CSM 216 FINAL PROJECT REPORT

Project Title:

TIC TAC TOE

Submitted By:

Kartik

Registration Number:

12319010

Section:

K23CH

Submitted To:

Aman Kumar Sir

Date: 23-11-2024

Acknowledgment

I would like to express my heartfelt gratitude to everyone who supported and guided me throughout the development of this project, the TIC TAC TOE.

First and foremost, I would like to thank my mentor, Aman Sir, for their invaluable advice and guidance, which provided the foundation and direction necessary to successfully complete this project. Their insights and constructive feedback were instrumental in refining the application's functionality and ensuring that it met the project objectives.

I would also like to extend my gratitude to my colleagues and peers who shared their expertise and provided thoughtful suggestions for enhancing the application's features. Their contributions played an essential role in helping me identify potential improvements and explore effective solutions to common challenges in task management.

Additionally, I am grateful to Lovely Professional University for providing a supportive environment and resources that enabled me to focus on and accomplish the project goals. Access to tools and platforms was critical in designing, implementing, and testing this application effectively.

Lastly, I would like to acknowledge the encouragement and patience of my family and friends, who supported me during this project's development. Their unwavering belief in my capabilities motivated me to remain dedicated to delivering a high-quality, user-friendly application.

Table of Contents

Acknowledgment.....2

Introduction.....4

Objectives and Scope of the Project.....4

A. Objectives: 5

B. Scope of the Project: 5

Application Tools.....6

A. Programming Language: Python 6

B. Integrated Development Environment (IDE): PyCharm 6

C. Libraries and Packages 7

D. Version Control: Git.....7

E. Additional Tools and Resources 7

Project Design.....8

Game Initialization.....8

Player Moves and Game State Management.....8

AI Logic and Minimax Algorithm.....8

Scoring and Round Management.....9

User Interface.....9

Flowchart.....10

Project Screenshot 13

Testing and Validation.....18

Project Implementation.....21

Conclusion 23

Refrences.....24

Introduction:

This report presents the development of a Tic Tac Toe game with a modern twist, designed using Python's Tkinter library for an interactive graphical user interface. Tic Tac Toe, also known as "Noughts and Crosses," is a timeless game that requires players to strategically place their symbols (X or O) in a 3x3 grid, aiming to align three symbols in a row, column, or diagonal to secure a win. Traditionally a two-player game, this digital version enhances the classic gameplay with two modes: Player vs. Player (PvP) and Player vs. AI.

In addition to allowing users to play against each other or the computer, this version introduces a best-of-three scoring system, where the first player to win two out of three rounds is declared the overall winner. An integrated scoreboard tracks scores and visually updates after each round, adding a sense of progression and competitiveness. The AI mode uses a minimax algorithm to offer a challenging opponent, capable of making optimal moves that test the player's skills.

With intuitive design and easy-to-follow gameplay, this Tic Tac Toe game combines classic mechanics with enhanced features to deliver an engaging and polished gaming experience. This report explores the structure, code implementation, and design considerations of the project, highlighting how modern programming techniques can elevate a simple yet strategic game into an interactive digital experience.

Objectives and Scope of the Project:

A. Objectives:

The main objectives of this Tic Tac Toe project are:

- To implement a digital version of the classic Tic Tac Toe game that offers both Player vs. Player (PvP) and Player vs. AI modes.
- To create an intuitive and user-friendly graphical interface using Python's Tkinter library.
- To introduce a best-of-three rounds feature, where players compete to win two out of three rounds for an overall victory.
- To integrate a scoreboard that tracks each player's score, providing a competitive and engaging experience.
- To ensure the game is accessible to players of all ages through a simple, clean design and straightforward functionality.

B. Scope of the Project:

1. **Target Users:**
    - The Tic Tac Toe game is intended for players of all ages who enjoy classic board games. It is particularly aimed at families, friends, and casual gamers looking for a quick and fun competition. The game's design allows for easy operation, making it accessible to users with minimal technical experience.
2. **Functional Scope:**
    - **Game Modes:** The application includes two game modes—3x3 PvP and 3x3 Player vs. AI. Players can select either mode from the main menu.
    - **Rounds and Scoring System:** The game follows a best-of-three rounds format. A scoreboard is displayed throughout gameplay to show the scores of each player, with an overall winner determined after two rounds are won by one player.
    - **AI Algorithm:** For the AI mode, a minimax algorithm is used to make optimal moves, offering a challenging experience when a player chooses to play against the computer.

- **Game Management:** The application includes features such as round resets, score resets, and winner announcements, ensuring a smooth and organized gameplay experience.

3. **Exclusions:**

   - This initial version focuses solely on the classic 3x3 grid format and does not include larger grid options like 4x4 or 5x5. Additional variations, such as customizable player symbols or sound effects, are also outside the current scope.

Application Tools:

This project relies on various tools and technologies that contribute to its functionality, user interface, and overall experience. Below is a detailed breakdown of the tools and technologies used:

A. Programming Language: Python

- **Python** was chosen due to its versatility, simplicity, and rich ecosystem of libraries. Python's extensive support for GUI and data handling libraries made it ideal for developing a game with interactive elements and intuitive flow. Additionally, Python's readability ensures that the project can be easily maintained and expanded upon in the future.

B. Integrated Development Environment (IDE): PyCharm

- **PyCharm** was used as the primary IDE for developing the Tic Tac Toe game. Its powerful debugging tools, syntax highlighting, and code management features significantly streamlined the coding process, helping to identify and resolve issues efficiently. PyCharm's version control integration allowed for organized code versioning, making it easier to track changes and revert if necessary.

C. Libraries and Packages:

Several Python libraries and packages were essential for building this project:

1. **Tkinter:**

    - Tkinter was used to design the graphical user interface, providing an easy-to-implement framework for creating buttons, labels, and game grids. Tkinter's event-driven architecture was key to handling player interactions within the game.

2. **Random:**
    - The random module was employed to introduce a slight element of unpredictability in the AI's decisions when there are equally optimal moves, giving the AI a more human-like playing style.
3. **Messagebox (from Tkinter):**
    - Tkinter's messagebox module was used to display pop-up notifications, alerting players to game results, round completions, and final score announcements in an engaging way.

Version Control: Git

- Git was utilized to manage project versions, allowing for efficient tracking and documentation of changes. Through Git, the project maintained a detailed history, facilitating easy rollbacks to previous versions and enabling potential collaboration with other developers. Frequent commits ensured each development stage was documented, and code quality could be maintained throughout the process.

GitHub Link for the Project:

https://github.com/Kartik070704/tic-tac-app-in-dual-mode.git

Project Design

The Tic Tac Toe game is designed with modular components, each focused on a specific aspect of functionality. This section outlines the key components, along with their purposes and interactions.

1. Game Initialization (Start and Setup Functions):

- **Purpose:** Manages the initialization and setup of the game board based on the selected mode.
- **Key Functions:**
    - start_game(): Initiates a new game, sets up the board, and resets the scoreboard.
    - initialize_board(): Clears the board for a new game and initializes buttons for player interactions.
    - show_main_menu(): Displays the main menu for mode selection (PvP or AI).

2. Player Moves and Game State Management (Gameplay Functions):

- **Purpose:** Handles player inputs, AI moves, and board updates.
- **Key Functions:**
    - on_button_click(): Registers player moves, updates the board, and checks for a winner.
    - ai_move(): Executes the AI's move using the minimax algorithm to make optimal decisions.
    - check_winner(): Evaluates the board state to determine if there is a winner or a draw.

3. AI Logic and Minimax Algorithm (AI Functions):

- **Purpose:** Implements AI behavior to create a challenging opponent.
- **Key Functions:**

- minimax(): A recursive function that evaluates potential moves and returns the best possible outcome for the AI, ensuring competitive play.
- calculate_best_move(): Identifies the optimal move by applying the minimax algorithm to the current board state.

4. Scoring and Round Management (Scoring Functions):

- **Purpose:** Manages the scoring system and tracks the winner across multiple rounds.
- **Key Functions:**
  - update_scoreboard(): Updates the on-screen scoreboard with the latest scores after each round.
  - reset_scores(): Resets scores for a new series of games.
  - check_overall_winner(): Declares the overall winner based on best-of-three scoring.

5. User Interface (UI Functions):

- **Purpose:** Handles the display and user interaction with the graphical interface.
- **Key Functions:**
  - display_welcome_screen(): Shows the welcome screen and allows players to select the game mode.
  - display_scoreboard(): Shows the scoreboard and updates it with player scores.
  - show_game_result(): Displays pop-up messages at the end of each game or round to inform players of the outcome.

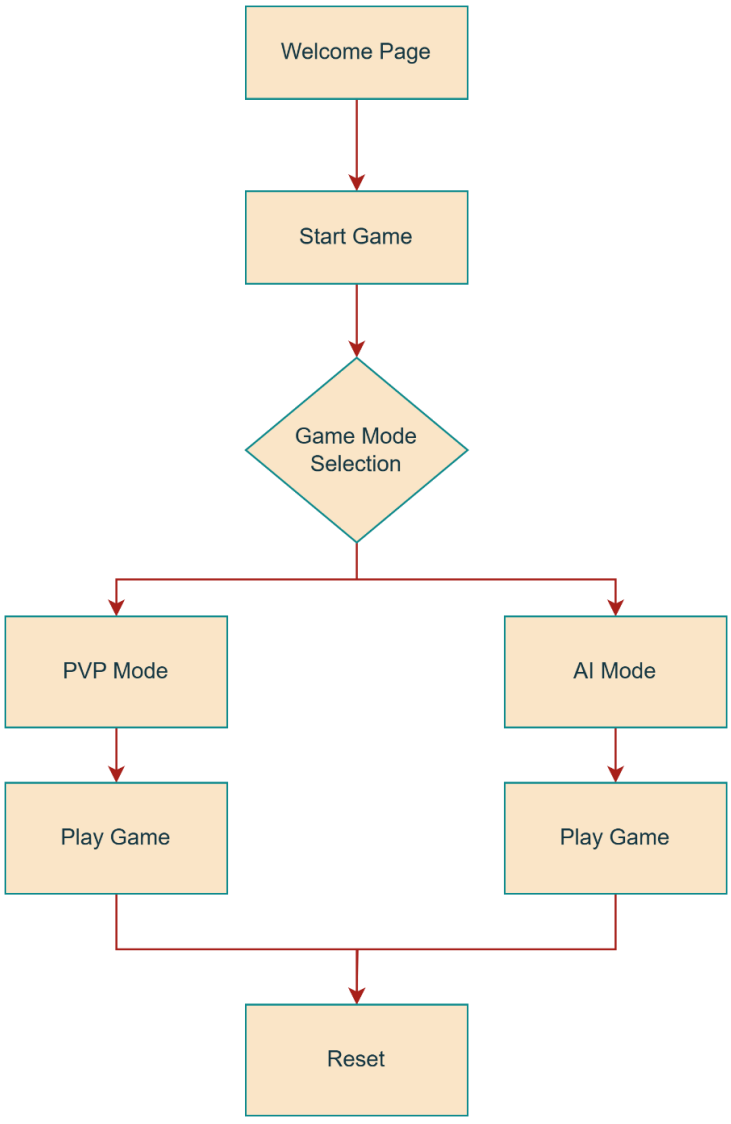
Interaction of Components

- **Gameplay Loop**: The user selects a mode, and the board is initialized. Each player takes turns (or the AI makes its move in AI mode), and after each move, the board

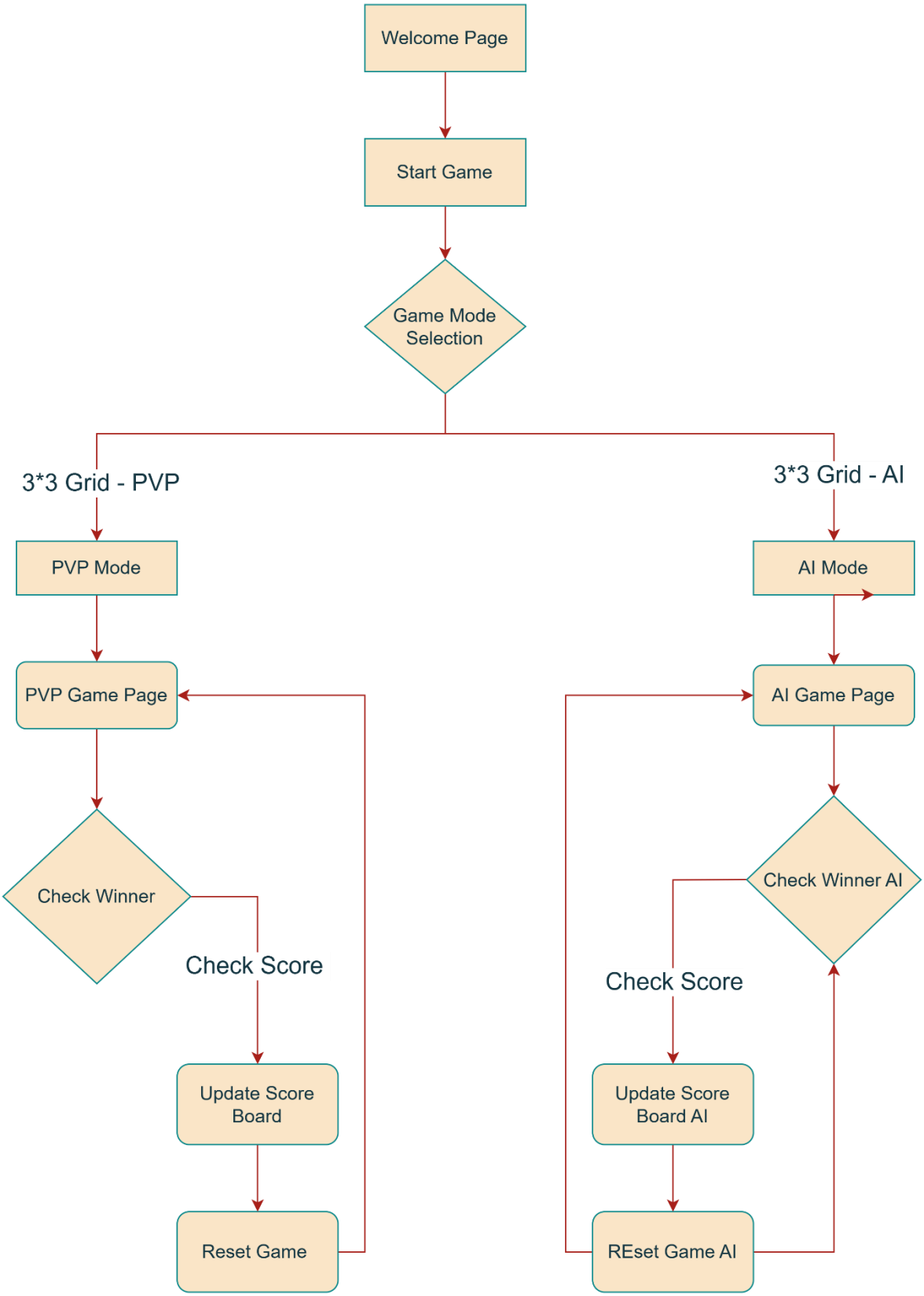
state is checked for a win or draw. If a round is won, the scores are updated. After three rounds, the final winner is declared.

- **Score Management**: Each player's score is tracked and displayed. The first player to win two rounds is crowned the overall game winner.
- **AI Interaction**: In AI mode, the AI's move is calculated based on board evaluation,adding depth to the gameplay experience.

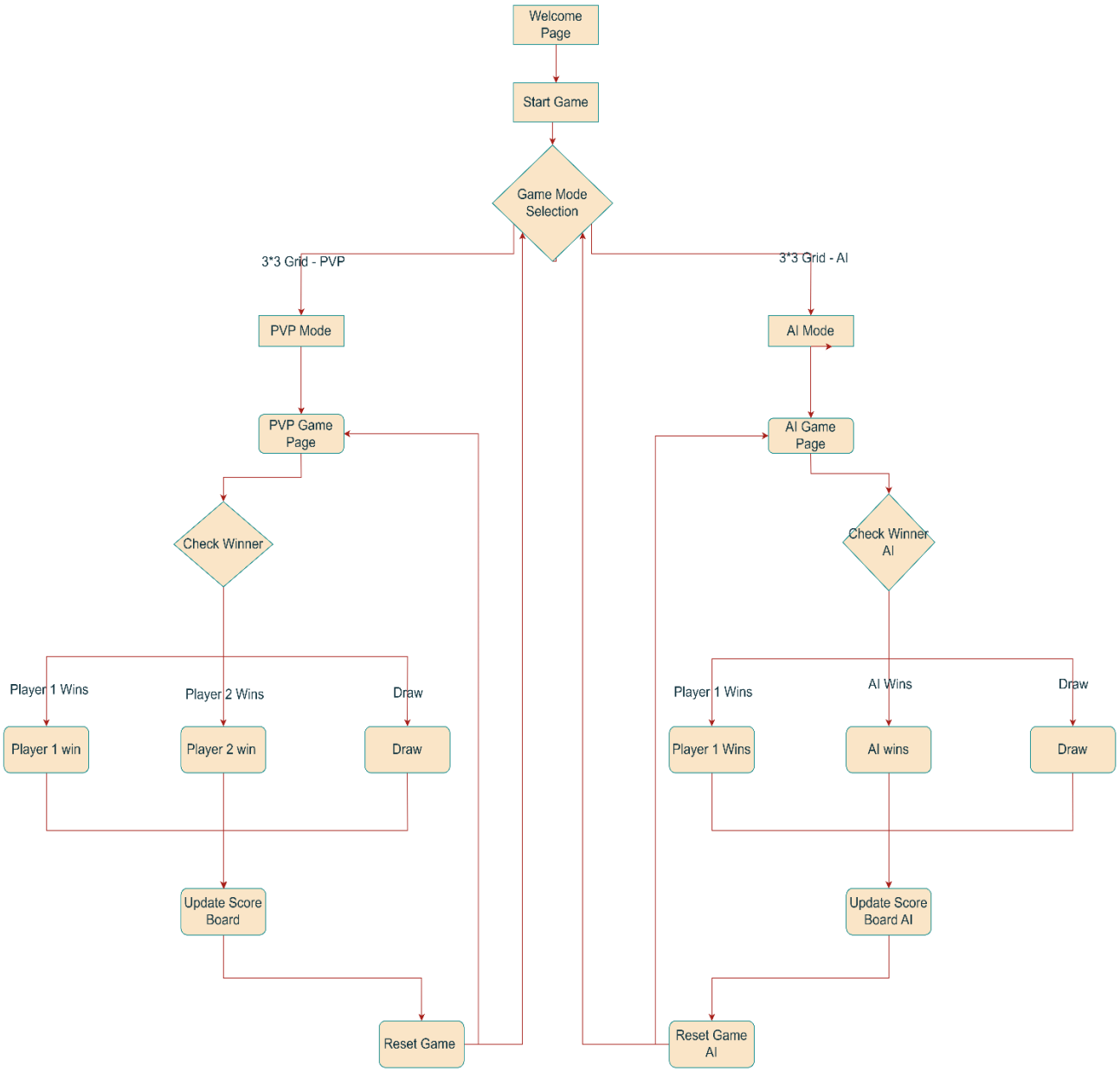
DFD`s: TIC TAC TOE:



Level 0 DFD Tic Tac Toe

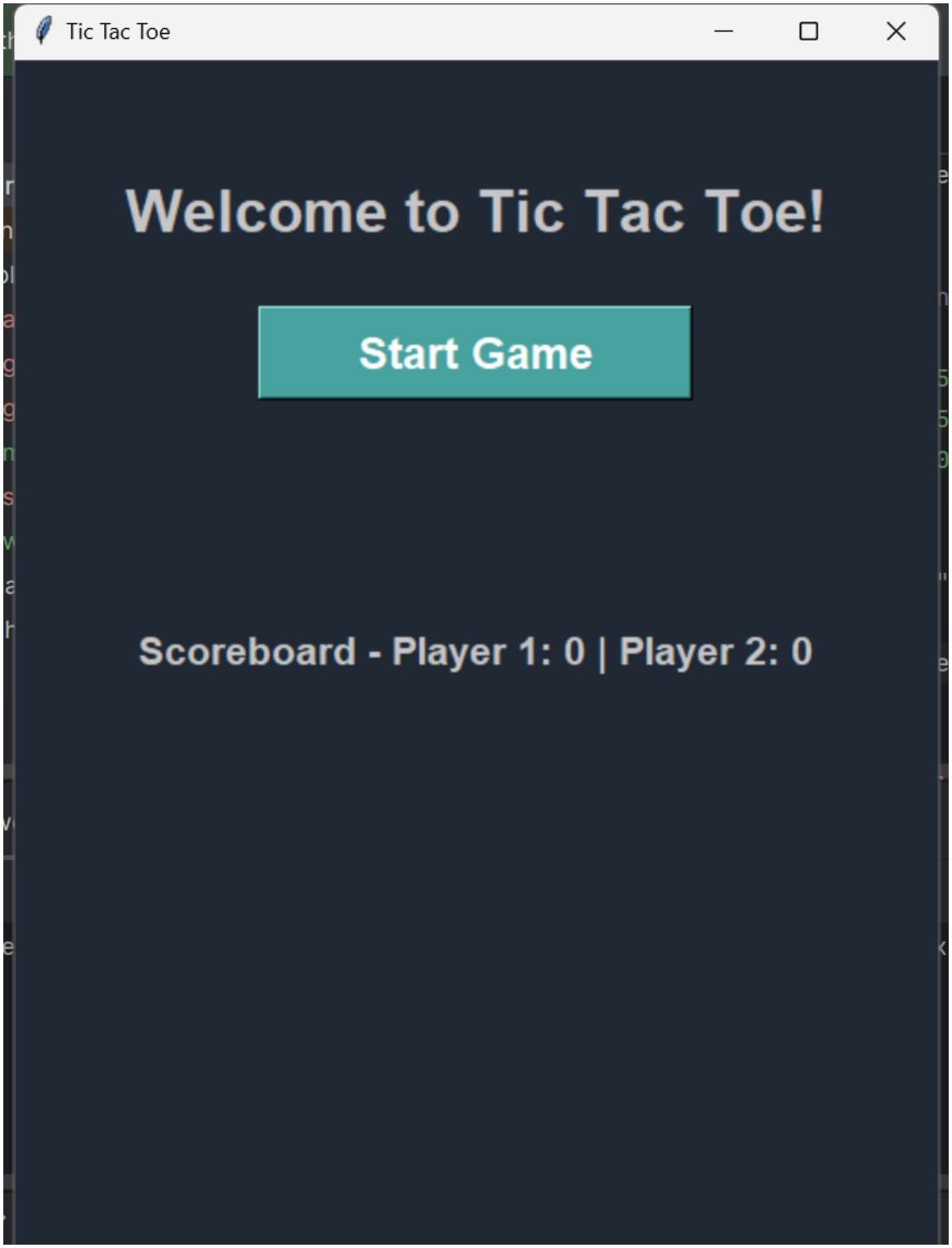


Level 1 DFD Tic Tac Toe

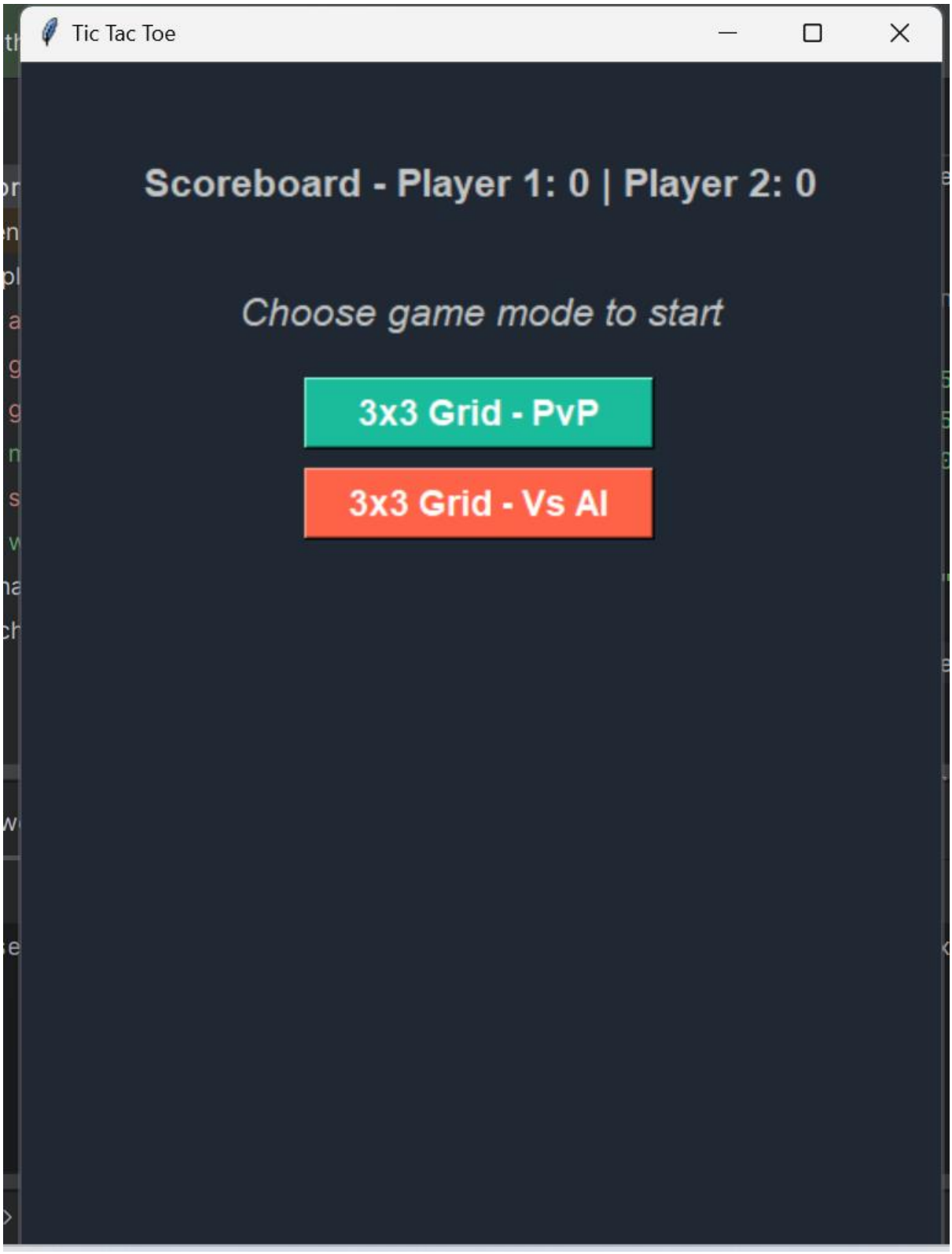


Level 2 DFD Tic Tac Toe

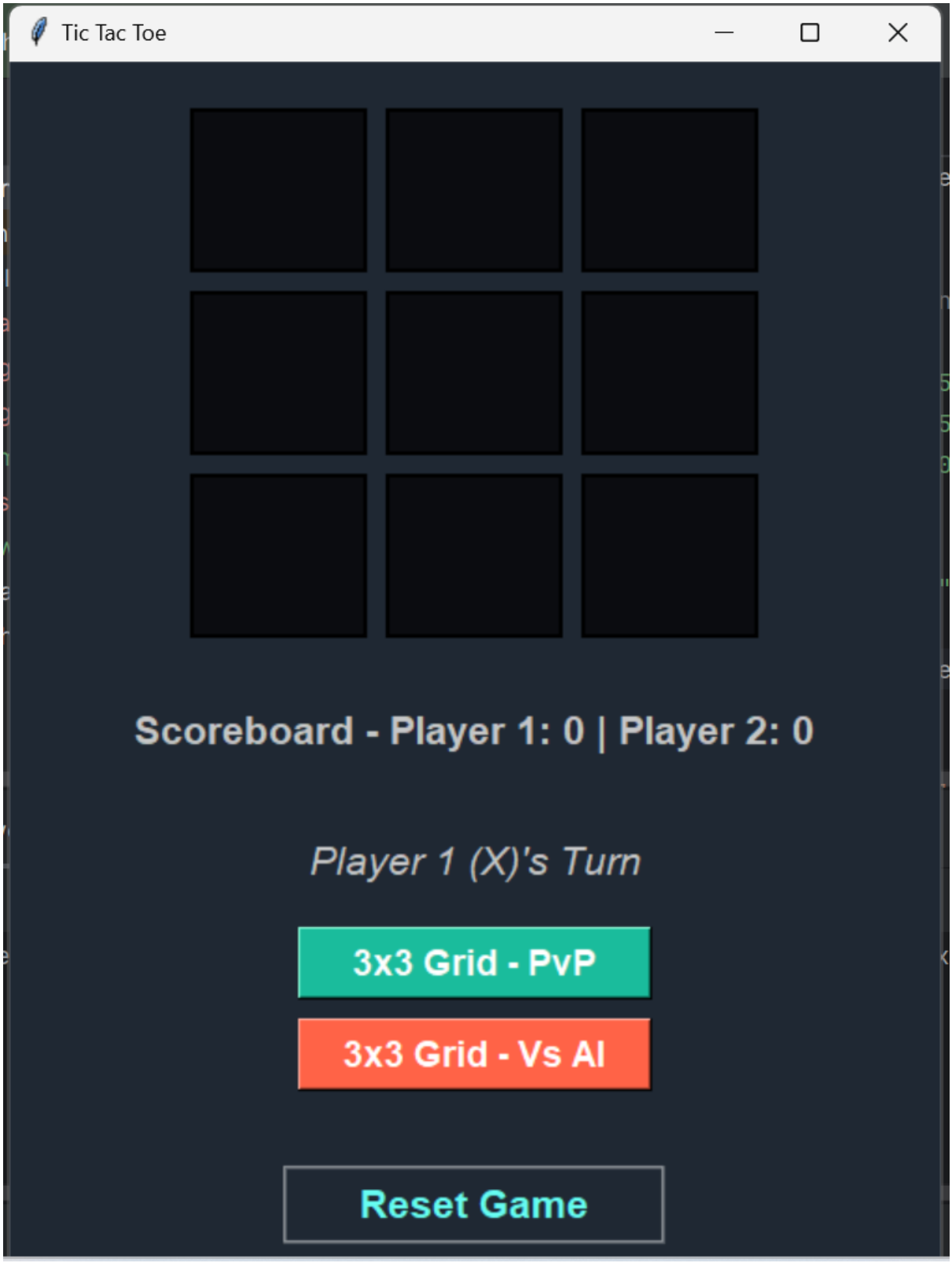
Screenshot Of Project :



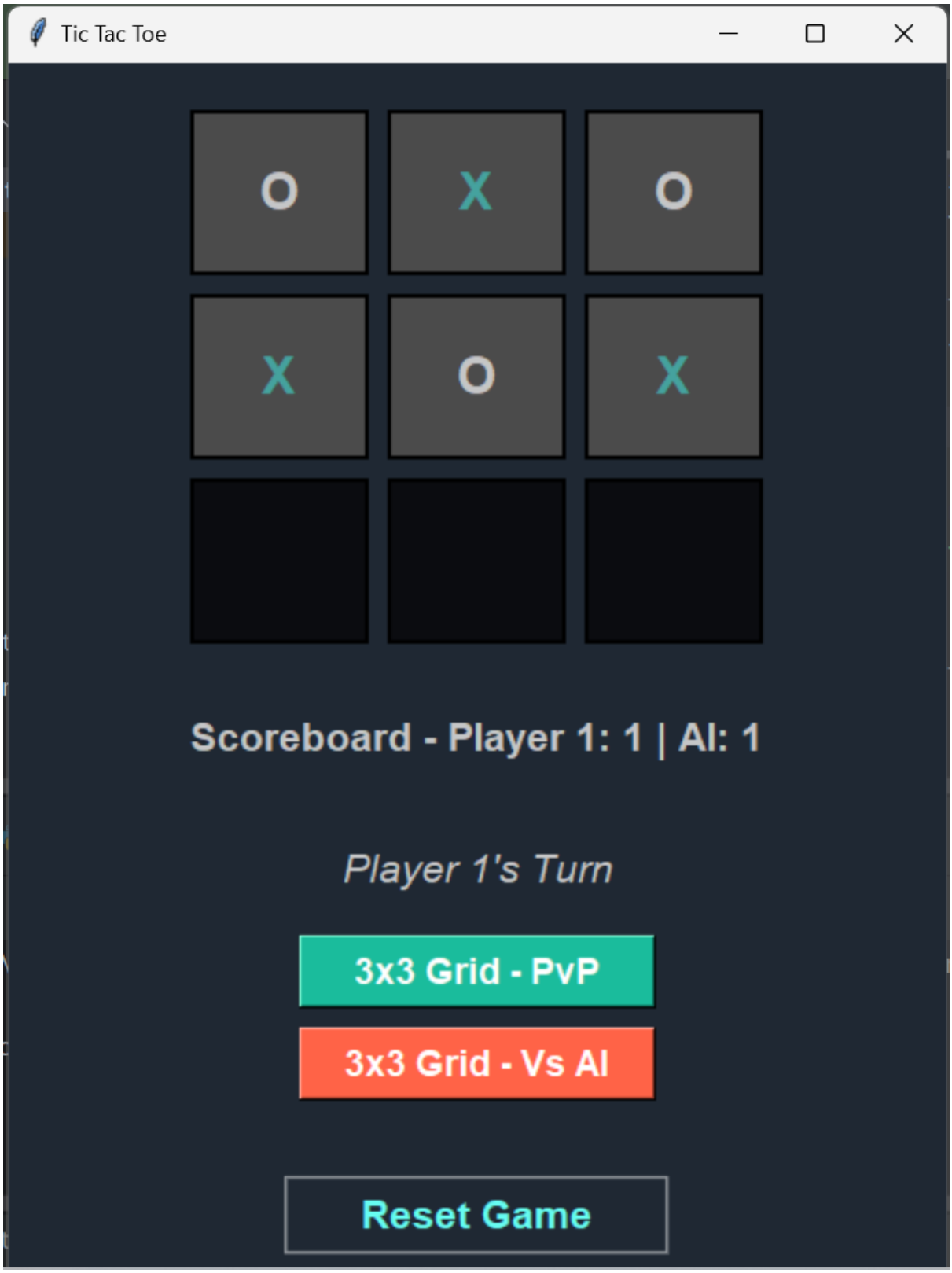
Welcome page: (fig-1)



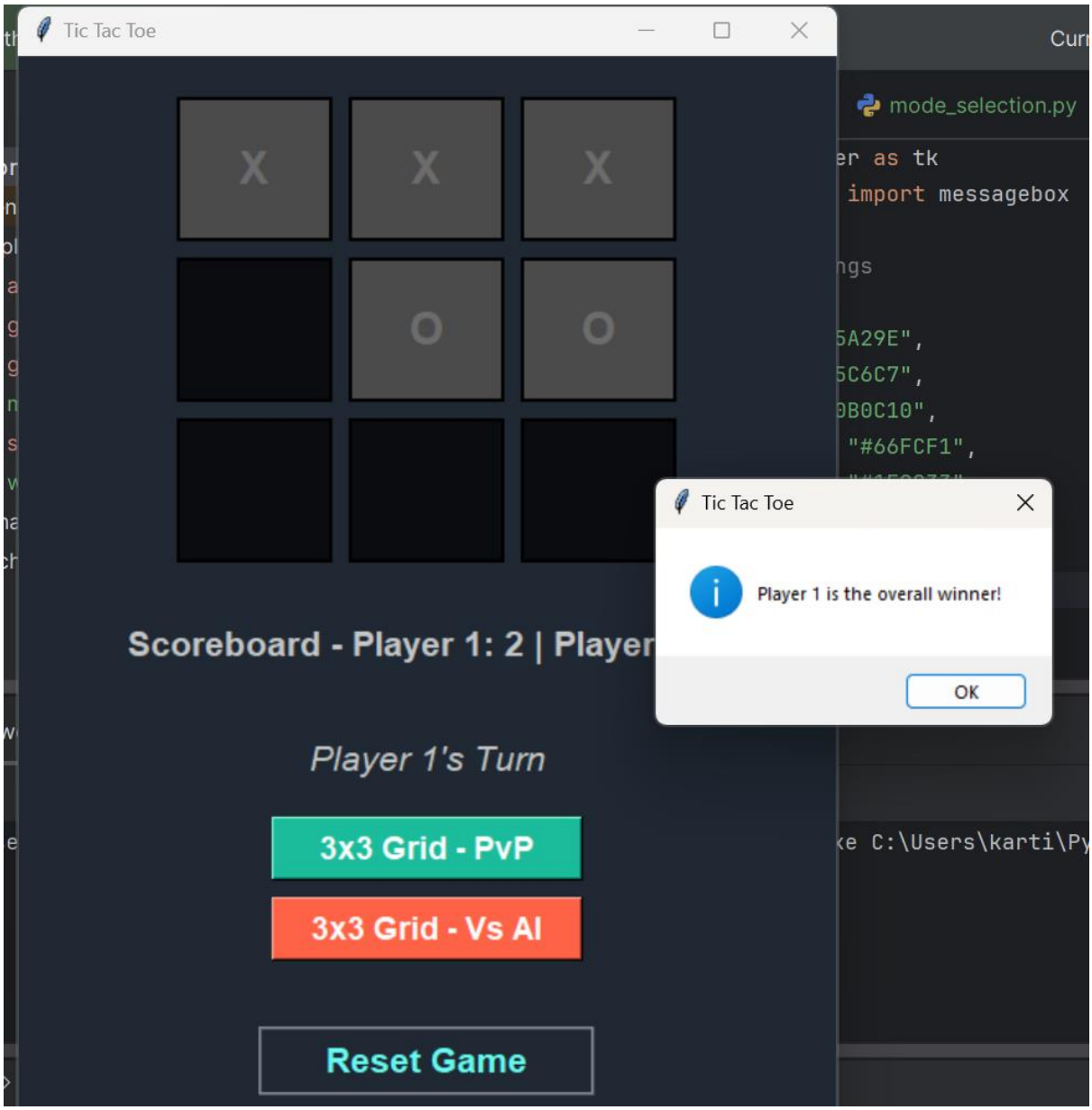
Selection page: (fig-2)



Game page: PVP Mode (fig-3)



Game Page: AI Mode (fig-4)



Score Board And Declaration of Winner: (fig-5)

Testing and Validation:

Game Logic Tests-

Test Case ID	Tested Function	Input Scenario	Expected Output	Validation Criteria
TC-001	check winner	Board state: ["X", "X", "X", "", "", "", "", "", "", ""]	Winner: "X"	Detects a winning row and returns "X"
TC-002	check winner	Board state: ["O", "", "", "O", "", "", "O", "", "", ""]	Winner: "O"	Detects a winning column and returns "O"
TC-003	check winner	Board state: ["X", "", "", "", "X", "", "", "", "X"]	Winner: "X"	Detects a winning diagonal and returns "X"
TC-004	check winner	Board state: ["X", "O", "X", "X", "X", "O", "O", "X", "O"]	Result: "Draw"	Returns "Draw" when the board is full, and no winner exists
TC-005	check winner	Board state: ["X", "O", "", "X", "", "", "", "", "", ""]	Result: None	Returns None when no winner exists, and the board is not full.

Player and Turn Management Tests-

Test Case ID	Tested Function	Input Scenario	Expected Output	Validation Criteria
TC-001	on_button_click	Player clicks an empty cell (e.g., index 0), current_player is "X"	Updates board at index 0 to "X" and switches turn to "O"	Board and current_player are updated correctly
TC-002	on_button_click	Player clicks an already occupied cell (e.g., index 0 containing "X")	No action taken, turn remains with the same player	Board and current_player remain unchanged
TC-003	Turn Alternation	Sequence of player moves: on_button_click(0), on_button_click(1)	Board updates to ["X", "O", "", "", "", "", "", "", ""]	Turns alternate correctly between "X" and "O"

AI Behaviour Tests-

Test Case ID	Tested Function	Input Scenario	Expected Output	Validation Criteria
TC-001	ai_move	Board state: ["X", "", "", "", "O", "", "", "", ""] (AI's turn)	AI places "O" in the optimal position (e.g., index 1)	AI places "O" in the best move determined by the minimax algorithm

TC-002	AI Optimal Move	Board state: ["X", "X", "", "", "O", "", "", "", ""], AI's turn	AI blocks the win by placing "O" in index 2	The AI uses the minimax algorithm to block the player from winning
TC-003	minimax	Board state: ["X", "O", "X", "X", "O", "O", "", "", ""], maximizing turn	Returns a valid score for the board state (e.g., 1 for a win, -1 for a loss, 0 for a draw)	The minimax algorithm evaluates the board correctly and returns the optimal score

Game Initialization and Reset Tests:

Test Case ID	**Tested Function**	**Input Scenario**	**Expected Output**	**Validation Criteria**
TC-001	reset game	After a completed round, board state: ["X", "O", "X", "O", "X", "O", "O", "X", "X"]	Resets board to ["", "", "", "", "", "", "", "", ""]. Sets current_player to "X"	The board is cleared, turn is reset to Player 1 ("X"), and buttons are reset to default state
TC-002	start game	Mode: "PvP", board is initially empty	Initializes board, resets scores, and displays the grid	All UI elements (grid, buttons, labels) are reset and ready for a new game

TC-003	start game	Mode: "AI", board is initially empty	Initializes board, resets scores, and AI logic is set up	Similar to "PvP", but enables AI logic for "O" player moves

Project Implementation:

```
import tkinter as tk
import random
from game_utils import Game

def start_ai_game(grid_size):  2 usages
    game = Game(grid_size=grid_size, ai_mode=True)
    game.setup_grid()

def ai_move(game):
    best_score = -float('inf')
    best_move = None

    for move in game.get_empty_indices():
        game.make_move(move)
        score = minimax(game, depth: 0, is_maximizing: False)
        game.undo_move()

        if score > best_score:
            best_score = score
            best_move = move

    game.make_move(best_move)

def minimax(game, depth, is_maximizing):  3 usages
    if game.is_game_over():
        return game.evaluate()
```

AI Bot Page (fig-6)

```python
import tkinter as tk

class ScoreCount:  2 usages
    def __init__(self):
        self.player1_score = 0
        self.player2_score = 0

    def update_score(self, winner):  1 usage
        if winner == "X":
            self.player1_score += 1
        elif winner == "O":
            self.player2_score += 1

    def display_score(self, root):  1 usage
        score_label = tk.Label(root, text=f"Player 1 (X): {self.player1_score} | Player 2
                               font=("Helvetica", 16), bg="#1F2833", fg="#C5C6C7")
        score_label.pack(pady=10)
```

Score Page (fig-7)

1). The **AI Game Module** adds a smart layer to Tic Tac Toe, simulating human-like decision-making:

1. **Start Game**: Initializes the grid and sets AI mode for solo play.
2. **AI Move**: The AI evaluates all possible moves to select the best one, blocking or winning.
3. **Minimax Algorithm**: It predicts moves, calculating the best strategy to win or block, just like a human opponent.

This module ensures the AI feels like a strategic, intelligent player, adapting to your moves (fig-6).

2). The provided Python code defines a class called **ScoreCount**. This class is designed to track and display scores for a game with two players, "X" and "O".

The code uses the tkinter library to create a simple graphical interface (GUI) for displaying the score. The **display_score** method is responsible for formatting and displaying the score information in a visually appealing way (fig-7).

Conclusion:

This Tic Tac Toe project demonstrates a well-rounded implementation of a simple yet engaging game, incorporating both player-vs-player (PvP) and player-vs-AI modes. The core mechanics, including the game flow, user interface, and AI behavior, are designed to provide an intuitive and enjoyable experience.

1. **Game Functionality**: The game allows users to choose between PvP and PvAI modes, where the board resets after every round, and scores are tracked across multiple rounds. The clear, colorful interface enhances user interaction.
2. **AI Implementation**: The AI, powered by the minimax algorithm, simulates a skilled opponent. By evaluating all possible moves and selecting the optimal one, the AI provides a challenge for the player, making each game unpredictable and exciting.
3. **User Interface**: A simple and responsive Tkinter-based UI ensures ease of play, with buttons that react to hover and click events. A visually appealing design keeps the user engaged throughout the game.
4. **Extensibility and Scalability**: The project is modular, allowing easy extensions such as adding different grid sizes (e.g., 4x4 or 5x5) or improving the AI's strategy for a more difficult challenge.
5. **Unit Testing**: Thorough testing has ensured that the game mechanics, AI logic, and reset functionalities perform as expected, confirming the robustness of the code.

In conclusion, this Tic Tac Toe project showcases a blend of solid game logic, user-friendly design, and strategic AI integration, offering an engaging and enjoyable experience for users of all ages. The structure allows for further enhancements, ensuring its adaptability and continued relevance.

References:

1. **Tkinter Documentation**:

   Tkinter is the Python library used for the graphical user interface (GUI). For more details on Tkinter and its widgets (like Button, Label, and Frame), you can refer to the official documentation:

   - Python's Tkinter Documentation: https://docs.python.org/3/library/tkinter.html

2. **Minimax Algorithm**:

   The AI logic for the Tic Tac Toe game uses the minimax algorithm to evaluate the best possible move. The following sources explain the concept of minimax in the context of game theory:

   - Wikipedia article on Minimax Algorithm: https://en.wikipedia.org/wiki/Minimax
   - GeeksforGeeks explanation of Minimax: https://www.geeksforgeeks.org/minimax-algorithm-in-game-theory-set-1-introduction/

3. **Unit Testing in Python**:

   Unit testing is crucial for ensuring the correctness of the code. The following resources provide information on how to write and run unit tests in Python using the unittest framework:

   - Official Python unittest documentation: https://docs.python.org/3/library/unittest.html
   - GeeksforGeeks article on Unit Testing in Python: https://www.geeksforgeeks.org/unit-testing-python-unittest/

4. **Game Design Principles**:

   The overall game design, including the implementation of user interfaces and player interactions, is based on standard principles of game development:

   - Game Design Principles: https://www.gamasutra.com/view/feature/130914/

- Game Development: https://www.tutorialspoint.com/game_development/

5. **AI in Games**:
   The project utilizes AI to play against the user. The use of algorithms like minimax in games is a well-explored topic in AI game development:
   - "Artificial Intelligence for Games" by Ian Millington is a book that details how AI can be applied in game development.
6. **Tkinter Projects and Game Development**:
   There are numerous resources on using Tkinter for game development, including Tic Tac Toe:
   - Real Python tutorial on creating a Tic Tac Toe game using Tkinter: https://realpython.com/python-gui-tkinter/

These references support the understanding and creation of a Tic Tac Toe game using Python, Tkinter, and AI algorithms, and provide further reading for exploring similar projects.