Tic-Tac-Toe Project Report

Title and Overview

This project focuses on creating a modern version of the classic Tic-Tac-Toe game using Python. The game includes a graphical user interface (GUI) built with `tkinter` and offers two modes:

1. **Single-Player Mode:** Players compete against an AI that has three difficulty levels: Easy, Medium, and Impossible.
2. **Two-Player Mode:** Two players take turns playing locally on the same device.

The goal of this project was to build a user-friendly interface while incorporating AI functionality to provide a dynamic and engaging gameplay experience.

Problem Statement

The problem addressed by this project was to enhance a simple game like Tic-Tac-Toe by:

- Adding a GUI for better user interaction.
- Introducing AI-driven single-player modes with adjustable difficulty.
- Making the game replayable without restarting the application.

Tic-Tac-Toe is a commonly implemented game, but many existing versions lack features like difficulty levels or seamless replay. By addressing these gaps, the project aims to deliver a polished and complete version of the game.

Approach

To achieve the project goals, the following steps were taken:

1. **Game Representation:**
   The game board was represented as a 3x3 2D array. Each cell in the array could be empty (`" "`), marked with an `"X"`, or marked with an `"O"`. This structure made it easy to validate moves and check for win or draw conditions.
2. **GUI Design:**
   The GUI was designed using Python's `tkinter` library. Buttons represented the game cells, and their states were updated dynamically based on player actions. The layout was made responsive to adapt to window resizing.
3. **AI Logic for Single-Player Mode:**
   The AI was implemented with three difficulty levels:

    - **Easy Mode:** AI makes random moves.
    - **Medium Mode:** AI uses heuristics to block the player or win if possible.
    - **Impossible Mode:** AI uses the minimax algorithm, which evaluates all possible outcomes to make optimal decisions.
4. **Game State Management:**
   The game logic handled player turns, move validation, win/draw detection, and board resetting. A pause option was added, allowing players to return to the main menu mid-game.

Description of the Software

This Tic-Tac-Toe software consists of two main components: the front-end GUI and the back-end game logic.

1. **Front-End (GUI):**
    - Built using `tkinter` for a clean and simple interface.
    - Features a main menu for selecting game modes and a responsive game board.
2. **Back-End (Game Logic):**
    - Handles core functionality such as move validation, win/draw detection, and resetting the board.
    - Includes AI logic to provide dynamic gameplay in single-player mode.
3. **Input and Output:**
    - **Input:** Player actions are taken as button clicks on the GUI.
    - **Output:** The game updates the board, displays results (win/draw), and provides options to restart or return to the main menu.

Evaluation

The project was thoroughly tested in all modes to ensure functionality, responsiveness, and correctness. The results include:

- **Single-Player Mode:** The AI responded appropriately in each difficulty level, with Impossible Mode being unbeatable as expected.
- **Two-Player Mode:** The game correctly alternated turns and detected winners or draws without issues.
- **Replayability:** The game allowed seamless replay without needing to restart the application.
- **GUI Responsiveness:** The interface adjusted well to resizing and provided clear feedback for player actions.

Diagram of Components

The software consists of three main components:

1. **Main Menu:** Serves as the entry point for game mode selection.
2. **Game Board:** Displays the 3x3 grid and manages gameplay logic.
3. **AI Logic:** Processes moves for the single-player mode based on the chosen difficulty level.

Flow of Interaction:

- Players select a game mode from the main menu.
- The game board updates based on player or AI moves.
- The game detects win/draw conditions and allows the player to restart or quit.

Conclusion and Future Work

Conclusion:
This project successfully delivered a functional and polished Tic-Tac-Toe game with an intuitive GUI, AI, and seamless replayability. It demonstrated the integration of GUI design, AI algorithms, and game logic into a cohesive program.

Future Work:

1. Add network multiplayer functionality to allow remote gameplay.
2. Use advanced GUI frameworks like PyQt for a more modern interface.
3. Introduce score tracking across multiple rounds.
4. Experiment with machine learning to create a dynamic, adaptive AI.

References

1. Python's `tkinter` documentation for GUI development: https://docs.python.org/3/library/tkinter.html
2. Online resources for the minimax algorithm, which were instrumental in implementing the Impossible Mode AI.
3. Community forums like Stack Overflow for debugging and solving specific `tkinter` and game logic challenges.