ensure a robust and efficient implementation.

- **Object-Oriented Programming (OOP):** Encapsulates game logic within cohesive classes.
- **Inheritance:** Extends JFrame for GUI creation.
- **Event Handling:** Implements ActionListener for button clicks and game logic.
- **AWT Framework:** Uses components like JButton, JPanel, and JLabel for GUI elements.
- **Control Statements:** Utilizes if-else and loops for game flow and decision-making.
- **Exception Handling:** Validates user inputs to prevent runtime errors.
- **Polymorphism:** Overrides actionPerformed for dynamic event handling.
- **String Manipulation**: Processes user input such as names and symbols.

  By integrating these concepts, the project ensures a smooth and modular design, offering extensibility and reliability.

CHAPTER 2
PROJECT METHODOLOGY

2.1 Proposed Work

The Tic Tac Toe Game project follows a systematic approach to ensure the creation of an engaging, functional, and user-friendly application. The development process begins with requirement analysis, where key features such as the game modes (Player vs Player and Player vs Computer), game logic (move validation and win conditions), randomized AI moves for the computer, and a scoreboard to track player performance are defined.

Once the features are established, the next step is system design, where the game is divided into separate modules to improve organization and ease of maintenance.

These modules include:

Player Input Module: Captures player names, symbols, and game mode (PvP or PvC).

Game Board Module: Handles the interactive 3x3 grid, updating based on player and computer moves.

AI Logic Module: Ensures the computer generates random moves in PvC mode.

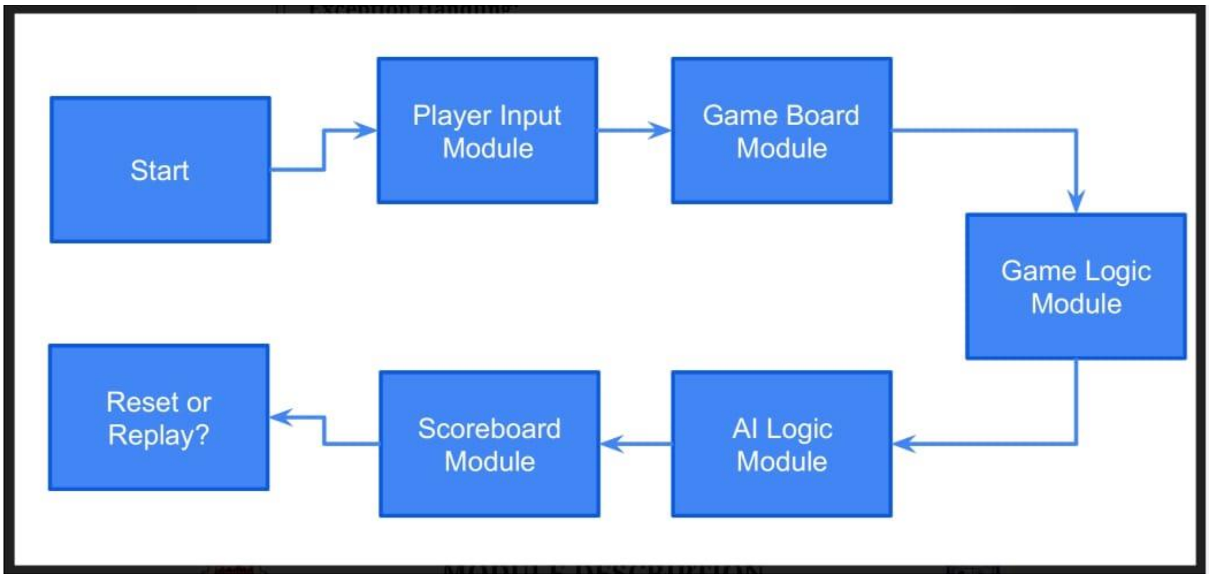
Scoreboard Module: Tracks and displays real-time player scores.

The implementation phase involves writing the code for these modules and ensuring they function seamlessly together. In the Player Input Module, players enter their names, select symbols, and choose the game mode. The Game Board Module creates the interactive grid, while the AI Logic Module generates computer moves randomly. The Scoreboard Module dynamically updates the scores as the game progresses.

Finally, testing and validation ensure that each module performs correctly. Unit testing is performed to validate individual modules, and integration testing ensures that the components work together effectively. The focus during testing is on usability, ensuring the game is intuitive, fair, and engaging for users.

Overall, the project emphasizes usability, fairness, and engagement, ensuring the game is fun and challenging. The game design allows flexibility for future enhancements, such as adding difficulty levels for AI or expanding the grid size for a more complex version of the game.

2.2 Block Diagram



CHAPTER 3
MODULE DESCRIPTION

3.1 Module 1 Player Input Module

This module captures player names, gameplay mode, and preferred symbols (X or O) before starting the game. Players can select either Player vs Player mode or challenge the computer in Player vs Computer mode. The module ensures a smooth user experience by validating inputs and dynamically updating the scoreboard with the entered names. The objective is to simplify the setup process, enabling players to start their game without delays.

3.2 Module 2 Game Board Module

The Game Board Module serves as the visual interface for the Tic Tac Toe game. It presents a 3x3 grid of buttons where players or the computer can make their moves. The board dynamically updates its cells with the respective symbols (X or O) after each turn. This module ensures fairness by validating move legality and preventing overwriting of cells. It also highlights the game state in real-time, such as ongoing turns, wins, or ties, enhancing the overall gameplay experience.

3.3 Module 3 Game Logic Module

The Game Logic Module forms the core of the application, governing rules and gameplay flow. It alternates turns between players or the computer, validates win conditions (three consecutive symbols), and detects ties when all cells are filled. The module integrates seamlessly with the game board, ensuring immediate feedback after each move. Its robust logic prevents illegal actions, making the game fair and enjoyable.

3.4 Module 4 AI Logic Module

The AI Logic Module is responsible for the computer's gameplay in Player vs Computer mode. It uses a randomization algorithm to select an empty cell for its moves, ensuring unpredictability and a challenging experience for the player. This module integrates with the game board and logic components to maintain fairness and smooth turn transitions. The objective is to provide a realistic single-player mode with engaging and varied gameplay.

3.5 Module 5 Scoreboard Module

The Scoreboard Module tracks and displays the performance of both players. After each round, it updates scores dynamically based on the game outcome (win or tie). The scoreboard ensures transparency by showing real-time results at the bottom of the interface. It serves as a motivational feature for players, encouraging competitive and strategic gameplay.

CHAPTER 4
CONCLUSION & FUTURE SCOPE

4.1 CONCLUSION

The Tic Tac Toe Game project has been successfully developed and implemented as a logic-based, interactive application using Java Swing and AWT, two powerful tools that allow the creation of a responsive graphical user interface (GUI). The game provides a highly engaging experience through its two primary gameplay modes: Player vs Player (PvP) and Player vs Computer (PvC). The PvC mode is made more interesting with randomized AI moves, offering players a challenging opponent, ensuring that no two games are alike. Moreover, the game includes real-time score tracking, allowing players to see their performance throughout the game, fostering a competitive environment.

By utilizing Object-Oriented Programming (OOP) principles such as encapsulation, inheritance, and polymorphism, the project delivers a robust and modular solution. The game logic and graphical components are separated into distinct modules, ensuring a clean and organized structure. Efficient event handling through the use of ActionListener allows for smooth gameplay, with quick response times to user actions. The game is designed with fairness in mind, ensuring that both players (or the player and AI) have equal opportunities to win, making it an enjoyable and balanced experience for users of all ages.

This project effectively modernizes the classic Tic Tac Toe game by adding new layers of interactivity, complexity, and engagement while maintaining the simplicity that makes it accessible. It serves as an example of how traditional games can be enhanced with modern technologies to create fun, user-centric applications. The focus on usability, accessibility, and engagement has allowed this project to meet the primary goal of offering a delightful experience, while also paving the way for future enhancements to further improve and expand the gameplay experience.

4.2 FUTURE SCOPE

While the Tic Tac Toe Game project successfully meets its objectives, there are numerous future enhancements that can be implemented to expand the game's functionality and improve the overall experience for players. One of the most significant improvements would be to enhance the AI by introducing difficulty levels such as easy, medium, and hard. This would provide players with the option to select the level of challenge they want, ensuring a more personalized experience. By fine-tuning the AI logic to simulate varying levels of difficulty, the game could appeal to a broader range of players, from beginners to seasoned strategists.

In addition, expanding the game beyond the classic 3x3 grid could increase the complexity and provide new strategic challenges. Larger grid sizes, such as 4x4 or 5x5, would add depth to the gameplay, forcing players to think in new ways and plan ahead. This would make the game more interesting and challenging, especially in the Player vs Player mode.

The game could also be made more accessible by introducing multi-platform support. Developing web-based and mobile versions of the game would allow players to enjoy the game on a variety of devices, ensuring that it is available anytime and anywhere. This would significantly increase the reach of the game, making it more accessible to a wider audience.

Further improvements could involve saving scores, maintaining leaderboards, and integrating multilingual support. The ability to track progress over time would motivate players to improve their skills and compete for high scores. Additionally, implementing a leaderboard could add a competitive aspect to the game, allowing players to compare their performance with others. Introducing multilingual support would make the game more inclusive, enabling players from different linguistic backgrounds to enjoy it.

Ultimately, these enhancements would make the game more versatile, accessible, and appealing to a global audience, providing a more comprehensive gaming experience. The ability to continually improve and evolve the game is a key advantage of this project, ensuring its relevance and longevity in the ever-growing gaming market.

APPENDIX A
(SOURCE CODE)

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;

public class TicTacToeWithInteractiveScoreboard extends JFrame implements
ActionListener {
    private JTextField player1Field, player2Field;
    private JComboBox<String> modeBox, symbolBox;
    private JButton[][] buttons = new JButton[3][3];
    private JLabel statusLabel, player1ScoreLabel, player2ScoreLabel;
    private JButton resetButton, replayButton;
    private String player1 = "Player 1", player2 = "Player 2";
    private char currentSymbol = 'X';
    private boolean isPlayerVsComputer = false;
    private String[][] board = new String[3][3];
    private int player1Score = 0, player2Score = 0;

    public TicTacToeWithInteractiveScoreboard() {
        setTitle("Tic Tac Toe with Scoreboard");
        setSize(700, 800);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setLayout(new BorderLayout(10, 10));

        // Top Panel: Player Names and Game Settings
        JPanel topPanel = new JPanel(new GridLayout(4, 2, 10, 10));
        topPanel.setBorder(BorderFactory.createTitledBorder("Game Settings"));
        player1Field = new JTextField("Player 1");
        player2Field = new JTextField("Player 2");
        topPanel.add(new JLabel("Player 1 Name:"));
        topPanel.add(player1Field);
        topPanel.add(new JLabel("Player 2/Computer Name:"));
        topPanel.add(player2Field);

        modeBox = new JComboBox<>(new String[]{"Player vs Player", "Player vs
Computer"});
        symbolBox = new JComboBox<>(new String[]{"X", "O"});
        topPanel.add(new JLabel("Game Mode:"));
        topPanel.add(modeBox);
        topPanel.add(new JLabel("Choose Symbol:"));
        topPanel.add(symbolBox);

        add(topPanel, BorderLayout.NORTH);

        // Center Panel: Game Board
        JPanel boardPanel = new JPanel(new GridLayout(3, 3, 5, 5));
        boardPanel.setBorder(BorderFactory.createTitledBorder("Game Board"));
        for (int i = 0; i < 3; i++) {
            for (int j = 0; j < 3; j++) {
                buttons[i][j] = new JButton("");
                buttons[i][j].setFont(new Font("Arial", Font.BOLD, 40));
                buttons[i][j].addActionListener(this);
                boardPanel.add(buttons[i][j]);
                board[i][j] = "";
            }
        }
        add(boardPanel, BorderLayout.CENTER);

        // Bottom Panel: Status, Scores, and Controls
        JPanel bottomPanel = new JPanel(new GridLayout(3, 1, 10, 10));
        statusLabel = new JLabel("Enter details and start playing!", JLabel.CENTER);
        statusLabel.setFont(new Font("Arial", Font.BOLD, 16));
        bottomPanel.add(statusLabel);

        JPanel scorePanel = new JPanel(new GridLayout(1, 2, 10, 10));
```

```
    player1ScoreLabel = new JLabel(player1 + ": 0", JLabel.CENTER);
    player1ScoreLabel.setFont(new Font("Arial", Font.BOLD, 16));
    player2ScoreLabel = new JLabel(player2 + ": 0", JLabel.CENTER);
    player2ScoreLabel.setFont(new Font("Arial", Font.BOLD, 16));
    scorePanel.add(player1ScoreLabel);
    scorePanel.add(player2ScoreLabel);
    bottomPanel.add(scorePanel);

    JPanel controlPanel = new JPanel(new FlowLayout());
    resetButton = new JButton("Reset");
    replayButton = new JButton("Replay");
    resetButton.addActionListener(e -> resetGame());
    replayButton.addActionListener(e -> replayGame());
    controlPanel.add(resetButton);
    controlPanel.add(replayButton);
    bottomPanel.add(controlPanel);

    add(bottomPanel, BorderLayout.SOUTH);

    setVisible(true);
}

@Override
public void actionPerformed(ActionEvent e) {
    JButton clickedButton = (JButton) e.getSource();

    if (clickedButton.getText().equals("")) {
        clickedButton.setText(String.valueOf(currentSymbol));
        updateBoard(clickedButton);

        if (checkWin()) {
            String winner = (currentSymbol == 'X') ? player1 : player2;
            statusLabel.setText(winner + " (" + currentSymbol + ") wins!");
            updateScore(winner);
```

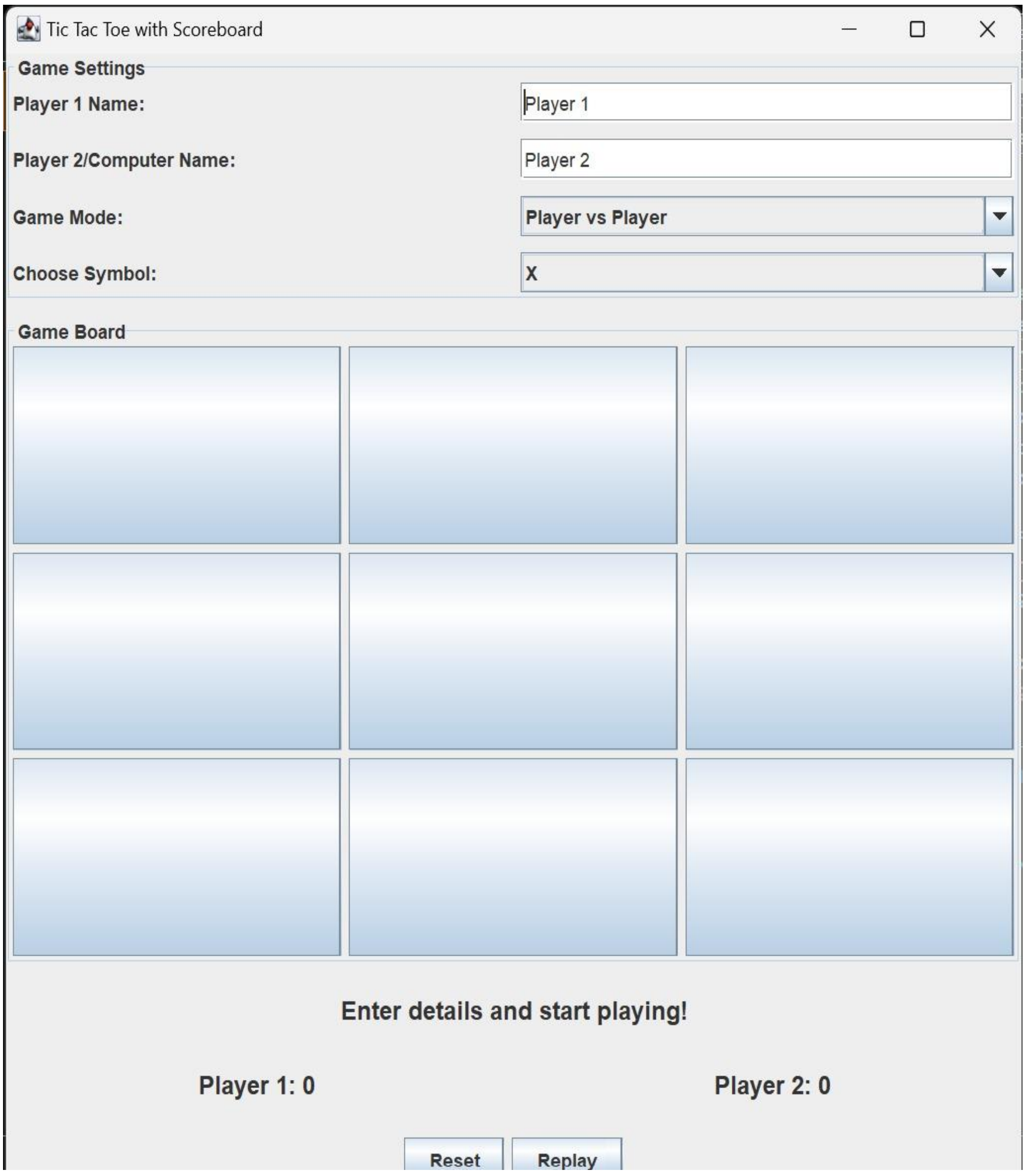
```
                disableButtons();
            } else if (isBoardFull()) {
                statusLabel.setText("It's a tie!");
                disableButtons();
            } else {
                switchTurn();
                if (isPlayerVsComputer && currentSymbol == 'O') {
                    computerMove();
                }
            }
        }
    }

    private void updateBoard(JButton clickedButton) {
        for (int i = 0; i < 3; i++) {
            for (int j = 0; j < 3; j++) {
                if (buttons[i][j] == clickedButton) {
                    board[i][j] = String.valueOf(currentSymbol);
                }
            }
        }
    }

    private void switchTurn() {
        currentSymbol = (currentSymbol == 'X') ? 'O' : 'X';
        statusLabel.setText((currentSymbol == 'X' ? player1 : player2) + "'s (" +
currentSymbol + ") turn.");
    }

    private void computerMove() {
        for (int i = 0; i < 3; i++) {
            for (int j = 0; j < 3; j++) {
                if (board[i][j].equals("")) {
                    buttons[i][j].setText("O");
                    board[i][j] = "O";
                    if (checkWin()) {
                        statusLabel.setText(player2 + " (O) wins!");
                        updateScore(player2);
                        disableButtons();
                    } else {
                        currentSymbol = 'X';
                        statusLabel.setText(player1 + "'s (X) turn.");
                    }
                    return;
                }
            }
        }
    }

    private void updateScore(String winner) {
        if (winner.equals(player1)) {
            player1Score++;
        } else if (winner.equals(player2)) {
            player2Score++;
        }
        player1ScoreLabel.setText(player1 + ": " + player1Score);
        player2ScoreLabel.setText(player2 + ": " + player2Score);
    }

    private boolean checkWin() {
        for (int i = 0; i < 3; i++) {
            if (board[i][0].equals(board[i][1]) && board[i][1].equals(board[i][2]) &&
!board[i][0].equals("")) return true;
            if (board[0][i].equals(board[1][i]) && board[1][i].equals(board[2][i]) &&
!board[0][i].equals("")) return true;
        }
        if (board[0][0].equals(board[1][1]) && board[1][1].equals(board[2][2]) &&
!board[0][0].equals("")) return true;
```

```
        if (board[0][2].equals(board[1][1]) && board[1][1].equals(board[2][0]) &&
!board[0][2].equals("")) return true;
        return false;
    }

    private boolean isBoardFull() {
        for (int i = 0; i < 3; i++) {
            for (int j = 0; j < 3; j++) {
                if (board[i][j].equals("")) return false;
            }
        }
        return true;
    }

    private void disableButtons() {
        for (int i = 0; i < 3; i++) {
            for (int j = 0; j < 3; j++) {
                buttons[i][j].setEnabled(false);
            }
        }
    }

    private void resetGame() {
        player1Field.setText("Player 1");
        player2Field.setText("Player 2");
        modeBox.setSelectedIndex(0);
        symbolBox.setSelectedIndex(0);
        player1Score = 0;
        player2Score = 0;
        player1ScoreLabel.setText(player1 + ": 0");
        player2ScoreLabel.setText(player2 + ": 0");
        replayGame();
    }
    private void replayGame() {
        for (int i = 0; i < 3; i++) {
            for (int j = 0; j < 3; j++) {
                buttons[i][j].setText("");
                buttons[i][j].setEnabled(true);
                board[i][j] = "";
            }
        }
        player1 = player1Field.getText();
        player2 = player2Field.getText();
        isPlayerVsComputer = modeBox.getSelectedIndex() == 1;
        currentSymbol = symbolBox.getSelectedItem().toString().charAt(0);
        statusLabel.setText(player1 + "'s (" + currentSymbol + ") turn.");
    }

    public static void main(String[] args) {
        new TicTacToeWithInteractiveScoreboard();
    }
}
```

APPENDIX B

(SCREENSHOTS)

1) OUTPUT INTERFACE


Tic Tac Toe with Scoreboard

Game Settings

Player 1 Name: Player 1

Player 2/Computer Name: Player 2

Game Mode: Player vs Player

Choose Symbol: X

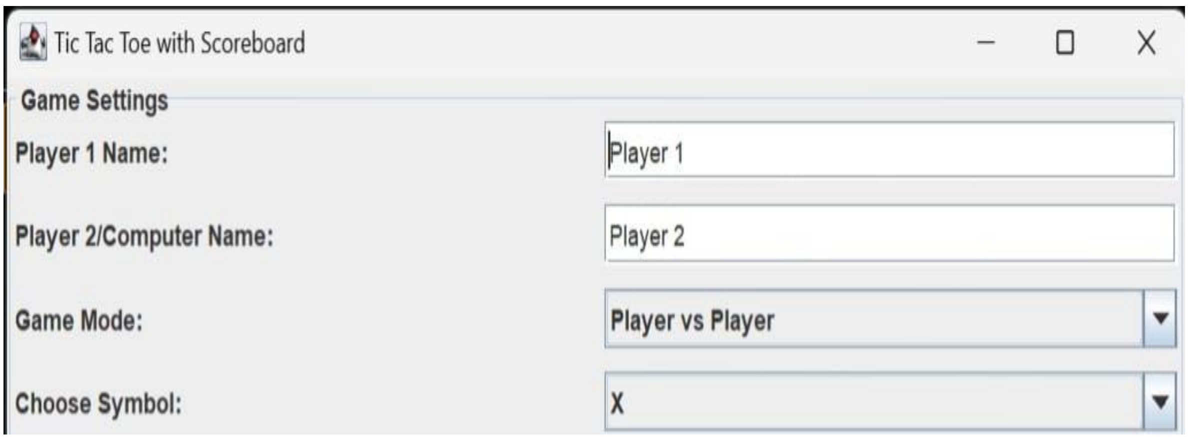
Game Board

Enter details and start playing!

Player 1: 0

Player 2: 0

2) ENTER GAMER MOOD SELECTION

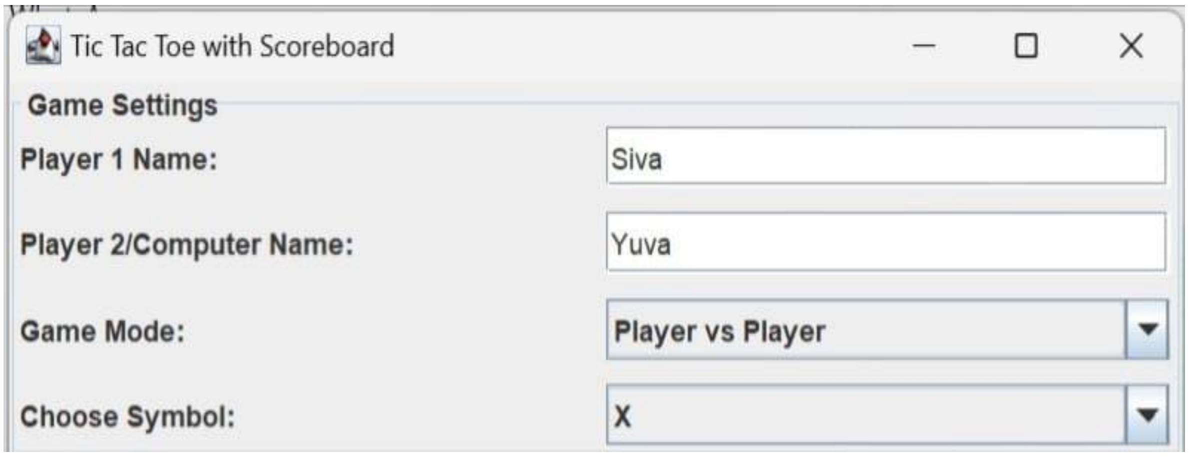


Player 1 Name: Player 1

Player 2/Computer Name: Player 2

Game Mode: Player vs Player

Choose Symbol: X



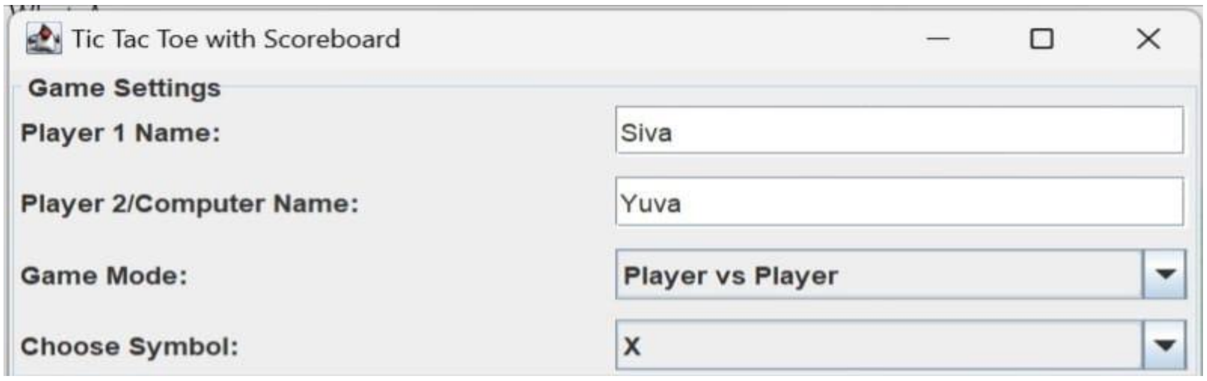
Player 1 Name: Siva

Player 2/Computer Name: Yuva

Game Mode: Player vs Player

Choose Symbol: X

3) PLAYER VS PLAYER



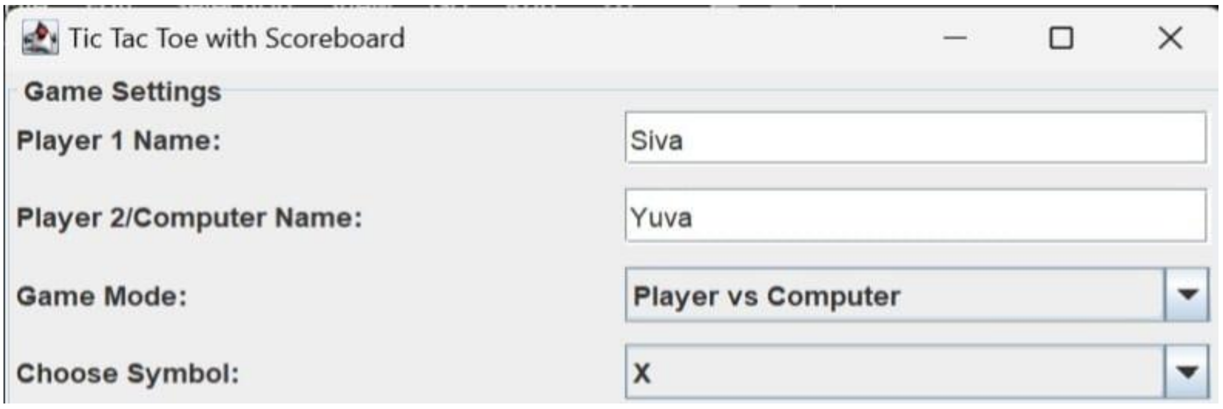
Player 1 Name: Siva

Player 2/Computer Name: Yuva

Game Mode: Player vs Player

Choose Symbol: X

4) PLAYER VS COMPUTER


Tic Tac Toe with Scoreboard

Game Settings

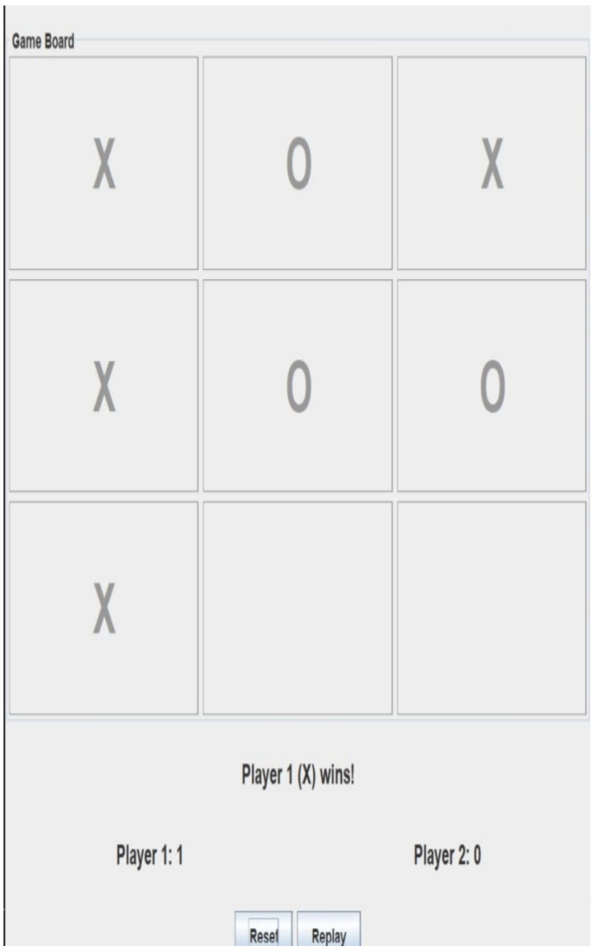
Player 1 Name: Siva

Player 2/Computer Name: Yuva

Game Mode: Player vs Computer

Choose Symbol: X

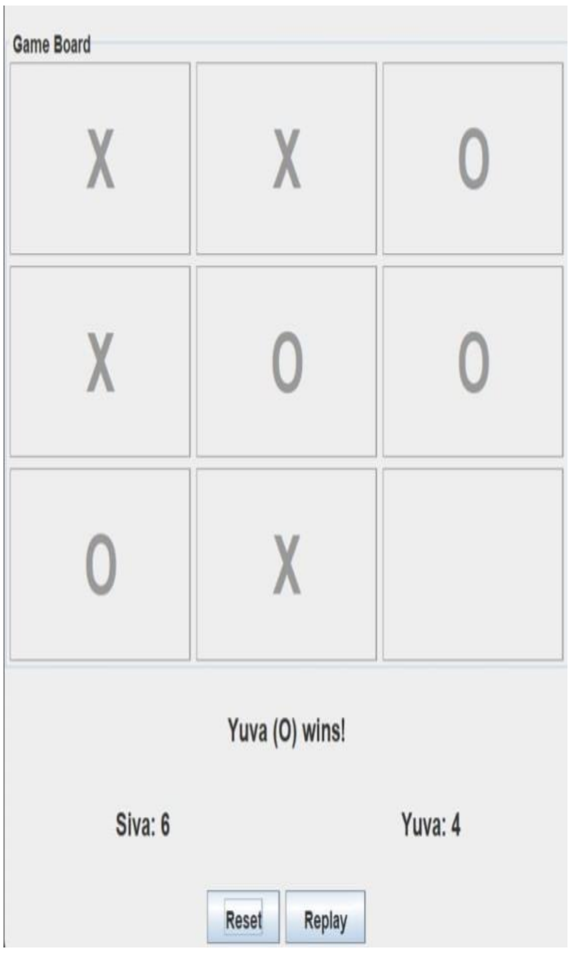
5) SCORE BOARD


Game Board

X	O	X
X	O	O
X		

Player 1 (X) wins!

Player 1: 1   Player 2: 0

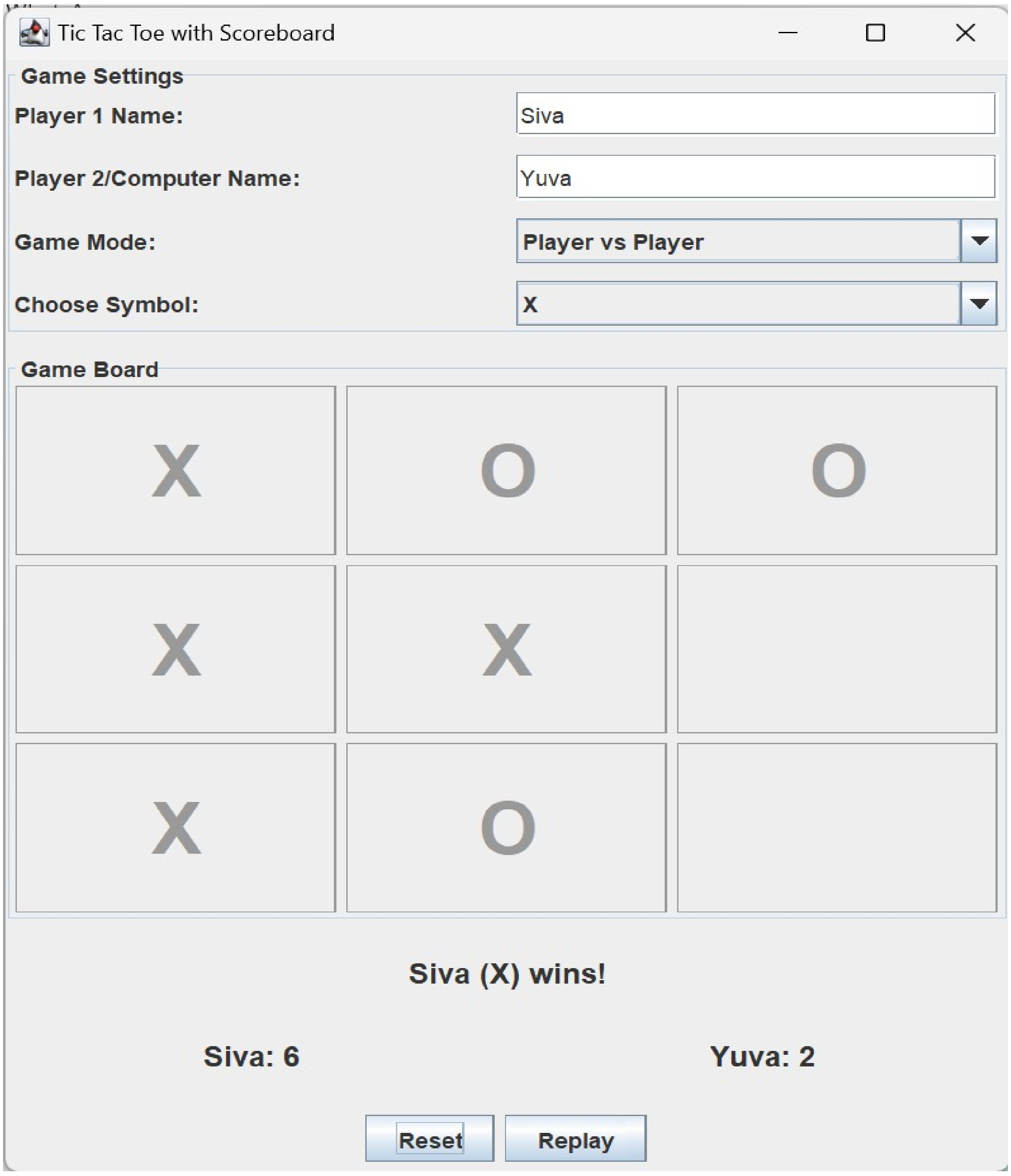

Game Board

X	X	O
X	O	O
O	X	

Yuva (O) wins!

Siva: 6   Yuva: 4

6) X WINS VERTICALLY AND HORIZONTALLY


Tic Tac Toe with Scoreboard

Game Settings

Player 1 Name: Siva

Player 2/Computer Name: Yuva

Game Mode: Player vs Player

Choose Symbol: X

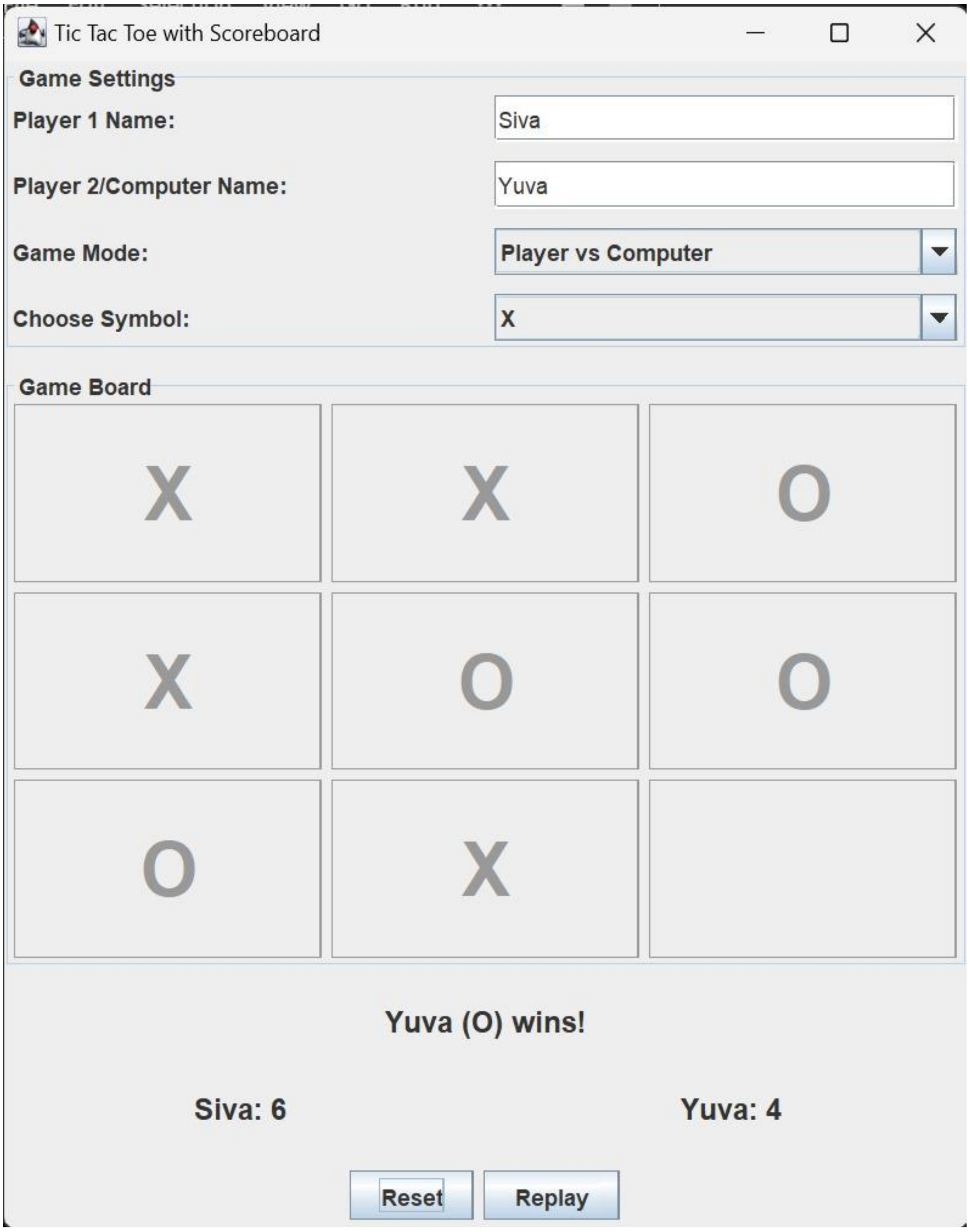
Game Board

X	O	O
X	X	
X	O	

Siva (X) wins!

Siva: 6 Yuva: 2

7) O WINS DIAGONALLY


Tic Tac Toe with Scoreboard

Game Settings

Player 1 Name: Siva

Player 2/Computer Name: Yuva

Game Mode: Player vs Computer

Choose Symbol: X

Game Board

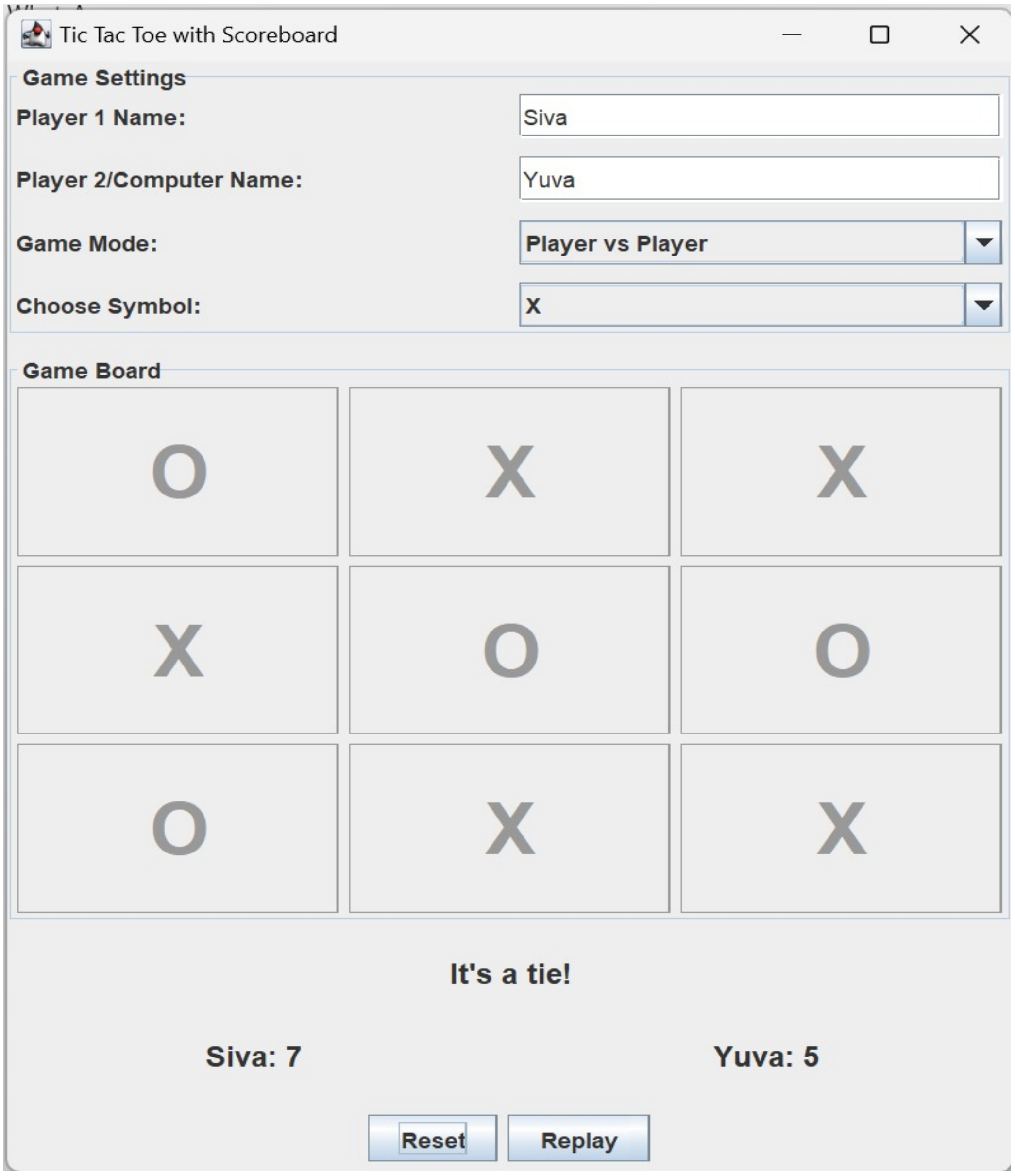
X	X	O
X	O	O
O	X	

Yuva (O) wins!

Siva: 6

Yuva: 4

8) TIE MOMENT


Tic Tac Toe with Scoreboard

Game Settings

Player 1 Name: Siva

Player 2/Computer Name: Yuva

Game Mode: Player vs Player

Choose Symbol: X

Game Board

O	X	X
X	O	O
O	X	X

It's a tie!

Siva: 7 Yuva: 5

REFERENCES

1. Sharma, Rahul, and Suresh Kumar. "Design and Implementation of Tic Tac Toe Game Using Java Swing and AWT." International Journal of Computer Science and Information Technologies (IJCSIT), vol. 12, no. 3, 2024. Design of Interactive Game Applications Using Java Swing.

2. Gupta, Richa, and Amit Sharma. "Interactive Game Development: A Study of Tic Tac Toe Game Using Java." International Journal of Engineering and Technology (IJET), vol. 9, no. 4, 2023. Game Development and User Interaction with Java.

3. Patel, Kiran, et al. "Design and Implementation of a Strategic AI for Tic Tac Toe: A Java-based Approach." Journal of Software Engineering and Applications, vol. 30, 2023. Enhancing AI Logic in Digital Games Using Java.