Assignment: Design and Implement a 3x3 Tic-Tac-Toe Game in Python using Tkinter

Objective:

Design and implement a classic 3x3 Tic-Tac-Toe game using Python's Tkinter library. This assignment is structured with three levels—Easy, Intermediate, and Advanced—so you can enhance your game based on your skill level.

Instructions by Level:

1. Easy Level

In this level, create a basic Tic-Tac-Toe game with the following features:

- **Grid Layout**: Set up a 3x3 grid using Tkinter's buttons or labels for each cell.
- **Player Interaction**: Allow two players to play the game by taking turns, marking each cell with "X" or "O".
- **Winning Logic**: Implement simple win detection to check for three "X" or "O" marks in a row, column, or diagonal.
- **End of Game**: Display a message indicating the winner or if the game is a draw. Include a reset button to restart the game.

Suggested Tkinter Widgets: `Button`, `Label`, `Frame`.

2. Intermediate Level

In addition to the easy-level requirements, add these features:

- **Automatic Scorekeeping**: Display the score for each player (number of wins) and update it after each game.
- **Game Restart Option**: Implement a "New Game" button that clears the board while keeping the current score.
- **Simple UI Enhancements**: Use colors, font styles, and Tkinter layout options to improve the game's visual appeal.

Suggested Tkinter Concepts: `StringVar` for text updating, grid layout management, and `config` for styling.

3. Advanced Level

Build on the intermediate level by adding more complex features:

- **Single Player Mode with AI**: Allow the user to choose between playing against another person or an AI opponent. Implement basic AI using random moves or a more advanced strategy such as the minimax algorithm.
- **Improved UX/UI**: Create a dynamic interface that offers options for player vs. player or player vs. AI, and provides a menu or settings panel.
- **Animations and Effects**: Add subtle animations (e.g., highlight winning cells) and sound effects to enhance the experience.

Suggested Tkinter Enhancements: Use canvas elements for custom designs, and explore Tkinter's `after()` method for timing animations.

Deliverables:

- A Python file containing your Tkinter Tic-Tac-Toe code.
- A brief explanation (1-2 paragraphs) describing which level you completed and any unique design choices you implemented.
- (Optional) Screenshot(s) showing the final interface and gameplay.

Grading:

The assignment will be graded on:

- Functionality: Completion of features based on the selected level.
- Code Quality: Clear, well-organized, and well-commented code.
- Creativity: Unique design, UI enhancements, and overall presentation.
- Effort Level: Attempting intermediate or advanced levels will earn extra points.

Good luck, and have fun building your Tic-Tac-Toe game!